

Modicon M262

Logic/Motion Controller

User Guide

08/2022



Table of Contents



1 Modicon M262 Logic/Motion Controller - Programming Guide. . . .	Part I
2 Modicon M262 Logic/Motion Controller - System Functions and Variables - System Library Guide.	Part II
3 Modicon M262 - CommonMotionPcrt - Library Guide.	Part III
4 Modicon M262 Logic/Motion Controller - EncoderLibrary Guide. . .	Part IV
5 Modicon M262 - MotionInterface - Library Guide.	Part V
6 Modicon M262 - SercosMaster - Library Guide.	Part VI
7 Modicon M262 - Synchronized Motion Control - Library Guide. . .	Part VII
8 Modicon M262 Logic/Motion Controller - Hardware Guide	Part VIII
9 Modicon M262 - Embedded Safety - Integration Guide.	Part IX

Modicon M262

Logic/Motion Controller

Programming Guide

EIO0000003651.08

06/2022



Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2022 – Schneider Electric. All rights reserved.

Table of Contents

Safety Information	7
About the Book	8
About the Modicon M262 Logic/Motion Controller	13
M262 Logic/Motion Controller Description	13
Modicon M262 Logic Controller	16
Modicon M262 Logic Controller	16
Modicon M262 Motion Controller	18
Modicon M262 Motion Controller	18
How to Configure the Controller	21
Configuring the Controller	21
Libraries	23
Libraries	23
Supported Standard Data Types	24
Supported Standard Data Types	24
Memory Mapping	25
Controller Memory Organization	25
Non-Volatile Memory Organization	26
RAM Memory Organization	28
NVRAM Memory Organization	30
Relocation Table	30
Tasks	33
Maximum Number of Tasks	33
Task Types	33
Task Configuration Screen	36
System and Task Watchdogs	37
Task Priorities	38
Default Task Configuration	40
Controller States and Behaviors	41
Controller State Diagram	42
Controller State Diagram	42
Controller States Description	45
Controller States Description	45
State Transitions and System Events	48
Controller States and Output Behavior	48
Commanding State Transitions	50
Error Detection, Types, and Management	57
Remanent Variables	58
Controller Device Editor	60
Controller Parameters	60
Communication Settings	61
PLC Settings	62
Services	63
Ethernet Services	64
Users Rights	69
Embedded Inputs and Outputs Configuration	79
Configuring the Fast I/Os	79
Embedded I/Os Configuration	79
Hardware Encoder Interface	83

Hardware Encoder Interface	83
Adding an Encoder	84
Encoder Motion Functions	86
Symbol Configuration Editor	88
Controller Cybersecurity	93
Security Settings Configuration with Cybersecurity Admin Expert Software	93
Expansion Modules Configuration	102
TM3 I/O Configuration General Description	102
TM3 I/O Bus Configuration	107
TMS Expansion Module Configuration	107
TM3 Expansion Module Configuration	108
Optional I/O Expansion Modules	109
Ethernet Configuration	112
Ethernet Features, Functions and Services	112
Presentation	112
IP Address Configuration	114
Modbus TCP Client/Server	119
FTP Server	120
SNMP	121
Web Server	122
Monitoring Menu	127
Diagnostic Menu	129
Maintenance Menu	134
Machine Assistant Menu	142
Firewall Configuration	142
Introduction	142
Dynamic Changes Procedure	144
Firewall Behavior	144
Firewall Script Commands	146
Industrial Ethernet	152
Industrial Ethernet Presentation	152
DHCP Server	156
Fast Device Replacement	156
Controller as a Target Device on EtherNet/IP	156
Controller as a Slave Device on Modbus TCP	175
Sercos Configuration	179
Overview of the Sercos Standard	179
Modicon M262 Logic/Motion Controller Sercos Configuration	180
Modicon M262 Motion Controller and Safety Controllers with Sercos	180
Single Wire Architecture	180
Serial Line Configuration	183
Serial Line Configuration	183
Machine Expert Network Manager	184
Modbus Manager	185
ASCII Manager	188
Modbus Serial IOScanner	190
Adding a Device on the Modbus Serial IOScanner	191
ControlChannel: Enables or Disables a Communication Channel	196

Adding a Modem to a Manager	197
SysLog Agent.....	199
OPC UA.....	201
Overview.....	201
OPC UA Overview	201
OPC UA Server Configuration.....	201
OPC UA Server Overview	201
OPC UA Server Configuration	202
OPC UA Server Symbols Configuration	208
OPC UA Server Performance.....	210
OPC UA Client Configuration	212
OPC UA Client Overview	212
Programming the OPC UA Client.....	213
Post Configuration	215
Post Configuration Presentation.....	215
Post Configuration File Management.....	216
Post Configuration Example	218
Connecting a Modicon M262 Logic/Motion Controller to a PC	221
Connecting the Controller to a PC	221
Updating Firmware	223
Updating the Controller Firmware by SD Card	223
Updating the Controller Firmware by Controller Assistant	225
Updating TM3 Expansion Modules Firmware	226
Updating TMSES4 Expansion Module Firmware	229
Managing Script Files	231
Creating a Script.....	231
Generating Scripts and Files.....	234
Transferring Scripts and Files	235
Cloning a Controller	237
Before Cloning a Controller.....	237
Cloning a Controller	238
Compatibility	240
Software and Firmware Compatibilities	240
Diagnostic.....	241
System Diagnostic	241
Diagnostic Messages	241
Machine Assistant	260
Accessing the Web Server Through Industrial Plug and Work.....	260
Launching the Web Server.....	260
Using the Machine Assistant.....	260
Launching the Machine Assistant	260
Managing the Network Scan	261
Managing the Devices Network Settings	262
Backing Up/Restoring Configuration	263
Exporting/Importing .semtdt Files	264
Appendices	265
How to Change the IP Address of the Controller	266
changeIPAddress: Change the IP address of the controller	266
Functions to Get/Set Serial Line Configuration in User Program	268
GetSerialConf: Get the Serial Line Configuration	268
SetSerialConf: Change the Serial Line Configuration	269

LinkNumber: Communication Port Number	270
SERIAL_CONF: Structure of the Serial Line Configuration Data	
Type	270
Controller Performance	272
Processing Performance	272
M262 Logic/Motion Controller Event Messages.....	274
SysLog Messages from M262 Logic/Motion Controller.....	274
Glossary	277
Index	289

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

Document Scope

The purpose of this document is to help you program and operate your Modicon M262 Logic/Motion Controller with the EcoStruxure Machine Expert software.

NOTE: Read and understand this document and all related documents before installing, operating, or maintaining your Modicon M262 Logic/Motion Controller.

The Modicon M262 Logic/Motion Controller users should read through the entire document to understand its features.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.3.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Related Documents

Title of Documentation	Reference Number
EcoStruxure Machine Expert - Programming Guide	EIO0000002854 (ENG)
	EIO0000002855 (FRE)
	EIO0000002856 (GER)
	EIO0000002857 (SPA)
	EIO0000002858 (ITA)
	EIO0000002859 (CHS)
Modicon M262 Logic/Motion Controller - Hardware Guide	EIO0000003659 (ENG)
	EIO0000003660 (FRE)
	EIO0000003661 (GER)
	EIO0000003662 (SPA)
	EIO0000003663 (ITA)
	EIO0000003664 (CHS)
	EIO0000003665 (POR)
	EIO0000003666 (TUR)

Title of Documentation	Reference Number
Modicon TM3 Expansion Modules - Programming Guide	EIO0000003119 (ENG) EIO0000003120 (FRE) EIO0000003121 (GER) EIO0000003122 (SPA) EIO0000003123 (ITA) EIO0000003124 (CHS) EIO0000003990 (POR) EIO0000003991 (CHS)
Modicon TM5 EtherNet/IP Fieldbus Interface - Programming Guide	EIO0000003707 (ENG) EIO0000003708 (FRE) EIO0000003709 (GER) EIO0000003710 (SPA) EIO0000003711 (ITA) EIO0000003712 (CHS)
Modicon TMS Expansion Modules - Programming Guide	EIO0000003691 (ENG) EIO0000003692 (FRE) EIO0000003693 (GER) EIO0000003694 (SPA) EIO0000003695 (ITA) EIO0000003696 (CHS) EIO0000003697 (POR) EIO0000003698 (TUR)
Modicon M262 Logic/Motion Controller - System Functions and Variables - System Library Guide	EIO0000003667 (ENG) EIO0000003668 (FRE) EIO0000003669 (GER) EIO0000003670 (SPA) EIO0000003671 (ITA) EIO0000003672 (CHS) EIO0000003673 (POR) EIO0000003674 (TUR)
Modicon TM3 Expert I/O Modules - HSC Library Guide	EIO0000003683 (ENG) EIO0000003684 (FRE) EIO0000003685 (GER) EIO0000003686 (SPA) EIO0000003687 (ITA) EIO0000003688 (CHS) EIO0000003689 (POR) EIO0000003690 (TUR)

Title of Documentation	Reference Number
Modicon M262 Logic/Motion Controller - Encoder Library Guide	EIO0000003675 (ENG) EIO0000003676(FRE) EIO0000003677(GER) EIO0000003678 (SPA) EIO0000003679 (ITA) EIO0000003680 (CHS) EIO0000003681 (POR) EIO0000003682 (TUR)
EcoStruxure Machine Expert - FtpRemoteFileHandling Library Guide	EIO0000002779 (ENG) EIO0000002780 (FRE) EIO0000002781 (GER) EIO0000002782 (SPA) EIO0000002783 (ITA) EIO0000002784 (CHS)
EcoStruxure Machine Expert - SnmpManager Library Guide	EIO0000002797 (ENG) EIO0000002798 (FRE) EIO0000002799 (GER) EIO0000002800 (SPA) EIO0000002801 (ITA) EIO0000002802 (CHS)
EcoStruxure Machine Expert - OpcUaHandling - Library Guide	EIO0000004021 (ENG) EIO0000004022 (FRE) EIO0000004023 (GER) EIO0000004025 (SPA) EIO0000004024 (ITA) EIO0000004026 (CHS)
EcoStruxure Machine Expert - SysLog - Library Guide	EIO0000004614 (ENG) EIO0000004615 (FRE) EIO0000004616 (GER) EIO0000004617 (ITA) EIO0000004618 (SPA) EIO0000004619 (CHS)
EcoStruxure Machine Expert - Modem Functions - Modem Library Guide	EIO0000000552 (ENG)
Cybersecurity Guidelines for EcoStruxure Machine Expert, Modicon and PacDrive Controllers and Associated Equipment, User Guide	EIO0000004242 (ENG)
Cybersecurity Admin Expert, User Manual	CAE_User_Guide (ENG)

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

About the Modicon M262 Logic/Motion Controller

Introduction

This chapter provides information about the Modicon M262 Logic/Motion Controller and devices that EcoStruxure Machine Expert can configure and program.

M262 Logic/Motion Controller Description

Overview

The M262 Logic/Motion Controller has various powerful features and can service a wide range of applications.

Programming Languages

The M262 Logic/Motion Controller is configured and programmed with the EcoStruxure Machine Expert software, which supports the following IEC 61131-3 programming languages:

- IL: Instruction List
- ST: Structured Text
- FBD: Function Block Diagram
- SFC: Sequential Function Chart
- LD: Ladder Diagram

EcoStruxure Machine Expert software can also be used to program these controllers using CFC (Continuous Function Chart) language.

Power Supply

The power supply of the M262 Logic/Motion Controller is 24 Vdc (see Modicon M262 Logic/Motion Controller, Hardware Guide).

Real Time Clock

The M262 Logic/Motion Controller includes a Real Time Clock (RTC) system (see Modicon M262 Logic/Motion Controller, Hardware Guide).

The system time is maintained by capacitors when the power is off. The time is maintained for 1 000 hours when the controller is not supplied.

Run/Stop

The M262 Logic/Motion Controller can be operated by the following:

- A hardware Run/Stop switch (see Modicon M262 Logic/Motion Controller, Hardware Guide).
- A Run/Stop operation by a dedicated digital input, defined in the software configuration. For more information, refer to [Configuration of Digital Inputs](#), page 79.

- An EcoStruxure Machine Expert software command.
- The system variable PLC_W in a Relocation Table, page 30.
- The Web Server, page 122.

Memory

This table describes the different types of memory:

Memory Type	Size	Use
RAM	256 Mbytes, of which 32 Mbytes are available for the application	For the execution of the application and the firmware.
Flash	1 Gbyte	Non-volatile memory dedicated to the retention of the program and data in case of a power interruption.
Non-volatile RAM	512 Kbytes	Non-volatile memory dedicated to the retention of the retain-persistent variables, and the diagnostic files and associated information.

Embedded Inputs/Outputs

The following embedded I/O types are available:

- Fast inputs
- Fast source outputs

Encoder for M262 Motion Controller

The following encoder modes are available:

- Incremental mode
- SSI mode

Removable Storage

The M262 Logic/Motion Controllers include an integrated SD card slot (see Modicon M262 Logic/Motion Controller, Hardware Guide).

The main uses of the SD card are:

- Initializing the controller with a new application
- Updating the controller and expansion module firmware, page 223
- Applying post configuration files to the controller, page 215
- Storing recipes files
- Receiving data logging files

Embedded Communication Features

The following types of communication ports are available:

- Serial Line (see Modicon M262 Logic/Motion Controller, Hardware Guide)
- USB Mini-B (see Modicon M262 Logic/Motion Controller, Hardware Guide)
- Ethernet (see Modicon M262 Logic/Motion Controller, Hardware Guide)

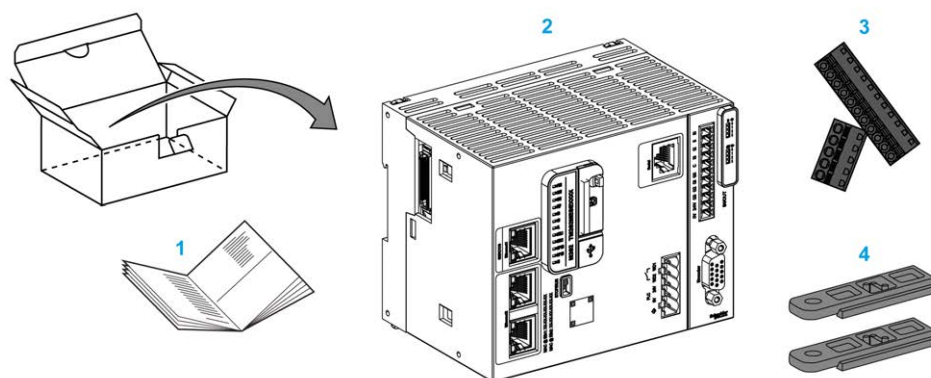
- Sercos (Ethernet 1) (see Modicon M262 Logic/Motion Controller, Hardware Guide)

Expansion Module and Bus Coupler Compatibility

The M262 Logic/Motion Controller supports expansion modules (see Modicon M262 Logic/Motion Controller, Hardware Guide). Refer also to the compatibility tables in the EcoStruxure Machine Expert - Compatibility and Migration User Guide (see EcoStruxure Machine Expert Compatibility and Migration, User Guide).

Delivery Content

The following figure presents the content of the delivery for the M262 Logic/Motion Controller:



- 1 M262 Logic/Motion Controller Instruction Sheet
- 2 M262 Logic/Motion Controller
- 3 Removable spring terminal blocks
- 4 Attachment parts

Modicon M262 Logic Controller

Modicon M262 Logic Controller

Controller Overview

TM262L• Controller Reference	Digital I/O	Power supply	Communication Ports	Terminal Type	Performance Duration for 1000 instructions	Motion Capabilities
TM262L01MESE8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 Ethernet port 1 dual port Ethernet switch	Removable spring	5 µs	Independent Axis on EIP / CANopen
TM262L10MESE8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 Ethernet port 1 dual port Ethernet switch	Removable spring	5 µs	Independent Axis on EIP / CANopen
TM262L20MESE8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 Ethernet port 1 dual port Ethernet switch	Removable spring	3 µs	Independent Axis on EIP / CANopen

Supported Features

- Cybersecurity access rights, encrypted communication see [Users Rights](#), page 69
- Web configuration Web server and WebVisualisation see [Web Server](#), page 122
- Protocol MQTT (signed / encrypted)
- OPC UA services, (signed / encrypted) see [OPC UA Server Overview](#), page 201
 - For TM262L01MESE8T and TM262L10MESE8T OPC UA Server, (signed / encrypted)
 - For TM262L20MESE8T OPC UA Client / Server, (signed / encrypted)
- 1 IO Scanner see [Modbus Serial IOScanner](#), page 190

- Supported Services:
 - HTTP (API)
 - DHCP (Client / Server) see DHCP Server, page 156
 - DNS Client
 - POP3 Client
 - RSTP (Eth2 port)
 - SMTP (Client / Agent)
 - SNMP see SNMP, page 121
 - FTP (Client / Server) see FTP Server, page 120
 - EtherNet IP (Adapter / Scanner) see Controller as a Target Device on EtherNet/IP, page 156
 - Modbus / TCP (Client / Server / NVL) see Modbus TCP Client/Server, page 119
 - Modbus / ASCII – RTU (Master / Slave / IO scanner / modem) see Modbus Manager, page 185
 - CANopen (Master)

Modicon M262 Motion Controller

Modicon M262 Motion Controller

Controller Overview

TM262M• Controller Reference	Digital I/O	Power supply	Communication Ports	Terminal Type	Encoder port	Performance Duration for 1000 instructions	Motion Capabilities
TM262M05MESS8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 dual port Ethernet switch 1 Ethernet port for fieldbus with Sercos interface	Removable spring	1 Encoder port	5 µs	Independent Axis on EIP / CANopen Synchronous axis on Sercos (4 axes maximum)
TM262M15MESS8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 dual port Ethernet switch 1 Ethernet port for fieldbus with Sercos interface	Removable spring	1 Encoder port	5 µs	Independent Axis on EIP / CANopen Synchronous axis on Sercos (4 axes maximum)
TM262M25MESS8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 dual port Ethernet switch 1 Ethernet port for fieldbus with Sercos interface	Removable spring	1 Encoder port	3 µs	Independent Axis on EIP / CANopen Synchronous axis on Sercos (8 axes maximum)
TM262M35MESS8T	4 fast inputs 4 fast source outputs	24 Vdc	1 serial line port 1 USB programming port 1 dual port Ethernet switch 1 Ethernet port for fieldbus with Sercos interface	Removable spring	1 Encoder port	3 µs	Independent Axis on EIP / CANopen Synchronous axis on Sercos (24 axes maximum)

Supported Features

- Cybersecurity access rights, encrypted communication see [Users Rights](#), page 69
- Web configuration Web server and WebVisualisation see [Web Server](#), page 122
- Protocol MQTT (signed / encrypted) for TM262M15MESS8T, TM262M25MESS8T and TM262M35MESS8T
- OPC UA services see [OPC UA Server Overview](#), page 201
 - For TM262M05MESS8T and TM262M15MESS8T OPC UA Server, (signed / encrypted)
 - For TM262M25MESS8T and TM262M35MESS8T OPC UA Client / Server, (signed / encrypted)

- 1 IO Scanner see Modbus Serial IO Scanner, page 190
- Supported Services:
 - HTTP (API)
 - DHCP (Client / Server) see DHCP Server, page 156
 - DNS Client
 - POP3 Client
 - RSTP (Eth2 port)
 - SMTP (Client / Agent)
 - SNMP see SNMP, page 121
 - FTP (Client / Server) see FTP Server, page 120
 - EtherNet IP (Adapter / Scanner) see Controller as a Target Device on EtherNet/IP, page 156
 - Modbus / TCP (Client / Server / NVL) see Modbus TCP Client/Server, page 119
 - Modbus / ASCII – RTU (Master / Slave / IO scanner / modem) see Modbus Manager, page 185
 - CANopen (Master)
 - Sercos (Master) see Sercos Configuration, page 180

Performance Overview

The Modicon TM262M• Motion Controller supports the features available in TM262L• Logic Controller, plus it integrates Motion functionalities.

The TM262M• range of Motion controllers is, without additional devices, ready for motion with the integrated Sercos motion bus. It merges the hard-real-time aspects of the Sercos interface with Ethernet. It is based upon and conforms to the Ethernet standard IEEE 802.3 and ISO/IEC 8802-3 to support the real-time application with high performance. Other features supporting motion functionalities include:

- Synchronous axis Sercos devices, managed by PLCopen libraries, are fully synchronous with the internal Motion task and the Sercos Cycle time, as for example: LXM32S.
- Non axis Sercos devices are also synchronized with the internal Motion task, for example, TM5NS01 islands or safety-related TM5CSLC100/ TM5CSLC200 controllers.
- External Encoder
 - External port for Incremental or SSI encoder. The encoder support is synchronized with the Motion application. It can be used like a real axis or a virtual axis.
- Fast input
 - The fast inputs support a touch probe function to capture position. The captured position can be used in the Motion application.
- Motion Kernel is embedded in the TM262M• Motion controller, allowing you to manage the motion functions:
 - Synchronous axis in coordinated movement in which the function blocks are based on the PLCopen Standard to control the position / speed of a single axis.
 - Gearing mode (Master / Slave Function Block).
 - Camming mode, based on recipes, with modifications on the fly. The recipe can be designed with the cam editor included in EcoStruxure Machine Expert.
 - G-code, based on recipes. The recipe can be designed with the CNC editor included in EcoStruxure Machine Expert.

Depending on the Motion controller and the Sercos cycle time, you can configure more or less synchronous axis and non axis Sercos devices.

A TM5 System island used on Sercos is managed as a non axis Sercos device. Although there are generally no restriction on the number of I/O in the Sercos configuration, the number of I/O configured increases the load of the Sercos bus and may lead to an overflow. If an overflow occurs, try increasing the Sercos cycle time. If increasing the Sercos cycle time is not compatible with your application, then optimize the application.

The following table indicates the performance capabilities of the Motion application:

Controller reference	Sercos cycle time	Synchronized axes on Sercos (activated and simulated)	Additional virtual axes FB_ControlledAxis	Additional Sercos devices
TM262M05MESS8T	1 ms	4	1	2
	2 ms	4	1	6
	4 ms	4	1	8
TM262M15MESS8T	1 ms	4	1	4
	2 ms	4	1	12
	4 ms	4	1	12
TM262M25MESS8T	1 ms	4	1	8
	2 ms	8	2	8
	4 ms	8	2	16
TM262M35MESS8T	1 ms	8	2	8
	2 ms	16	4	8
	4 ms	24	16	40

The Motion Sizer is embedded in EcoStruxure Machine Expert to help you define your kinematic architecture. For more information on these features, refer to OneMotionSizer Online Help (see Motion Sizer, Online Help).

How to Configure the Controller

Introduction

This chapter shows the default configuration of a project.

Configuring the Controller

Introduction

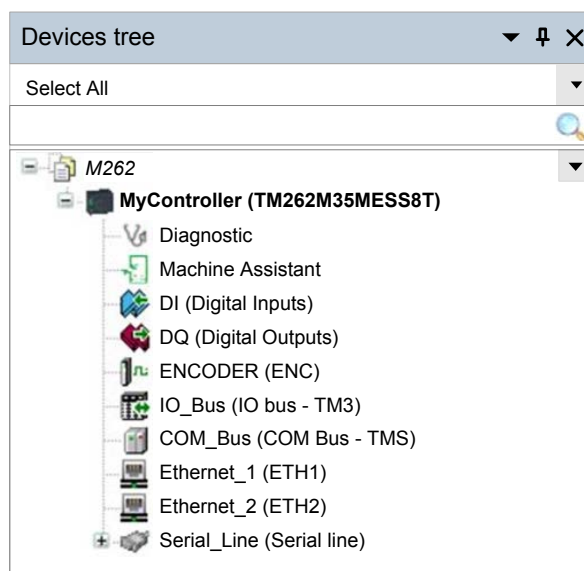
First, create a new project or open an existing project in the EcoStruxure Machine Expert software.

Refer to the EcoStruxure Machine Expert Programming Guide for information on how to:

- Add a controller to your project
- Add expansion modules to your controller
- Replace an existing controller
- Convert a controller to a different but compatible device

Devices Tree

The **Devices tree** presents a structured view of the hardware configuration. When you add a controller to your project, a number of nodes are added to the **Devices tree**, depending on the functions the controller provides.



Item	Use to Configure...
Diagnostic	Diagnostic messages and status.
Machine Assistant	Devices discovery and configuration.
DI	Embedded digital inputs of the controller.
DQ	Embedded digital outputs of the controller.
ENCODER	Incremental or SSI Encoder interface of the controller.
IO_Bus	Expansion modules connected to the controller.
COM_Bus	Communication modules connected to the controller.

Item	Use to Configure...
Ethernet_1	Embedded Ethernet dedicated to Motion Bus Sercos on TM262M•, dedicated to devices on TM262L•.
Ethernet_2	Embedded Ethernet communication.
Serial_Line	Serial line communication interface.

Applications Tree

The **Applications tree** allows you to manage project-specific applications as well as global applications, POUs, and tasks.

Tools Tree

The **Tools tree** allows you to configure the HMI part of your project and to manage libraries.

The **Tools tree** allows you to:

- Configure the HMI part of your project.
- Access to the **Library Manager** tool.
- Access to the **Message logger** tool, page 137.

Libraries

Introduction

This chapter describes the default libraries of the Modicon M262 Logic/Motion Controller.

Libraries

Introduction

Libraries provide functions, function blocks, data types and global variables that can be used to develop your project.

The **Library Manager** of EcoStruxure Machine Expert provides information about the libraries included in your project and allows you to install new ones. For more information on the **Library Manager**, refer to the Functions and Libraries User Guide. For more information about the libraries compatible with your controller, refer to the EcoStruxure Machine Expert Libraries Overview.

Modicon M262 Logic/Motion Controller

When you select a Modicon M262 Logic/Motion Controller for your application, EcoStruxure Machine Expert automatically loads the following libraries:

Library name	Description
Breakpoint Logging Functions ⁽¹⁾	Provides functions which can be used in Breakpoints for logging.
DeviceAbstractionLayer ⁽¹⁾	Interfaces and parameters of functionalities which are exposed by device objects.
DeviceIntegrationCommon ⁽¹⁾	Common functionalities shared by many devices (reserved for internal use only).
Diagnostic Device Support	Provides function blocks (reserved for internal use only) for devices to provide diagnostic information to the system diagnostic component on Schneider Electric controllers.
IoStandard	Library for the IO-configuration. This library provides the I/O interface for every IEC I/O driver.
M262 PLCSystem	M262 system functions and variables.
M262MotionExtension ⁽¹⁾	Exposes the on-board encoder input and touch-probe functionalities of an M262 to the controller application (reserved for internal use only).
MotionInterface ⁽¹⁾	Low level access for motion control.
PLCCommunication	Management of explicit data exchanges between controller and devices through Modbus or ASCII protocols.
PLCopen MC part 1 ⁽¹⁾	Motion control according to PLCopen Motion Control Part 1 v2.0 (formerly parts 1 and 2).
Relocation Table	Allows you to optimize exchanges between the Modbus client and the controller, by regrouping non-contiguous data into a contiguous table of registers. See <i>Relocation Table</i> , page 30.
SerialLineSystem	Provides serial line diagnostics data.
Standard	Contains IEC programming standard functions and function blocks.
TM3System	Contains functions and function blocks for TM3 I/O bus diagnostic information.
TMSSystem	Contains the function block and enumerated types for TMS I/O bus diagnostic information.
UserFunctionsBase ⁽¹⁾	Core implementation for exposing device functionalities to the controller application.
Util	Programming additional functions and function blocks: Analog Monitors, BCD Conversions, Bit/Byte Functions, Controller Datatypes, Function Manipulators, Mathematical Functions, Signals.
(1) Compatible with TM262M• references only.	

Supported Standard Data Types

Introduction

This chapter provides the different IEC data types supported by the Controller.

Supported Standard Data Types

Supported Standard Data Types

The controller supports the following IEC data types:

Data Type	Lower Limit	Upper Limit	Information Content
BOOL	FALSE	TRUE	1 Bit
BYTE	0	255	8 Bit
WORD	0	65,535	16 Bit
DWORD	0	4,294,967,295	32 Bit
LWORD	0	$2^{64}-1$	64 Bit
SINT	-128	127	8 Bit
USINT	0	255	8 Bit
INT	-32,768	32,767	16 Bit
UINT	0	65,535	16 Bit
DINT	-2,147,483,648	2,147,483,647	32 Bit
UDINT	0	4,294,967,295	32 Bit
LINT	-2^{63}	$2^{63}-1$	64 Bit
ULINT	0	$2^{64}-1$	64 Bit
REAL	$1.175494351\text{e-}38$	$3.402823466\text{e}+38$	32 Bit
LREAL	$2.2250738585072014\text{e-}308$	$1.7976931348623158\text{e}+308$	64 Bit
STRING	1 character	–	1 character = 1 byte
WSTRING	1 character	–	1 character = 1 word
TIME	0	4294967295	32 Bit

For more information on ARRAY, LTIME, DATE, TIME, DATE_AND_TIME, and TIME_OF_DAY, refer to the EcoStruxure Machine Expert Programming Guide.

Memory Mapping

Introduction

This chapter describes the memory maps and sizes of the different memory areas in the Modicon M262 Logic/Motion Controller. These memory areas are used to store user program logic, data and the programming libraries.

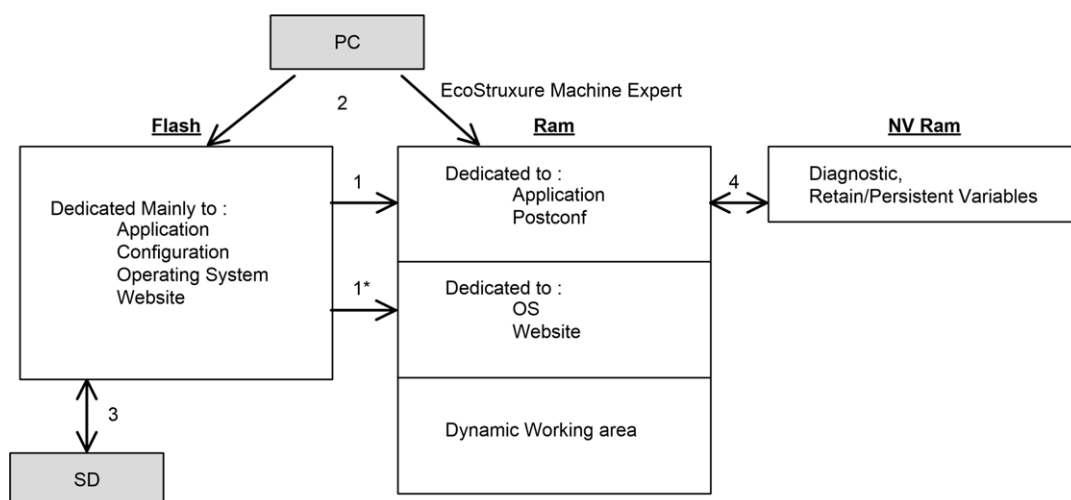
Controller Memory Organization

Introduction

The controller memory is composed of three types of physical memory:

- The Non-Volatile Memory, page 26 (NVM) contains files (application, configuration files, and so on).
- The Random Access Memory (RAM) is used for application execution.
- The Non-Volatile Random Access Memory (NVRAM) is used to save the retain-persistent variables and diagnostic information.

Files Transfers in Memory



Item	Controller State	File Transfer Events	Connection	Description
1	–	Initiated automatically at Power ON and Reboot	Internal	Files transfer from non-volatile memory to RAM. The content of the RAM is overwritten.
1*	–	Initiated automatically at Power ON and Reboot	Internal	Operating system files transfer.
2	All states except INVALID_OS ⁽¹⁾	Initiated by user	Ethernet or USB programming port	Files can be transferred via: • Web Server, page 122 • FTP server, page 120 • Controller Assistant • EcoStruxure Machine Expert (see EcoStruxure Machine Expert, Programming Guide)
3	All states	Initiated automatically by script (data transfer) or by power cycle (cloning) when an SD card is inserted	SD card	Up/download with SD card ⁽¹⁾ .

Item	Controller State	File Transfer Events	Connection	Description
4	All states	Initiated by system	Internal	Save of modified retain-persistent variables and the context on power OFF.
(1) If the controller is in the INVALID_OS state, the only accessible memory is the SD card and only for firmware upgrades.				

NOTE: The modification of files in non-volatile memory does not affect a running application. Any changes to files in non-volatile memory are taken into account at the next reboot, except for the user files directly used by the application.

Non-Volatile Memory Organization

Introduction

The non-volatile memory contains the file system used by the controller.

File Type

The Modicon M262 Logic/Motion Controller manages the following file types:

System function (/sys)	Description
Operating System (OS)	Controller firmware that can be written to non-volatile memory. The firmware file is applied at next reboot of the controller.

User functions (/usr)	Description
Boot application	This file resides in non-volatile memory and contains the compiled binary code of the executable application. Each time the controller is rebooted, the executable application is extracted from the boot application and copied into the controller RAM ⁽¹⁾ .
Application source	Source file that can be uploaded from non-volatile memory to the PC if the source file is not available on the PC ⁽²⁾ .
Post configuration	File that contains Ethernet and serial line parameters. The parameters specified in the file override the parameters in the executable application at each reset.
Firewall parameters	Settings used to configure the firewall of the M262 Logic/Motion Controller. These settings restrict access to authorized personnel and protocols only. See <i>Firewall Configuration</i> , page 142 for more information.
Data logging	Files in which the controller logs events as specified by the application.
(1) The creation of a boot application is optional in EcoStruxure Machine Expert, according to application properties. Default option is to create the boot application on download. When you download an application from EcoStruxure Machine Expert to the controller, you are transferring only the binary executable application directly to RAM. (2) EcoStruxure Machine Expert does not support uploading of either the executable application or the boot application to a PC for modification. Program modifications must be made to the application source. When you download your application, you have the option to store the source file to the non-volatile memory.	

File Organization

This table shows the file organization of the non-volatile memory:

Disk	Directory	File	Content	Up/Downloaded data type
/sys	Pkg	Temporary file	Internal use	N/A
/usr	App	Application.app	Boot application	Application
		Application.crc		–
		Archive.prj ⁽¹⁾	Application source	–
	Cfg	Machine.cfg ⁽¹⁾	Post configuration file, page 215	Configuration
		CodesysLateConf.cfg	Name of application to launch.	Configuration
		FirewallDefault.cmd	Default firewall settings. By default, this file does not exist. It can be added optionally.	Configuration
		ntp.conf	Contains the network time protocol (NTP) configuration.	Configuration
		ntp.drift.	Contains the calculated drift of the system clock compared to UTC time.	Configuration
	Log	UserDefinedLogName_1.log	All *.log files created using the data logging functions (see EcoStruxure Machine Expert, Data Logging Functions, DataLogging Library Guide). You must specify the total number of files created and the names and contents of each log file using the datalogging feature.	log file
		UserDefinedLogName_n.log		–
	pki	–	Certificate store for M262 secured protocols.	–
	Rcp	–	Main directory for recipes.	–
	Syslog	crash.txt ⁽¹⁾ LoggerFile_xxx.mel	Record of detected system errors. For use by Schneider Electric Technical Support.	Log file
	Visu	–	Used for the WebVisualisation feature.	–
	_cnc	UserDefinedName.cnc	Pre-programmed control commands	G-code data
	Alarms	Application.alarmstorage.X.sqlite Application.alarmstorage.X.sqlite.metadata	Database of configured alarms	Alarm manager data
	Trend	Application.TrendRecording.X.sqlite Application.TrendRecording.X.sqlite.metadata	Database of configured trends	Trend recorder data
/sd0	–	–	SD card. Refer to Managing Script Files, page 231.	–
	–	User files	–	–

⁽¹⁾ If the files had been created due to specific events or customer requirements.

NOTE: For more information on libraries and available function blocks, refer to [Libraries](#), page 23.

Files Redirection

When system, program or certain user activity creates specific file types, the M262 Logic/Motion Controller examines the file extension and automatically moves the file to a corresponding folder in non-volatile memory.

The following table lists the file types that are moved in this way and the destination folder in non-volatile memory:

File extensions	Non-volatile memory folder
*.app, *.ap_, *.err, *.crc, *.frc, *.prj	/usr/App
*.cfg, *.cf_	/usr/Cfg
*.log	/usr/Log
*.rcp, *.rsi	/usr/Rcp

Backup Data Logging File

Data logging files can become large to the point of exceeding the space available in the file system. Therefore, you should develop a method to archive the log data periodically on an SD card. You could split the log data into several files, for example `LogMonth1`, `LogMonth2`, and use the **ExecuteScript** command to copy the first file to an SD card. Afterwards, you may remove it from the internal file system while the second file is accumulating data. If you allow the data logging file to grow and exceed the limits of the file size, you could lose data.

NOTICE

LOSS OF APPLICATION DATA

- Backup SD card data regularly.
- Do not remove power or reset the controller, and do not insert or remove the SD card while it is being accessed.

Failure to follow these instructions can result in equipment damage.

RAM Memory Organization

Introduction

This section describes the Random Access Memory (RAM) size for different areas of the Modicon M262 Logic/Motion Controller.

Memory Mapping

The RAM is composed of two areas:

- Dedicated application memory
- OS memory

This table describes the dedicated application memory:

Area	Element
System area	System Area Mappable Addresses %MW0...%MW59999
	System and diagnostic variables (%MW60000...%MW60199) This memory is accessible through Modbus requests only. These must be read-only requests.
	Dynamic Memory Area: Read Relocation Table, page 30 (%MW60200...%MW61999) This memory is accessible through Modbus requests only. These must be read-only requests.
	System and diagnostic variables (%MW62000...%MW62199) This memory is accessible through Modbus requests only. These can be read or write requests.
	Dynamic Memory Area: Write Relocation Table, page 30 (%MW62200...%MW63999) This memory is accessible through Modbus requests only. These can be read or write requests.
User area	Symbols
	Variables
	Libraries
	Application

System and Diagnostic Variables

Variables	Description
PLC_R	Structure of controller read-only system variables.
PLC_W	Structure of controller read/write system variables.
ETH_R	Structure of Ethernet read-only system variables (Ethernet counters).
ETH_W	Structure of Ethernet read/write system variables. Allows you to reset Ethernet counters.
SERIAL_R	Structure of Serial Line read-only system variables (Serial Line counters).
SERIAL_W	Structure of Serial Line read-write system variables. Allows you to reset Serial Line counters.
TM3_MODULE_R	Structure of TM3 modules read-only system variables.
TM3_BUS_W	Structure of TM3 bus read-write system variables.
TMS_BUS_DIAG_R	Structure of TMS bus read-only system variables (diagnostic).
TMS_MODULE_DIAG_R	Structure of TMS modules read-only system variables (diagnostic).

For more information on system and diagnostic variables, refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

NVRAM Memory Organization

Introduction

The NVRAM memory contains:

- Files saved for the diagnostics
- Remanent (retain-persistent) variables

NVRAM size

The following table describes the size of the NVRAM:

User Function	Description	Size
System diagnostics	Contain the controller context saved on power off.	128 Kbytes
Remanent (retain-persistent) variables	Modified and saved in the NVRAM. This action impacts the cycle time. Retain: saved after each cycle. Persistent: saved after each modification.	Retain: 64 Kbytes Persistent: 64 Kbytes

Remanent or retain-persistent variables are saved in the NVRAM. Each subsequent read/write access to these variables requires a NVRAM access. For more information about remanent variables, see [Remanent Variables](#), page 58. For more information about the impact on performance, see [Processing Performance](#), page 272.

NOTE: For an optimum cycle time, only access the retain-persistent variables when necessary. For frequent (Read) access, copy theses variables to a working memory in the RAM.

Relocation Table

Introduction

The **Relocation Table** allows you to organize data to optimize communication between the controller and other equipment by regrouping non-contiguous data into a contiguous table of located registers, accessible through Modbus.

NOTE: A relocation table is considered an object. Only one relocation table object can be added to a controller.

Relocation Table Description

This table describes the **Relocation Table** organization:

Register	Description
60200...61999	Dynamic Memory Area: Read Relocation Table The %MW registers are read from the variables at each cycle.
62200...63999	Dynamic Memory Area: Write Relocation Table The %MW registers are copied to the variables at each cycle.

For further information, refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

Adding a Relocation Table

This table describes how to add a **Relocation Table** to your project:

Step	Action
1	In the Applications tree tab, select the Application node.
2	Click the right mouse button.
3	Click Objects > Relocation Table.... Result: The Add Relocation Table window is displayed.
4	Click Add . Result: The new relocation table is created and initialized. NOTE: As a relocation table is unique for a controller, its name is Relocation Table and cannot be changed.

Relocation Table Editor

The relocation table editor allows you to organize your variables in the relocation table.

To access the relocation table editor, double-click the **Relocation Table** node in the **Tools tree** tab:



This picture describes the relocation table editor:

Relocation Table [MyController_1:PLC Logic: Application]

Read:









ID	Variable	Address	Length	Validity
1	PLC_GVL.PLC_R.i_dwSerialNumber	%MW60200	2	True
2	PLC_GVL.PLC_R.i_sNodeName	%MW60202	16	True
3	PLC_GVL.PLC_R.i_sProductRef	%MW60218	16	True
4	GVL.DIG_IO_LOOPS_STS	%MW60234	1	True

Write:









ID	Variable	Address	Length	Validity
1	PLC_GVL.PLC_W.q_wResetCounterEvent	%MW62200	1	True
2	PLC_GVL.ETH_W.q_wResetCounter	%MW62201	1	True
3	GVL.AckDigLoopFit	%MW62202	1	True
4	GVL.TempLoop1SetPoint	%MW62203	2	True

Icon	Element	Description
	New Item	Adds an element to the list of mapped variables.
	Move Down	Moves down the selected element of the list.
	Move Up	Moves up the selected element of the list.
	Delete Item	Removes the selected elements of the list.
	Copy	Copies the selected elements of the list.
	Paste	Pastes the elements copied.
	Erase Empty Item	Removes all the elements of the list for which the "Variable" column is empty.
-	ID	Automatic incremental integer (not editable).
-	Variable	The name or the full path of a variable (editable).
-	Address	The address of the system area where the variable is stored (not editable).
-	Length	Variable length in word.
-	Validity	Indicates if the entered variable is valid (not editable).

NOTE: If a variable is undefined after program modifications, the content of the cell is displayed in red, the related **Validity** cell is False, and **Address** is set to -1.

Tasks

Introduction

The **Task Configuration** node in the **Applications tree** allows you to define one or more tasks to control the execution of your application program.

The task types available are:

- Cyclic
- Freewheeling
- Event
- External event

This chapter begins with an explanation of these task types and provides information regarding the maximum number of tasks, the default task configuration, and task prioritization. In addition, this chapter introduces the system and task watchdog functions and explains its relationship to task execution.

Maximum Number of Tasks

Maximum Number of Tasks

The maximum number of tasks you can define for the Modicon M262 Logic/Motion Controller are:

- Total number of tasks = 16
- Cyclic tasks = 8
- Freewheeling tasks = 1
- Cyclic tasks + freewheeling tasks = 8
- Event tasks = 8
- External Event tasks = 8

Special Considerations for Freewheeling

A Freewheeling task, page 34 does not have a fixed duration. In Freewheeling mode, the task scan starts when the previous scan has been completed and after a period of system processing (30% of the total duration of the Freewheeling task). If the system processing period is reduced to less than 15% for more than 3 seconds due to interruptions by other tasks, a system error is detected. For more information, refer to *System and Task Watchdogs*, page 37.

NOTE: You may wish to avoid using a Freewheeling task in a multi-task application when some high priority and time-consuming tasks are running. Doing so may provoke a task Watchdog Timeout. You should not assign CANopen to a freewheeling task. CANopen should be assigned to a cyclic task.

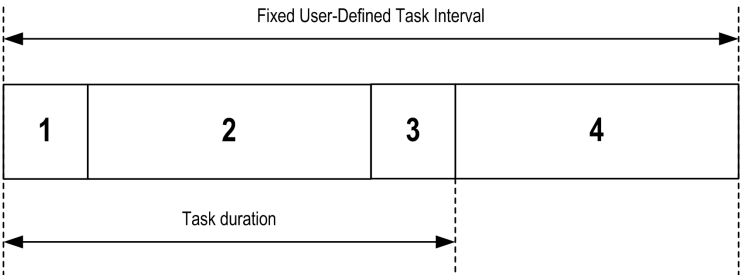
Task Types

Introduction

The following section describes the various task types available for your program, along with a description of the task type characteristics.

Cyclic Task

A Cyclic task is assigned a fixed cycle time using the interval setting in the type section of the configuration subtab for that task. Each Cyclic task type executes as follows:

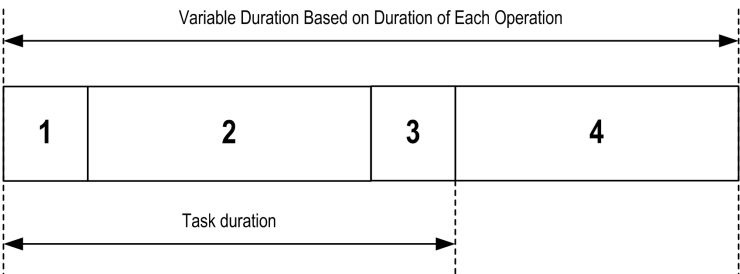


1.	Read Inputs: The physical input states are written to the %I input memory variables and other system operations are executed.
2.	Task Processing: The user code (POU, and so on) defined in the task is processed. The %Q output memory variables are updated according to your application program instructions but not yet written to the physical outputs during this operation.
3.	Write Outputs: The %Q output memory variables are modified with the output forcing that has been defined; however, the writing of the physical outputs depends upon the type of output and instructions used. For more information on defining the bus cycle task, refer to the EcoStruxure Machine Expert Programming Guide and PLC Settings, page 62. For more information on I/O behavior, refer to Controller States Detailed Description, page 45.
4.	Remaining Interval time: The controller firmware carries out system processing and other lower priority tasks.

NOTE: If you define too short a period for a cyclic task, it will repeat immediately after the write of the outputs and without executing other lower priority tasks or any system processing. This will affect the execution of all tasks and cause the controller to exceed the system watchdog limits, generating a system watchdog exception.

Freewheeling Task

A Freewheeling task does not have a fixed duration. In Freewheeling mode, each task scan begins when the previous scan has been completed and after a short period of system processing. Each Freewheeling task type executes as follows:



1.	Read Inputs: The physical input states are written to the %I input memory variables and other system operations are executed.
2.	Task Processing: The user code (POU, and so on) defined in the task is processed. The %Q output memory variables are updated according to your application program instructions but not yet written to the physical outputs during this operation.

3.	Write Outputs: The %Q output memory variables are modified with the output forcing that has been defined; however, the writing of the physical outputs depends upon the type of output and instructions used. For more information on defining the bus cycle task, refer to the EcoStruxure Machine Expert Programming Guide and PLC Settings, page 62. For more information on I/O behavior, refer to Controller States Detailed Description, page 45.
4.	System Processing: The controller firmware carries out system processing and the other lower priority tasks (for example: HTTP management, Ethernet management, parameters management).

NOTE: If you want to define the task interval, refer to [Cyclic Task](#), page 34.

Event Task

This type of task is event-driven and is initiated by a program variable. It starts at the rising edge of the boolean variable associated to the trigger event unless pre-empted by a higher priority task. In that case, the Event task will start as dictated by the task priority assignments.

For example, if you have defined a variable called `my_Var` and would like to assign it to an Event, proceed as follows:

Step	Action
1	Double-click the TASK in the Applications tree .
2	Select Event from the Type list in the Configuration tab.
3	Click the Input Assistant button  to the right of the Event field. Result: The Input Assistant window appears.
4	Navigate in the tree of the Input Assistant dialog box to find and assign the <code>my_Var</code> variable.

NOTE: When the event task is triggered with too high a frequency, the controller may detect an error and transition to the HALT state (Exception).

The maximum rate of events is:

- 12 events per millisecond for TM262L01MESE8T, TM262L10MESE8T, TM262M05MESS8T and TM262M15MESS8T
- 16 events per millisecond for TM262L20MESE8T, TM262M25MESS8T and TM262M35MESS8T

If the event task is triggered at a higher frequency than this, the message 'ISR Count Exceeded' is logged in the application log page.

External Event Task

This type of task is event-driven and is initiated by the detection of a hardware or hardware-related function event. It starts when the event occurs unless pre-empted by a higher priority task. In that case, the External Event task will start as dictated by the task priority assignments.

For example, an External event task could be associated with an HSC Stop event. To associate the **HSC0_STOP** event to an External event task, select it from the **External event** drop-down list on the **Configuration** tab.

The external event task can be associated with the CAN Sync event. To associate the **CAN_1_SYNC** event to an external event task, select it from the **External event** dropdown list in the **Configuration** tab.

Different types of events can be associated with an External event task:

- HSC thresholds (see Modicon TM3 Expert I/O Modules, HSC Library Guide)
- HSC Stop
- CAN Sync
- AFTER_RTP
- HSC Event Periodometer
- Event input

NOTE: CAN Sync is a specific event object, depending on the **CANopen manager** configuration.

NOTE: When the external event task is triggered with too high a frequency, the controller may detect an error and transition to the HALT state (Exception).

The maximum rate of events is:

- 12 events per millisecond for TM262L01MESE8T, TM262L10MESE8T, TM262M05MESS8T and TM262M15MESS8T
- 16 events per millisecond for TM262L20MESE8T, TM262M25MESS8T and TM262M35MESS8T

If the event task is triggered at a higher frequency than this, the message 'ISR Count Exceeded' is logged in the application log page.

Task Configuration Screen

Screen Description

This screen allows you to configure the tasks. Double-click the task that you want to configure in the **Applications tree** to access this screen.

Each configuration task has its own parameters that are independent of the other tasks.

The **Configuration** window is composed of four parts:

The screenshot shows the 'Configuration' window for a task named 'MAST'. The window is divided into four main sections:

- Priority (0..31):** A text field containing the value '1'.
- Type:** A dropdown menu set to 'Cyclic' and an 'Interval (e.g. t#200ms):' field set to 't#20ms'.
- Watchdog:** A section with a checked 'Enable' checkbox, a 'Time (e.g. t#200ms):' field set to '100' with a unit dropdown set to 'ms', and a 'Sensitivity:' field set to '1'.
- Task List:** A table with two columns: 'POU' and 'Comment'. Above the table is a toolbar with icons for 'Add Call', 'Remove Call', 'Change Call', 'Move Up', 'Move Down', and 'Open POU'.

The table describes the fields of the **Configuration** screen:

Field Name	Definition
Priority	Configure the priority of each task with a number from 0 to 31 (0 is the highest priority, 31 is the lowest). Only one task at a time can be running. The priority determines when the task runs: a higher priority task pre-empts a lower priority task. NOTE: Do not assign tasks with the same priority. If there are yet other tasks that attempt to pre-empt tasks with the same priority, the result could be indeterminate and unpredictable. For important information, refer to <i>Task Priorities</i>, page 38.
Type	These task types are available: • Cyclic, page 34 • Event, page 35 • External , page 35 • Freewheeling, page 34
Watchdog	To configure the watchdog, page 37, define these two parameters: • Time: enter the timeout before watchdog execution. • Sensitivity: defines the number of expirations of the watchdog timer before the controller stops program execution and enters a HALT state.
POUs	The list of Programming Organization Units (POUs) controlled by the task is defined in the EcoStruxure Machine Expert, Programming Guide: • To add a POU linked to the task, use the command Add Call and select the POU in the Input Assistant editor. • To remove a POU from the list, use the command Remove Call. • To replace the selected POU of the list by another one, use the command Change Call. • POUs are executed in the order shown in the list. To move the POUs in the list, select a POU and use the command Move Up or Move Down. NOTE: You can create as many POUs as you want. An application with several small POUs, as opposed to one large POU, can improve the refresh time of the variables in online mode.

System and Task Watchdogs

Introduction

Two types of watchdog functionality are implemented for the Modicon M262 Logic/Motion Controller:

- **System Watchdogs:** These watchdogs are managed by the controller firmware. You cannot configure them.
- **Task Watchdogs:** These watchdogs are optional watchdogs that you can define for each task. These are configurable in EcoStruxure Machine Expert.

System Watchdogs

Three system watchdogs are defined for the Modicon M262 Logic/Motion Controller. They are managed by the controller firmware and are therefore sometimes referred to as hardware watchdogs in the EcoStruxure Machine Expert online help. When one of the system watchdogs exceeds its threshold conditions, an error is detected.

The threshold conditions for the three system watchdogs are defined as follows:

- If all of the tasks require more than 85% of the processor resources for more than 3 seconds, a system error is detected. The controller enters the HALT state.

- If the total execution time of the tasks with priorities between 0 and 24 reaches 100% of processor resources for more than 1 second, an application error is detected. The controller responds with an automatic reboot into the EMPTY state.
- If the lowest priority task of the system is not executed during an interval of 10 seconds, a system error is detected. The controller responds with an automatic reboot into the EMPTY state.

NOTE: You cannot configure the system watchdogs.

Task Watchdogs

EcoStruxure Machine Expert allows you to configure an optional task watchdog for every task defined in your application program. (Task watchdogs are sometimes also referred to as software watchdogs or control timers in the EcoStruxure Machine Expert online help). When one of your defined task watchdogs reaches its threshold condition, an application error is detected and the controller enters the HALT state.

When defining a task watchdog, the following options are available:

- **Time:** This defines the maximum execution time for a task. When a task takes longer than this, the controller reports a task watchdog exception.
- **Sensitivity:** The sensitivity field defines the number of task watchdog exceptions that must occur before the controller detects an application error.

To access the configuration of a task watchdog, double-click the **Task** in the **Applications tree**.

NOTE: For more information on watchdogs, refer to EcoStruxure Machine Expert Programming Guide.

Task Priorities

Task Priority Configuration

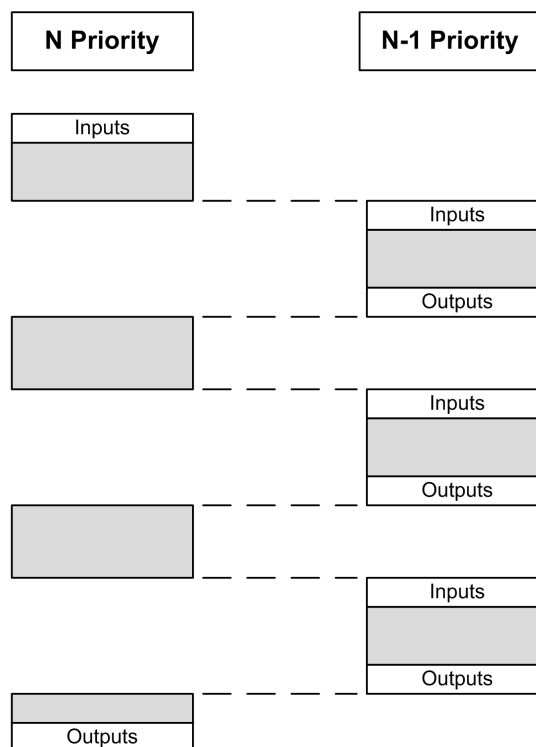
You can configure the priority of each task between 0 and 31 (0 is the highest priority, 31 is the lowest). Each task must have a unique priority. Assigning the same priority to more than one task leads to a build error.

Task Priority Suggestions

- Priority 0 to 24: Controller tasks. Assign these priorities to tasks with a high availability requirement.
- Priority 25 to 31: Background tasks. Assign these priorities to tasks with a low availability requirement.

Task Priorities of Embedded I/Os

When a task cycle starts, it can interrupt any task with lower priority (task preemption). The interrupted task resumes when the higher priority task cycle is finished.



NOTE: If the same input is used in different tasks the input image may change during the task cycle of the lower priority task.

To improve the likelihood of proper output behavior during multitasking, a build error message is displayed if outputs in the same byte are used in different tasks.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Map your inputs so that tasks do not alter the input images in an unexpected manner.

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

Task Priorities of TM3 Modules and CANopen I/Os

You can select the task that drives TM3 I/Os and CANopen physical exchanges. In the **PLC settings**, select **Bus cycle task** to define the task for the exchange. By default, the task is set to **MAST**. This definition at the controller level can be overridden by the I/O bus configuration, page 107. During the read and write phases, all physical I/Os are refreshed at the same time. TM3 and CANopen data is copied into a virtual I/O image during a physical exchanges phase, as shown in this figure:



Inputs are read from the I/O image table at the beginning of the task cycle.
Outputs are written to the I/O image table at the end of the task.

NOTE: TM3 influence the application execution time. You can configure the **Bus cycle options** using **I/O mapping** tab. Refer to the Modicon TM3 Expansion Modules – Programming Guide.

Default Task Configuration

Default Task Configuration

The MAST task can be configured in Freewheeling or Cyclic mode. The MAST task is automatically created by default in Cyclic mode. Its preset priority is medium (15), its preset interval is 10 ms, and its task watchdog service is activated with a time of 50 ms and a sensitivity of 1. Refer to [Task Priorities](#), page 38 for more information on priority settings. Refer to [Task Watchdogs](#) for more information on watchdogs.

Designing an efficient application program is important in systems approaching the maximum number of tasks. In such an application, it can be difficult to keep the resource utilization below the system watchdog threshold. If priority reassignments alone are not sufficient to remain below the threshold, some lower priority tasks can be made to use fewer system resources if the SysTaskWaitSleep function, contained in the SysTask library, is added to those tasks.

NOTE: Do not delete or change the name of the MAST task. Otherwise, EcoStruxure Machine Expert detects an error when you attempt to build the application, and you are not able to download it to the controller.

Controller States and Behaviors

Introduction

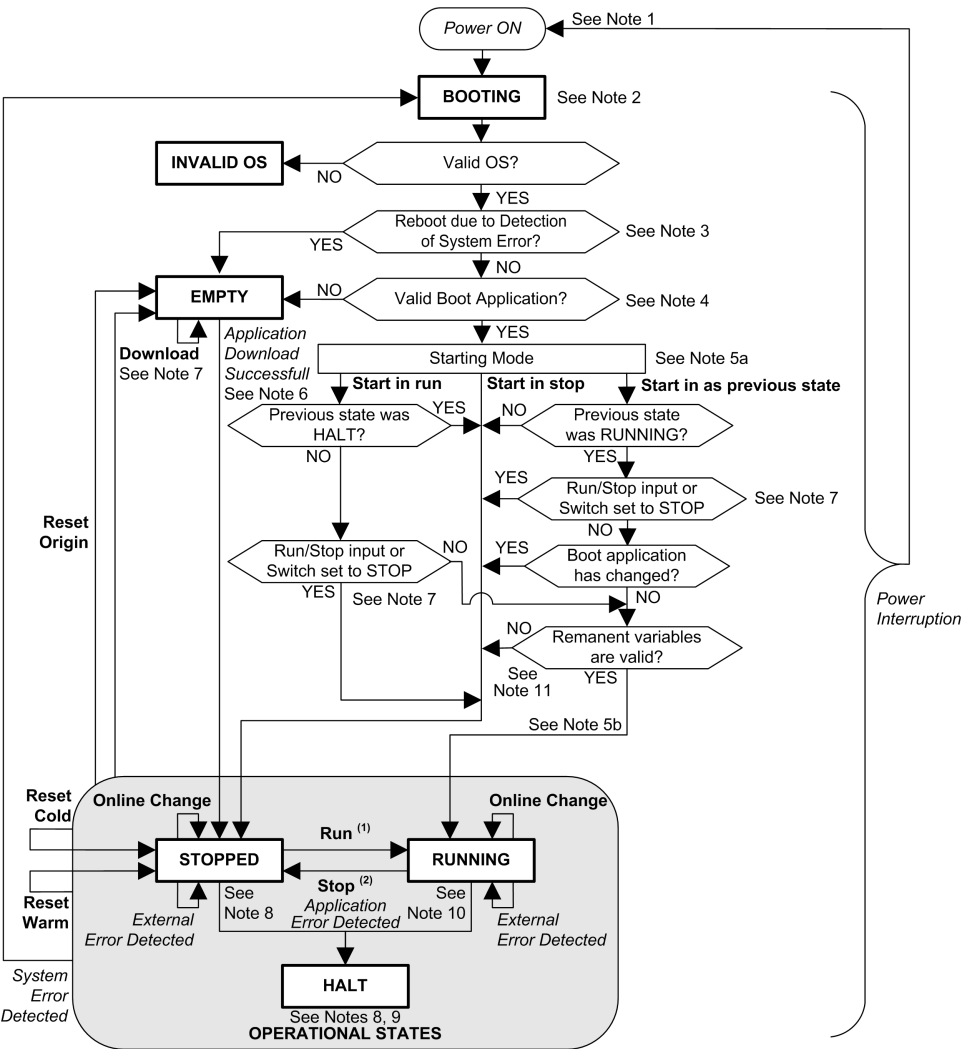
This chapter provides information on controller states, state transitions, and behaviors in response to system events. It begins with a detailed controller state diagram and a description of each state. It then defines the relationship of output states to controller states before explaining the commands and events that result in state transitions. It concludes with information about Remanent variables and the effect of EcoStruxure Machine Expert task programming options on the behavior of your system.

Controller State Diagram

Controller State Diagram

Controller State Diagram

This diagram describes the controller operating mode:



ALL-CAPS BOLD: Controller states

Bold: User and application commands

Italics: System events

Normal text: Decisions, decision results, and general information

(1) For details on STOPPED to RUNNING state transition, refer to Run Command, page 50.

(2) For details on RUNNING to STOPPED state transition, refer to Stop Command, page 51.

Note 1

The alarm relay is open.

Note 2

The outputs assume their hardware initialization states. The encoder power supply is not enabled. The voltage is 0. The alarm relay is closed.

Note 3

In some cases, when a system error is detected, it causes the controller to reboot automatically into the EMPTY state as if no Boot application were present in the non-volatile memory. However, the Boot application is not deleted from the non-volatile memory. In this case, the **ERR LED** (red) flashes fast and regularly.

Note 4

After verification of a valid Boot application the following events occur:

- The application is loaded into RAM.
- The [Post Configuration, page 215](#) file settings (if any) are applied.

During the load of the boot application, a Check context test occurs to verify that the Remanent variables are valid. If the Check context test is invalid, the boot application loads but the controller transitions to the STOPPED state, [page 55](#).

Note 5a

The **Starting Mode** is set in the **PLC settings** tab of the **Controller Device Editor**, [page 62](#).

Note 5b

When a power interruption occurs, the controller continues in the RUNNING state for at least 4 ms before shutting down. If you have configured and provide power to the Run/Stop input from the same source as the controller, the loss of power to this input is detected immediately, and the controller behaves as if a STOP command was received. Therefore, if you provide power to the controller and the Run/Stop input from the same source, your controller reboots normally into the STOPPED state after a power interruption when **Starting Mode** is set to **Start as previous state**.

Note 6

During a successful application download the following events occur:

- The application is loaded directly into RAM.
- By default, the Boot application is created and saved into the non-volatile memory.

Note 7

The default behavior after downloading an application program is for the controller to enter the STOPPED state irrespective of the Run/Stop input setting, the Run/Stop switch position or the last controller state before the download.

However, there are two considerations in this regard:

Online Change:	An online change (partial download) initiated while the controller is in the RUNNING state returns the controller to the RUNNING state if successful and provided the Run/Stop input is configured and set to Run or Run/Stop switch is set to Run. Before using the Login with online change option, test the changes to your application program in a virtual or non-production environment and confirm that the controller and attached equipment assume their expected conditions in the RUNNING state. ⚠ WARNING UNINTENDED EQUIPMENT OPERATION Always verify that online changes to a RUNNING application program operate as expected before downloading them to controllers. Failure to follow these instructions can result in death, serious injury, or equipment damage. NOTE: Online changes to your program are not automatically written to the Boot application, and are overwritten by the existing Boot application at the next reboot. If you wish your changes to persist through a reboot, manually update the Boot application by selecting Create boot application in the online menu (the controller must be in the STOPPED state to achieve this operation).
Multiple Download:	EcoStruxure Machine Expert has a feature that allows you to perform a full application download to multiple targets on your network or fieldbus. One of the default options when you select the Multiple Download... command is the Start all applications after download or online change option, which restarts all download targets in the RUNNING state, provided their respective Run/Stop inputs are commanding the RUNNING state, but irrespective of their last controller state before the multiple download was initiated. Deselect this option if you do not want all targeted controllers to restart in the RUNNING state. In addition, before using the Multiple Download option, test the changes to your application program in a virtual or non-production environment and confirm that the targeted controllers and attached equipment assume their expected conditions in the RUNNING state. ⚠ WARNING UNINTENDED EQUIPMENT OPERATION Always verify that your application program will operate as expected for all targeted controllers and equipment before issuing the “Multiple Download...” command with the “Start all applications after download or online change” option selected. Failure to follow these instructions can result in death, serious injury, or equipment damage. NOTE: During a multiple download, unlike a normal download, EcoStruxure Machine Expert does not offer the option to create a Boot application. You can manually create a Boot application at any time by selecting Create boot application in the Online menu on all targeted controllers.

Note 8

The EcoStruxure Machine Expert software platform allows many powerful options for managing task execution and output conditions while the controller is in the STOPPED or HALT states. Refer to [Controller States Description](#), page 45 for further details.

Note 9

To exit the HALT state it is necessary to issue one of the Reset commands (Reset Warm, Reset Cold, Reset Origin), download an application or cycle power.

In case of non-recoverable event (hardware watchdog or internal error), a power cycle is mandatory.

Note 10

The RUNNING state has two exception conditions:

- **RUNNING with External Error:** this exception condition is indicated by the I/O LED, which displays solid red. You may exit this state by clearing the external error (probably changing the application configuration). No controller commands are required, but may however include the need of a power cycle of the controller. For more information, refer to *I/O Configuration General Description*, page 102.
- **RUNNING with Breakpoint:** this exception condition is indicated by the RUN LED, which displays a single green flash. Refer to *Controller States Description*, page 45 for further details.

Note 11

The boot application can be different from the application loaded. It can happen when the boot application was downloaded through SD card, FTP, or file transfer or when an online change was performed without creating the boot application.

Controller States Description

Controller States Description

Introduction

This section provides a detailed description of the controller states.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Never assume that your controller is in a certain controller state before commanding a change of state, configuring your controller options, uploading a program, or modifying the physical configuration of the controller and its connected equipment.
- Consider the effect of any of these operations on all connected equipment before performing any of these operations.
- Positively confirm the controller state by viewing its LEDs before acting on the controller.
- Confirm the condition of the Run/Stop input (if so equipped and configured) and/or the Run/Stop switch (if so equipped) before acting on the controller.
- Verify the presence of output forcing before acting on the controller.
- Review the controller status information via EcoStruxure Machine Expert before acting on the controller.⁽¹⁾

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

⁽¹⁾ The controller states can be read in the PLC_R.i_wStatus system variable of the M262 System library.

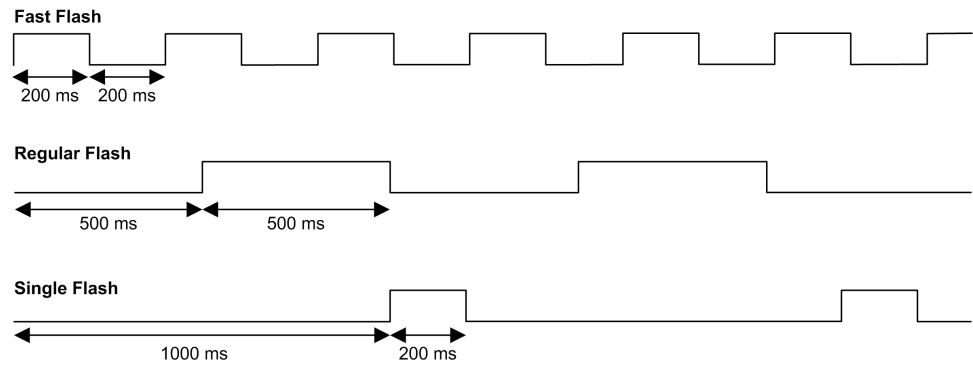
Controller States Table

The following tables describe the controller states:

Controller State	Description	LED Status
BOOTING	The controller executes the boot firmware and its own internal self-tests. It then verifies the checksum of the firmware and user applications.	Each LED, from the PWR LED to the NS or S3 LED, depending on the controller reference, flashes before turning solid green. The boot sequence is complete when all the LEDs are solid green. The LEDs then flash together briefly to indicate that the controller is operational.
INVALID_OS	There is not a valid firmware file present in the non-volatile memory or the firmware is not from Schneider Electric. The controller does not execute the application. Refer to the Updating Firmware , page 223 section to restore a correct state.	FSP LED stays solid red after the boot sequence.

Controller State	Description	LED		
		RUN (Green)	ERR (Red)	I/O (Red)
EMPTY	The controller has no application.	OFF	Single flash	OFF
EMPTY after a system error detected	This state is the same as the other EMPTY state. However, the application is present, and is intentionally not loaded. A reboot (power cycle), or a new application download, restores a correct state.	OFF	Fast flash	OFF
RUNNING	The controller is executing a valid application.	ON	OFF	OFF
RUNNING with breakpoint	This state is same as the RUNNING state with the following exceptions: - The task-processing portion of the program does not resume until the breakpoint is cleared. - The LED indications are different. For more information on breakpoint management, refer to EcoStruxure Machine Expert Programming Guide .	Single flash	OFF	OFF
RUNNING with external error detected	The controller is executing a valid application and a configuration, TM3, SD card, or other I/O error is detected. When I/O LED is ON, the details about the detected error can be found in <i>PLC_R.i_lwSystemFault_1</i> and <i>PLC_R.i_lwSystemFault_2</i> . Any of the detected error conditions reported by these variables cause the I/O LED to be ON.	ON	OFF	ON
STOPPED	The controller has a valid application that is stopped. See details of the STOPPED state, page 47 for an explanation of the behavior of outputs and field buses in this state.	Regular flash	OFF	OFF
STOPPED with external error detected	The controller has a valid application that is stopped and a configuration, TM3, SD card, or other I/O error is detected.	Regular flash	OFF	ON
HALT	The controller stops executing the application because it has detected an application error.	Regular flash	ON	–
Boot Application not saved	The controller has an application in memory that differs from the application in non-volatile memory. At next power cycle, the application is changed by the one from non-volatile memory.	ON or regular flash	Single flash	OFF

This figure shows the difference between the fast flash, the regular flash and single flash:



Details of the STOPPED State

The following statements are true for the STOPPED state:

- The input configured as the Run/Stop input remains operational.
- The output configured as the Alarm output remains operational and goes to a value of 0.
- Ethernet, Serial (Modbus, ASCII, and so on), and USB communication services remain operational and commands written by these services can continue to affect the application, the controller state, and the memory variables.
- WebVisualisation is not operational.
- Outputs initially assume their configured default state (**Keep current values** or **Set all outputs to default**) or the state dictated by output forcing if used. The subsequent state of the outputs depends on the value of the **Update IO while in stop** setting and on commands received from remote devices. For more information on the behavior of the TM3 outputs, refer to Modicon TM3 Expansion Modules Configuration - Programming Guide (see Modicon TM3, Expansion Modules, Programming Guide).

Task and I/O Behavior When Update IO While In Stop Is Selected	When the Update IO while in stop setting is selected: • The Read Inputs operation continues normally. The physical inputs are read and then written to the %I input memory variables. • The Task Processing operation is not executed. • The Write Outputs operation continues. The %Q output memory variables are updated to reflect either the Keep current values configuration or the Set all outputs to default configuration, adjusted for any output forcing, and then written to the physical outputs.
CANopen Behavior When Update IO While In Stop Is Selected	The following is true for the CANopen buses when the Update IO while in stop setting is selected: • The CANopen bus remains operational. Devices on the CANopen bus continue to perceive the presence of a functional CANopen Master. • TPDO and RPDO continue to be exchanged. • The optional SDO, if configured, continue to be exchanged. • The Heartbeat and Node Guarding functions, if configured, continue to operate. • If the Behaviour for outputs in Stop field is set to Keep current values, the TPDOs continue to be issued with the last values. • If the Behaviour for outputs in Stop field is Set all outputs to default the last values are updated to the default values and subsequent TPDOs are issued with these default values.

Task and I/O Behavior When Update IO While In Stop Is Not Selected	When the Update IO while in stop setting is not selected, the controller sets the I/O to either the Keep current values or Set all outputs to default condition (as adjusted for output forcing if used). After this, the following becomes true: • The Read Inputs operation ceases. The %I input memory variables are frozen at their last values. • The Task Processing operation is not executed. • The Write Outputs operation ceases. The %Q output memory variables can be updated via the Ethernet, Serial, and USB connections. However, the physical outputs are unaffected and retain the state specified by the configuration options.
CANopen Behavior When Update IO While In Stop Is Not Selected	The following is true for the CANopen buses when the Update IO while in stop setting is not selected: • The CANopen Master ceases communications. Devices on the CANopen bus assume their configured fallback states. • TPDO and RPDO exchanges cease. • Optional SDO, if configured, exchanges cease. • The Heartbeat and Node Guarding functions, if configured, stop. • The current or default values, as appropriate, are written to the TPDOs and sent once before stopping the CANopen Master.

State Transitions and System Events

Overview

This section begins with an explanation of the output states possible for the controller. It then presents the system commands used to transition between controller states and the system events that can also affect these states. It concludes with an explanation of the Remanent variables, and the circumstances under which different variables and data types are retained through state transitions.

Controller States and Output Behavior

Introduction

The Modicon M262 Logic/Motion Controller defines output behavior in response to commands and system events in a way that allows for greater flexibility. An understanding of this behavior is necessary before discussing the commands and events that affect controller states.

The possible output behaviors and the controller states to which they apply are:

- Managed by **Application Program**
- **Keep current values**
- **Set all outputs to default**
- Hardware **Initialization Values**
- Software **Initialization Values**
- **Output Forcing**

NOTE: For TM3 **Expert module** reflex output behavior, refer to Modicon TM3 Expansion Modules - Programming Guide.

Managed by Application Program

Your application program manages outputs normally. This applies in the RUNNING and RUNNING with External Error Detected states.

NOTE: An exception to this is if the RUNNING with External Error Detected state has been provoked by a I/O expansion bus error. For more information, refer to *I/O Configuration General Description*, page 102.

Keep Current Values

Select this option by choosing **Controller Editor > PLC settings > Behavior for outputs in Stop > Keep current values**. To access the Controller Editor, right-click on the controller in the **Devices tree** and select **Edit Object**.

This output behavior applies in the STOPPED controller state. It also applies to CAN bus in the HALT controller state. Outputs maintain their state, although the details of the output behavior vary greatly depending on the setting of the **Update I/O while in stop** option and the actions commanded via configured fieldbusses. Refer to *PLC Settings*, page 62 for more details on these variations.

Set All Outputs to Default

Select this option by choosing **Controller Editor > PLC settings > Behavior for outputs in Stop > Set all outputs to default**. To access the Controller Editor, right-click on the controller in the **Devices tree** and select **Edit Object**.

This output behavior applies:

- when the controller is going from RUNNING state to STOPPED state.
- if the controller is going from RUNNING state to HALT state.
- after application download.
- after reset warm/cold command.
- after a reboot.

It also applies to CAN bus in the HALT controller state. Outputs maintain their state, although the details of the output behavior vary greatly depending on the setting of the **Update I/O while in stop** option and the actions commanded via configured fieldbusses. Refer to *Controller States Description*, page 45 for more details on these variations.

Hardware Initialization Values

This output state applies in the BOOTING, EMPTY (following power cycle with no boot application or after the detection of a system error), and INVALID_OS states.

In the initialization state, analog, transistor, and relay outputs assume the following values:

- For an analog output: Z (high impedance)
- For a fast transistor output: Z (high impedance)
- For a regular transistor output: 0 Vdc
- For a relay output: Open

Software Initialization Values

This output state applies when downloading or when resetting the application. It applies at the end of the download or at the end of a reset warm or cold.

The software **Initialization Values** are the initialization values of outputs images (%I, %Q, or variables mapped on %I or %Q).

By default, they are set to 0 but it is possible to map the I/O in a GVL and assign to the outputs a value different than 0.

Output Forcing

The controller allows you to force the state of selected outputs to a defined value for the purposes of system testing, commissioning, and maintenance.

You are only able to force the value of an output while your controller is connected to EcoStruxure Machine Expert.

To do so, use the **Force values** command in the **Debug** menu.

Output forcing overrides other commands (except write immediate) to an output irrespective of the task programming that is being executed.

When you logout of EcoStruxure Machine Expert when output forcing has been defined, you are presented with the option to retain output forcing settings. If you select this option, the output forcing continues to control the state of the selected outputs until you download an application or use one of the Reset commands.

When the option **Update I/O while in stop**, if supported by your controller, is checked (default state), the forced outputs keep the forcing value even when the controller is in STOPPED state.

Output Forcing Considerations

The output you wish to force must be contained in a task that is currently being executed by the controller. Forcing outputs in unexecuted tasks, or in tasks whose execution is delayed either by priorities or events has no effect on the output. However, once the task that had been delayed is executed, the forcing takes effect at that time.

Depending on task execution, the forcing could impact your application in ways that may not be obvious to you. For example, an event task could turn on an output. Later, you may attempt to turn off that output but the event is not being triggered at the time. This would have the effect of the forcing being apparently ignored. Further, at a later time, the event could trigger the task at which point the forcing would take effect.

In case of any forced variable, the FSP LED is flashing red, regular flash.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have a thorough understanding of how forcing will affect the outputs relative to the tasks being executed.
- Do not attempt to force I/O that is contained in tasks that you are not certain will be executed in a timely manner, unless your intent is for the forcing to take affect at the next execution of the task whenever that may be.
- If you force an output and there is no apparent affect on the physical output, do not exit EcoStruxure Machine Expert without removing the forcing.

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

Commanding State Transitions

Run Command

Effect: Commands a transition to the RUNNING controller state.

Starting Conditions: BOOTING or STOPPED state.

Methods for Issuing a Run Command:

- Refer to Run/Stop Input, page 80 for more information.

- EcoStruxure Machine Expert Online Menu: Select the **Start** command.
- RUN command from Web server
- By an external call via Modbus request using the PLC_W.q_wPLCControl and PLC_W.q_uiOpenPLCControl system variables of the M262 System library.
- **Login with online change** option: An online change (partial download) initiated while the controller is in the RUNNING state returns the controller to the RUNNING state if successful.
- **Multiple Download** Command: sets the controllers into the RUNNING state if the **Start all applications after download or online change** option is selected, irrespective of whether the targeted controllers were initially in the RUNNING, STOPPED or EMPTY state.
- The controller is restarted into the RUNNING state automatically under certain conditions.

Refer to [Controller State Diagram](#), page 42 for further details.

Stop Command

Effect: Commands a transition to the STOPPED controller state.

Starting Conditions: BOOTING, EMPTY, or RUNNING state.

Methods for Issuing a Stop Command:

- Run/Stop Input: If configured, command a value of 0 to the Run/Stop input. Refer to [Run/Stop Input](#), page 80 for more information.
- EcoStruxure Machine Expert Online Menu: Select the **Stop** command.
- STOP command from Web server
- By an internal call by the application or an external call via Modbus request using the PLC_W.q_wPLCControl and PLC_W.q_uiOpenPLCControl system variables of the M262 System library.
- **Login with online change** option: An online change (partial download) initiated while the controller is in the STOPPED state returns the controller to the STOPPED state if successful.
- **Download** Command: implicitly sets the controller into the STOPPED state.
- **Multiple Download** Command: sets the controllers into the STOPPED state if the **Start all applications after download or online change** option is not selected, irrespective of whether the targeted controllers were initially in the RUNNING, STOPPED or EMPTY state.
- REBOOT by Script: The file transfer script on an SD card can issue a REBOOT as its final command. The controller is rebooted into the STOPPED state provided the other conditions of the boot sequence allow this to occur. Refer to [Reboot](#), page 55 for further details.
- The controller is restarted into the STOPPED state automatically under certain conditions.

Refer to [Controller State Diagram](#), page 42 for further details.

Reset Warm

Effect: Resets the variables, except for the remanent variables, to their default values. Places the controller into the STOPPED state.

Starting Conditions: RUNNING, STOPPED, or HALT states.

Methods for Issuing a Reset Warm Command:

- EcoStruxure Machine Expert Online Menu: Select the **Reset warm** command.
- By an internal call by the application or an external call via Modbus request using the PLC_W.q_wPLCControl and PLC_W.q_uiOpenPLCControl system variables of the M262 System library.

Effects of the Reset Warm Command:

1. The application stops.
2. Forcing is erased.
3. Diagnostic indications for errors are reset.
4. The values of the retain variables are maintained.
5. The values of the retain-persistent variables are maintained.
6. The non-located and non-remanent variables are reset to their initialization values.
7. The values of the 0...59999 %MW registers are reset to 0.
8. The fieldbus communications are stopped and then restarted after the reset is complete.
9. The inputs are reset to their initialization values. The outputs are reset to their software initialization values or their default values if no software initialization values are defined.
10. The Post Configuration file is read, page 215.

For details on variables, refer to [Remanent Variables](#), page 58.

Reset Cold

Effect: Resets the variables, except for the retain-persistent type of remanent variables, to their initialization values. Places the controller into the STOPPED state.

Starting Conditions: RUNNING, STOPPED, or HALT states.

Methods for Issuing a Reset Cold Command:

- EcoStruxure Machine Expert Online Menu: Select the **Reset cold** command.
- By an internal call by the application or an external call via Modbus request using the PLC_W.q_wPLCControl and PLC_W.q_uiOpenPLCControl system variables of the M262 System library.

Effects of the Reset Cold Command:

1. The application stops.
2. Forcing is erased.
3. Diagnostic indications for errors are reset.
4. The values of the retain variables are reset to their initialization value.
5. The values of the retain-persistent variables are maintained.
6. The non-located and non-remanent variables are reset to their initialization values.
7. The values of %MW0 to %MW59999 registers are reset to 0.
8. The fieldbus communications are stopped and then restarted after the reset is complete.
9. The inputs are reset to their initialization values. The outputs are reset to their software initialization values or their default values if no software initialization values are defined.
10. The Post Configuration file is read, page 215.

For details on variables, refer to [Remanent Variables](#), page 58.

Reset Origin

Effect: Resets all variables, including the remanent variables, to their initialization values. Erases all user files on the controller, including user rights and certificates. Reboots and places the controller into the EMPTY state.

Starting Conditions: RUNNING, STOPPED, or HALT states.

Methods for Issuing a Reset Origin Command:

- EcoStruxure Machine Expert Online Menu: Select the **Reset origin** command.

Effects of the Reset Origin Command:

1. The application stops.
2. Forcing is erased.
3. The WebVisualisation files are erased.
4. The user files (Boot application, Post Configuration, App, App/MFW, Cfg) are erased.
5. Diagnostic indications for errors are reset.
6. Nodename of the controller is reset to the default value.
7. The values of the retain variables are reset.
8. The values of the retain-persistent variables are reset.
9. The non-located and non-remanent variables are reset.
10. The fieldbus communications are stopped.
11. The other inputs are reset to their initialization values.
The other outputs are reset to their hardware initialization values.
Security certificates are erased.
12. Controller reboots.
13. FwLog.txt is maintained and all other System Log files are erased.

For details on variables, refer to [Remanent Variables](#), page 58.

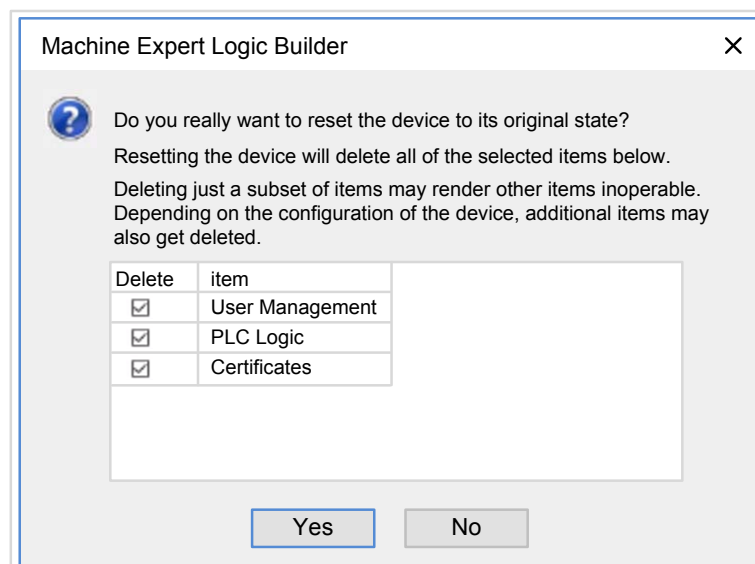
Reset Origin Device

Effect: Resets all variables, including the remanent variables, to their initialization values. Places the controller into the EMPTY state if **PLC Logic** is selected.

Starting Conditions: RUNNING, STOPPED, or HALT states.

Methods for Issuing a Reset Origin Device Command:

- EcoStruxure Machine Expert Online Menu: Right-click **My controller > Reset Origin Device** command. **Result:** a dialog box allows you to select the items to remove:
 - **User Management**
 - **PLC Logic**
 - **Certificates**



When **User Management** is selected:

- User and groups are reset to default value.

NOTE: If the controller **user rights** are disabled before this command is used, you can connect to the controller without login prompt afterwards. Use the dedicated command in Online menu: **Security > Reset user rights management to default** to enforce again the use of user management.

When **PLC Logic** is selected:

1. The application stops.
2. Forcing is erased.
3. The WebVisualisation files are erased.
4. Diagnostic indications for errors are reset.
5. The values of the retain variables are reset.
6. The values of the retain-persistent variables are reset.
7. The non-located and non-remanent variables are reset.
8. The fieldbus communications are stopped.
9. Embedded Expert I/O are reset to their previous user-configured default values.
10. The other inputs are reset to their initialization values.
The other outputs are reset to their hardware initialization values.
11. System Logs are maintained.

When **Certificates** is selected:

- Certificate used for encrypted communication is reset.
- Certificates used for Web server, FTP server and OPC UA server/Client are not reset.

For details on variables, refer to [Remanent Variables](#), page 58.

Reboot

Effect: Commands a reboot of the controller.

Starting Conditions: Any state.

Methods for Issuing the Reboot Command:

- Power cycle
- REBOOT by Script

Effects of the Reboot:

1. The state of the controller depends on a number of conditions:

- a. The controller state is RUNNING if:

The Reboot was provoked by a power cycle and:

- the **Starting Mode** is set to **Start in run**, and if the Run/Stop input is not configured, and if the controller was not in HALT state before the power cycle, and if the remanent variables are valid.

- the **Starting Mode** is set to **Start in run**, and if the Run/Stop input is configured and set to RUN, and if the controller was not in HALT state before the power cycle, and if the remanent variables are valid.

- the **Starting Mode** is set to **Start as previous state**, and Controller state was RUNNING before the power cycle, and if the Run/Stop input is not configured and the boot application has not changed and the remanent variables are valid.

- the **Starting Mode** is set to **Start as previous state**, and Controller state was RUNNING before the power cycle, and if the Run/Stop input is configured and is set to RUN and the remanent variables are valid.

The Reboot was provoked by a script and:

- the **Starting Mode** is set to **Start in run**, and if the Run/Stop input is configured and set to RUN, or the switch is set to RUN, and if the controller was not in HALT state before the power cycle, and if the remanent variables are valid.

- b. The controller state is STOPPED if:

The Reboot was provoked by a power cycle and:

- the **Starting Mode** is set to **Start in stop**.

- the **Starting Mode** is set to **Start as previous state** and the controller state was not RUNNING before the power cycle.

- the **Starting Mode** is set to **Start as previous state** and the controller state was RUNNING before the power cycle, and if the Run/Stop input is not configured, and if the boot application has changed.

- the **Starting Mode** is set to **Start as previous state** and the controller state was RUNNING before the power cycle, and if the Run/Stop input is not configured, and if the boot application has not changed, and if the remanent variables are not valid.

- the **Starting Mode** is set to **Start as previous state** and the controller state was RUNNING before the power cycle, and if the Run/Stop input is configured and is set to STOP.

- the **Starting Mode** is set to **Start in run** and if the controller state was HALT before the power cycle.

- the **Starting Mode** is set to **Start in run**, and if the controller state was not HALT before the power cycle, and if the Run/Stop input is configured and is set to STOP.

- the **Starting Mode** is set to **Start as previous state**, and if the Run/Stop input is configured and set to RUN, or the switch is set to RUN, and if the controller was not in HALT state before the power cycle.

- the **Starting Mode** is set to **Start as previous state**, and if the Run/Stop input is configured and set to RUN, or the switch is set to RUN, HALT state before the power cycle.

- c. The controller state is EMPTY if:
 - There is no boot application or the boot application is invalid, or
 - The reboot was provoked by specific System Errors.
 - d. The controller state is INVALID_OS if there is no valid firmware.
2. Forcing is maintained if the boot application is loaded successfully. If not, forcing is erased.
 3. Diagnostic indications for errors are reset.
 4. The values of the retain variables are restored if saved context is valid.
 5. The values of the retain-persistent variables are restored if saved context is valid.
 6. The non-located and non-remanent variables are reset to their initialization values.
 7. The values of %MW0 to %MW59999 registers are reset to 0.
 8. The fieldbus communications are stopped and restarted after the boot application is loaded successfully.
 9. The inputs are reset to their initialization values. The outputs are reset to their hardware initialization values and then to their software initialization values or their default values if no software initialization values are defined.
 10. The Post Configuration file is read, page 215.
 11. The controller file system is initialized and its resources (sockets, file handles, and so on) are deallocated.

The performance of the boot-up time of the controller depends on the number of files stored in its file system. Reducing their number as much as possible allows you to obtain better performance.

The file system employed by the controller needs to be periodically re-established by a power cycle of the controller. If you do not perform regular maintenance of your machine, or if you are using an Uninterruptible Power Supply (UPS), you must force a power cycle (removal and reapplication of power) to the controller at least once a year.

NOTICE

DEGRADATION OF PERFORMANCE

Reboot your controller at least once a year by removing and then reapplying power.

Failure to follow these instructions can result in equipment damage.

For details on variables, refer to *Remanent Variables*, page 58.

NOTE: The Check context test concludes that the context is valid when the application and the remanent variables are the same as defined in the Boot application.

NOTE: If you provide power to the Run/Stop input from the same source as the controller, the loss of power to this input is detected immediately, and the controller behaves as if a STOP command was received. Therefore, if you provide power to the controller and the Run/Stop input from the same source, your controller reboots normally into the STOPPED state after a power interruption when **Starting Mode** is set to **Start as previous state**.

NOTE: If you make an online change to your application program while your controller is in the RUNNING or STOPPED state but do not manually update your Boot application, the controller detects a difference in context at the next reboot, the remanent variables are reset as per a Reset cold command, and the controller enters the STOPPED state.

Download Application

Effect: Loads your application executable into the RAM memory. Optionally, creates a Boot application in the non-volatile memory.

Starting Conditions: RUNNING, STOPPED, HALT, and EMPTY states.

Methods for Issuing the Download Application Command:

- EcoStruxure Machine Expert:
 - 2 options exist for downloading a full application:
 - Download command.
 - Multiple Download command.
 - For important information on the application download commands, refer to Controller State Diagram.
- FTP: Load Boot application file to the non-volatile memory using FTP. The updated file is applied at the next reboot.
- SD card: Load Boot application file using an SD card in the controller. The updated file is applied at the next reboot. Refer to File Transfer with SD Card for further details.

Effects of the EcoStruxure Machine Expert Download Command:

1. The existing application stops and then is erased.
2. If valid, the new application is loaded and the controller assumes a STOPPED state.
3. Forcing is erased.
4. Diagnostic indications for errors are reset.
5. The values of the retain variables are reset to their initialization values.
6. The values of any existing retain-persistent variables are maintained.
7. The non-located and non-remanent variables are reset to their initialization values.
8. The values of %MW0 to %MW59999 registers are reset to 0.
9. The fieldbus communications are stopped and then the configured fieldbus of the new application is started after the download is complete.
10. Embedded Expert I/O are reset to their previous user-configured default values and then set to the new user-configured default values after the download is complete.
11. The inputs are reset to their initialization values. The outputs are reset to their hardware initialization values and then to their software initialization values or their default values if no software initialization values are defined, after the download is complete.
12. The Post Configuration file is read, page 215.

For details on variables, refer to Remanent Variables, page 58.

Effects of the FTP or SD Card Download Command:

There are no effects until the next reboot. At the next reboot, the effects are the same as a reboot with an invalid context. Refer to Reboot, page 55.

Error Detection, Types, and Management

Error Management

The controller detects and manages three types of errors:

- External errors
- Application errors
- System errors

This table describes the types of errors that may be detected:

Type of Error Detected	Description	Resulting Controller State
External Error	External errors are detected by the system while RUNNING or STOPPED but do not affect the ongoing controller state. An external error is detected in the following cases: - A connected device reports an error to the controller. - The controller detects an error with an external device, for example, when the external device is communicating but not properly configured for use with the controller. - The controller detects an error with an output. - The controller detects a communication interruption with a device. - The controller is configured for an expansion module that is not present or not detected, and has not otherwise been declared as an optional module ⁽¹⁾. - The boot application in non-volatile memory is not the same as the one in RAM. - The I/O LED is red ON.	RUNNING with External Error Detected Or STOPPED with External Error Detected
Application Error	An application error is detected when improper programming is encountered or when a task watchdog threshold is exceeded. The ERR LED is red ON.	HALT
System Error	A system error is detected when the controller enters a condition that cannot be managed during runtime. Most such conditions result from firmware or hardware exceptions, but there are some cases when incorrect programming can result in the detection of a system error, for example, when attempting to write to memory that was reserved during runtime, or when a system watchdog occurs. The ERR LED is fast flashing RED. NOTE: There are some system errors that can be managed by runtime and are therefore treated like application errors.	BOOTING → EMPTY
(1) Expansion modules may appear to be absent for any number of reasons, even if the absent I/O module is physically present on the bus. For more information, refer to I/O Configuration General Description, page 102.		

NOTE: Refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide) for more detailed information on diagnostics.

Remanent Variables

Overview

Remanent data refers to variables that are defined in Programming Organization Units (POUs) as retain or retain-persistent. In the event of power outages, reboots, resets, and application program downloads, remanent variables can either be reinitialized or retain their values.

This table describes the behavior of remanent variables in each case:

Action	VAR	VAR RETAIN	VAR GLOBAL RETAIN PERSISTENT
Online change to application program	X	X	X
Online change modifying the boot application ⁽¹⁾	–	X	X
Stop	X	X	X
Power cycle	–	X	X
Reset warm	–	X ⁽²⁾	X
Reset cold	–	–	X
Reset origin	–	–	–
Reset origin device	–	–	–
Download of application program using EcoStruxure Machine Expert ⁽³⁾	–	–	X
Download of application program using an SD card ⁽³⁾	–	–	–

X The value is maintained.

– The value is reinitialized.

(1) Retain variable values are maintained if an online change modifies only the code part of the boot application (for example, `a:=a+1; => a:=a+2;`). In all other cases, retain variables are reinitialized.

(2) For more details on VAR RETAIN, refer to [Effects of the Reset warm Command](#), page 51.

(3) If the downloaded application contains the same retain-persistent variables as the existing application, the existing retain variables maintain their values.

Adding Retain-Persistent Variables

Declare retain-persistent (**VAR GLOBAL PERSISTENT RETAIN**) variables in the **PersistentVars** window:

Step	Action
1	In the Applications tree , select the Application node.
2	Click the right mouse button.
3	Choose Add Objects > Persistent variables
4	Click Add . Result: The PersistentVars window is displayed.

Retain and Persistent Variables: Performance Impact

Retain or retain-persistent variables are located in a dedicated non-volatile memory. Each time these variables are accessed during Programming Organization Unit (POU) execution, the non-volatile memory is accessed. The access time of these variables is slower than the access time of regular variables, which can impact performance. This is an important fact to take into account when writing performance-sensitive POUs.

For more information about the impact of retain and retain persistent variables on cycle time during POU execution, see [Processing Performance](#), page 272.

Controller Device Editor

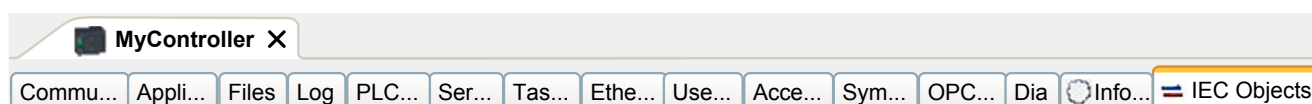
Introduction

This chapter describes how to configure the controller.

Controller Parameters

Controller Parameters

To open the device editor, double-click **MyController** in the **Devices tree**:



Tabs Description

Tab	Description	Restriction
Communication settings , page 61	Manages the connection between the PC and the controller: - Helping you find a controller in a network, - Presenting the list of available controllers, so you can connect to the selected controller and manage the application in the controller, - Helping you physically identify the controller from the device editor, - Helping you change the communication settings of the controller. The controller list is detected through NetManage or through the Active Path based on the communication settings. To access the Communication settings, click Project > Project Settings... in the menu bar. For more information, refer to the EcoStruxure Machine Expert Programming Guide (<i>Communication Settings</i>).	–
Applications	Presents the application running on the controller and allows removing the application from the controller. If the state is EMPTY , boot application is deleted.	Online mode only
Files , page 26	File management between the PC and the controller. Only one controller disk at a time can be seen through this tab. This tab displays the content of the <i>/usr</i> directory of the internal non-volatile memory of the controller.	Online mode only
Log	View the controller log file.	Online mode only
PLC settings , page 62	Configuration of: - Starting mode options - I/O behavior in stop - Bus cycle options	–
Services , page 63	Date and time settings, versions.	Online mode only
IEC Objects	Allows you to access to the device from the IEC application through the listed objects. Displays a monitoring view in online mode. For more information, refer to IEC Object in CODESYS Online Help.	–
Task deployment	Displays a list of I/Os and their assignments to tasks.	After compilation only
Ethernet Services , page 64	Ethernet_1 and Ethernet_2 tabs summarize the Ethernet connections. The IP Routing tab allows you to configure the routes and the cross network transparency through IP Forwarding options. The Ethernet Resources tab allows you to calculate the number of connections and channels configured.	–

Tab	Description	Restriction
Users and Groups , page 69	The Users and Groups tab is provided for devices supporting online user management. It allows setting up users and access-rights groups and assigning them access rights to control the access on EcoStruxure Machine Expert projects and devices in online mode. For more information, refer to the EcoStruxure Machine Expert Programming Guide.	–
Access Rights , page 69	The Access Rights tab allows you to define the device access rights of users. For more information, refer to the EcoStruxure Machine Expert Programming Guide .	–
Symbol Rights	Allows the Administrator to configure Users and Groups access to the symbol sets. For more information, refer to Symbol Configuration in CODESYS Online Help.	–
OPC UA Server Configuration	Displays the OPC UA Server Configuration, page 202 window.	–
Diagnostic Table	Displays the data of the controller. The data can be accessed using the syntax <code>NameOfControllerInDeviceTree.NameofParameter</code> . Example: <code>MyController.SA_NbPowerOn</code> .	Online mode only
Information	Displays general information about the device (name, description, provider, version, image).	–

Communication Settings

Introduction

This tab allows you to manage the connection from the PC to the controller:

- Helping you find a controller in a network.
- Presenting the list of controllers, so you can connect to the selected controller and manage the application inside the controller.
- Helping you physically identify the controller from the device editor.
- Helping you change the communication settings of the controller.

Editing Communication Settings

The **Edit communication settings** window lets you change the Ethernet communication settings. To do so, click **Communication settings** tab. The list of controllers available in the network appears. Select and right-click the required row and click **Edit communication settings...** in the contextual menu.

You can configure the Ethernet settings in the **Edit communication settings** window in two ways:

- Without the **Save settings permanently** option:
Configure the communication parameters and click **OK**. These settings are immediately taken into account and are not kept if the controller is reset. For the next resets, the communication parameters configured into the application are taken into account.
- With the **Save settings permanently** option:
You can also activate the **Save settings permanently** option before you click **OK**. Once this option is activated, the Ethernet parameters configured here are taken into account on reset instead of the Ethernet parameters configured into the EcoStruxure Machine Expert application.

For more information on the **Communication settings** view of the device editor, refer to the EcoStruxure Machine Expert Programming Guide.

Editing the communication settings modifies the settings of the Ethernet interface used for the connection.

NOTE: If you are connected by USB, the Ethernet_2 settings are modified.

NOTE: Click the update icon to apply the changes.

PLC Settings

Overview

The figure below presents the **PLC Settings** tab:

Element		Description
Application for I/O handling		By default, set to Application because there is only one application in the controller.
PLC settings	Update IO while in stop	If this option is activated (default), the values of the input and output channels are also updated when the controller is stopped.
	Behavior for outputs in Stop	From the selection list, choose one of the following options to configure how the values at the output channels should be handled in case of controller stop: • Keep current values • Set all outputs to default
	Always update variables	By default, set to Enabled 1 (use bus cycle task if not used in task) and not editable.
Bus cycle options	Bus cycle task	This configuration setting is the parent for all Bus cycle task parameters used in the application Devices tree. Some devices with cyclic calls, such as a CANopen manager, can be attached to a specific task. In the device, when this setting is set to Use parent bus cycle setting, the setting set for the controller is used. The selection list offers all tasks currently defined in the active application. The default setting is the MAST task. NOTE: <unspecified> means that the task is in "slowest cyclic task" mode.
Additional settings	Generate force variables for IO mapping	Not used.
	Enable Diagnosis for devices	Not used.
	Show I/O warnings as errors	Not used.

Element		Description
Starting mode Options	Starting mode	This option defines the starting mode on a power-on. For further information, refer to State behavior diagram, page 42. Select with this option one of these starting modes: • Start as previous state • Start in stop • Start in run

Services

Services Tab

The **Services** tab is divided in three parts:

- RTC Configuration
- Device Identification
- Post Configuration

The figure below shows the **Services** tab:

RTC Configuration

PLC Time

Read

Local Time

Date: Tuesday 6 September 2016

Time: 16:24:27

Write

☒ Write as UTC

Synchronize with local's date/time

Device Identification

Firmware Version:

Boot Version:

Coprocesor Version:

Post Configuration

Parameters overwritten by the Post configuration:

Read

NOTE: To have controller information, you must be connected to the controller.

NOTE: RTC information can be configured by Web server or using the **SysTimeRtcSet** function block. For more information, refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide.

NOTE: When the controller is configured as NTP client in the tab **Ethernet Services**, time is in UTC and cannot be set manually.

Element		Description
RTC Configuration	PLC Time	Displays the date and time read from the controller when the Read button is clicked, with no conversion applied. This read-only field is initially empty. If Write as UTC is selected, PLC Time is in Coordinated Universal Time (UTC).
	Read	Reads the date and time saved on the controller and displays the values in the PLC Time field.
	Local Time	Lets you define a date and a time that are sent to the controller when the Write button is clicked. If necessary, modify the default values before clicking the Write button. A message box informs you about the result of the command. The date and time fields are initially filled with the current PC settings.
	Write	Writes the date and time defined in the Local time field to the controller. A message box informs you of the result of the command. Select the Write as UTC checkbox before running this command to write the values in UTC format.
	Synchronize with local date/time	Lets you directly send the PC settings. A message box informs you of the result of the command. Select Write as UTC before running this command to use UTC format. Use UTC time when using secure communication.
Device Identification		Displays the Firmware Version , the Boot Version , and the Coprocessor Version of the selected controller, if connected.
Post Configuration		Displays the application parameters overwritten by the Post configuration, page 215.

Ethernet Services

Presentation

This tab displays the list of Ethernet or Sercos devices which are configured to be controlled by Modicon M262 Logic/Motion Controller.

- **Ethernet_1**
- **Ethernet_2**
- **Ethernet Resources**
- **IP Routing**
- **NTP**

Ethernet_1 and Ethernet_2 Toolbar

The following table describes the toolbar:

Element	Description
Generate IP address	Allows you to generate the configurations of each device configured in the Devices tree .
Filter Options	Allows you to display more information on the configured devices.
Discover devices	Start the Machine Assistant which allows you to discover and to configure the devices.

Network Settings

To view the configuration of a device, click the tab above the toolbar. The following information displays:

- **IP Address**

- **Subnet Mask**
- **Gateway**
- **Subnet Address**

Configured Devices in the Project

Element	Description	Restriction
Device Name	Name of the device from the Devices tree . Click the device name to access the device configuration.	Cannot be edited.
Device Type	Type of the device.	Cannot be edited.
IP Address	IP Address of the device. Can be left blank for Sercos devices	–
MAC Address	MAC address of the target device. Can be left blank for Sercos devices	Can be edited if IP Address by BOOTP selected in the configuration of the device.
DHCP Device Name	Hostname of the target device	Can be edited if IP Address by DHCP selected in the configuration of the device.
Subnet Mask	Subnet mask of the device	Visible if Expert Mode selected in Filter Options .
Gateway Address	Gateway address of the device	Visible if Expert Mode selected in Filter Options .
Identified by	Four identification modes are possible: • None • Fixed • BOOTP • DHCP	–
Protocol	Protocol used	Cannot be edited.
Identifier	Identifier of the device	Can be edited for Sercos device.
Identification Mode	Identification mode of the device	Can be edited for Sercos device.
Operating Mode	Three operating modes are possible: • Activated • Simulated • Optional	Can be edited for Sercos device.

Ethernet Resources

The **Ethernet Resources** subtab:

- Displays the number of configured connections and channels.
- Displays the number of input words.
- Displays the number of output words.
- Displays the scanner load.

IP Routing

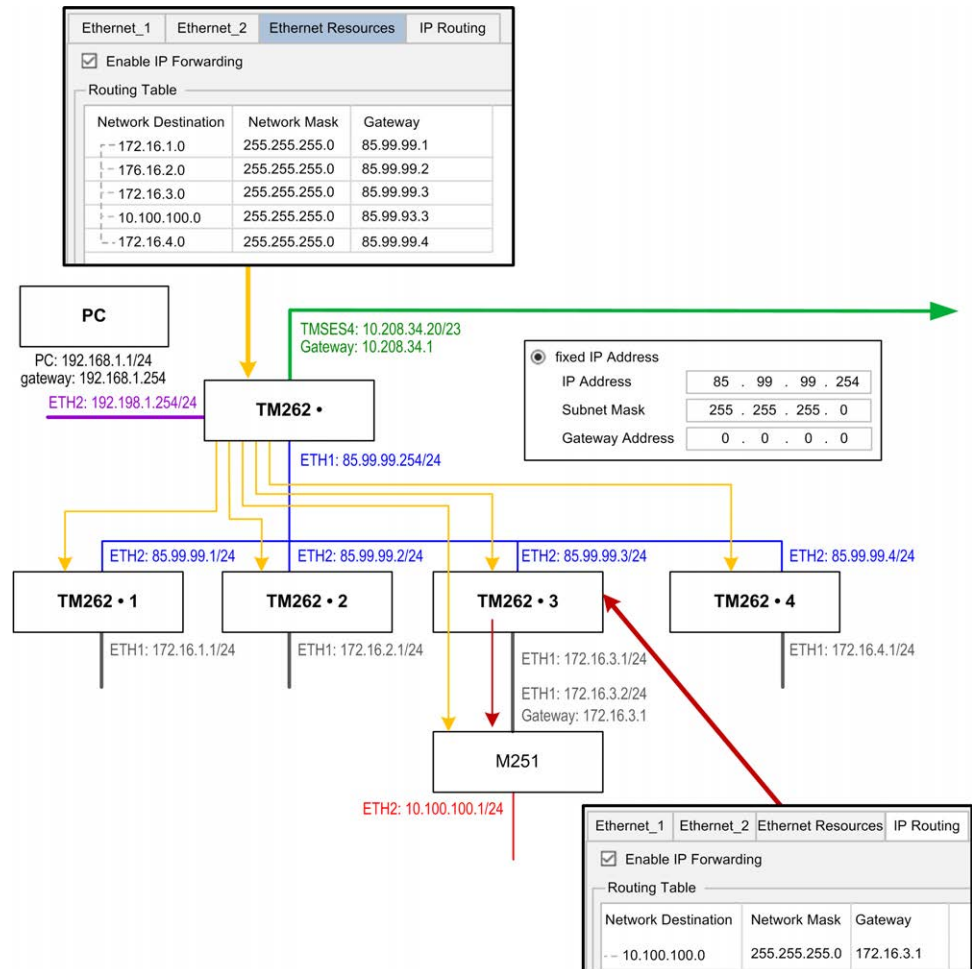
The **IP Routing** subtab allows you to configure the IP routes in the controller.

The parameter **Enable IP forwarding** allows you to deactivate the IP forwarding service of the controller. When deactivated, the communication is not routed from a network to another one. The devices on the device network are no longer accessible from the control network and related features like Web pages access on device or commissioning of device via DTM, EcoStruxure Machine Expert - Safety and so on are not possible.

The Modicon M262 Logic/Motion Controller can have up to three Ethernet interfaces. Using a routing table is necessary to communicate with remote

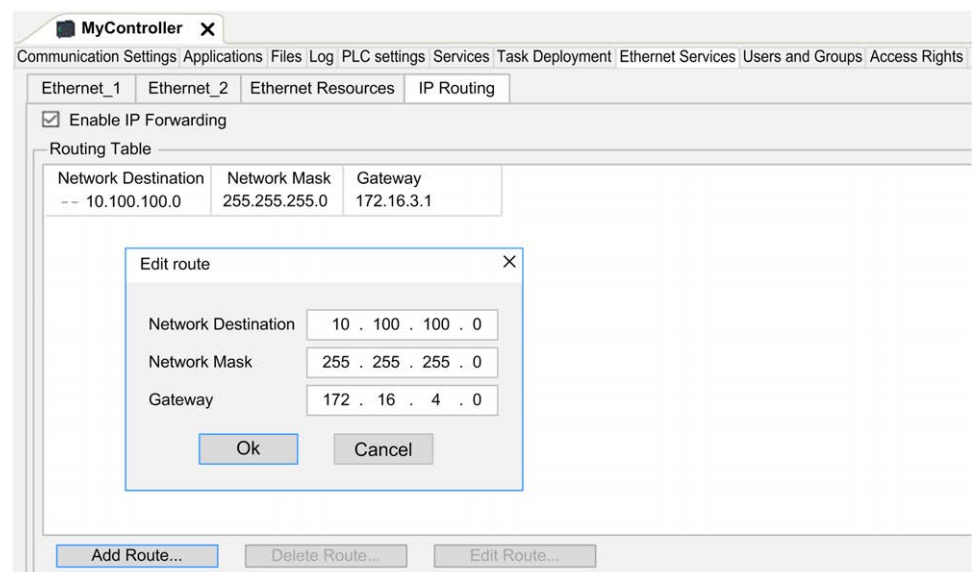
networks connected to different Ethernet interfaces. The gateway is the IP address used to connect to the remote network, which needs to be in local network of the controller.

This graphic depicts an example network, in which the last two rows of devices (gray and red) need to be added in the routing table:



Use the routing tables to manage the IP forwarding.

To add a route, double click **My controller** then click **Ethernet Services > IP Routing > Add Route**.



For reasons of network security, TCP/IP forwarding is disabled by default. Therefore, you must manually enable TCP/IP forwarding if you want to access

devices through the controller. However, doing so may expose your network to possible cyberattacks if you do not take additional measures to protect your enterprise. In addition, you may be subject to laws and regulations concerning cybersecurity.

⚠ WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT NETWORK INTRUSION

- Observe and respect any an all pertinent national, regional and local cybersecurity and/or personal data laws and regulations when enabling TCP/IP forwarding on an industrial network.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.

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

NTP

The **NTP** Protocol synchronizes the clock of device and resists the effects of variable latency (jitter).

The **NTP** subtab is divided in three parts:

- **General** (1)
- **Client** (2)
- **Server** (3)

The figure below shows the **NTP** subtab:

MyController X

Communication Settings Applications Files Log PLC settings Services Task Deployment Ethernet Services

Ethernet 1 Ethernet 2 Ethernet Resources IP Routing NTP

General

Symmetric key file

Client

	IP Address	Min Poll	Max Poll	Key Index
Primary Server	115.165.145.2	8	12	6
Secondary Server	145.156.148.7	10	10	119

Server

☒ Enable NTP Server

Orphan Stratum

Symmetric keys (Count: 4)

Double-click to edit

- 1
- 5
- 15
- 56

General section

Element	Description
/usr/Cfg*	Folder to which the trusted key file is to be uploaded. Not editable.
Empty*	File name of the Symmetric keys file. Editable. Can be left empty if no key index is defined. - Maximum length: 22 characters - File extension: .keys - Allowed characters: a...z, A...Z, 0...9, -, _ NOTE: You must enter a valid file name or leave the field empty. NOTE: The only authentication method for the key algorithm is MD5 for NTP.

Client section

Element	Description	Value	Constraint
IP Address	The server IP Address.	Default value: 0.0.0.0	- The address must be used by another server - First byte must be between 1 and 223 - Loopback address is forbidden
Min Poll	The minimum poll value.	Default value: 6 Value range: 3...17 ⁽¹⁾	Minimum poll value must be inferior to maximum poll value.
Max Poll	The maximum poll value.	Default value: 10 Value range: 3...17 ⁽¹⁾	Maximum poll value must be superior to minimum poll value.
Key Index	The key index value.	Default value: 0 Value range: 0...65535	0 means "no key index".

(1): 3 corresponds to 8 seconds (2^3), 17 corresponds to 131072 seconds (2^{17}).

Server section

Element	Description	Value	Constraint
Enable NTP Server	Allows you to enable/disable the NTP Server.	Checked/unchecked	You must define stratum for orphan mode or NTP Client Primary Server if NTP Server is enabled.
Orphan Stratum	The orphan stratum level.	Default value: 0 Value range: 0...15	0 means: no Orphan Stratum . See Orphan Stratum , page 68.
Symmetric Keys	The list of key indexes.	Value range: 1...65535	Maximum of 32 key indexes, including Primary Server and Secondary Server key indexes.

NOTE: If you are using the default NTPv3 server of Microsoft Windows, the following configuration should be done on the server: [Configuring Systems for High Accuracy](#).

Orphan Stratum

NTP uses a hierarchical system where each level is called a stratum. These levels are assigned a number starting at 0 for the reference at the top level.

When the controller is both client and server, the stratum is calculated automatically from the NTP server it is connected to. When the **Orphan Stratum** is 0, if the NTP server used by the controller becomes unreachable, the controller indicates to its NTP client that its clock is not synchronized. Otherwise, the value selected is used.

If the controller is only configured as NTP server, it will use the selected value in **Orphan Stratum**. You should select an appropriate stratum value according to the NTP hierarchy of your architecture.

NTP Keys File Syntax Usage

- NTP keys file only supports MD5 hash algorithm.
- The keys file must not have a header.
- No spaces allowed at the beginning line of a key.
- If you insert a comment at the end of a key line, you must add two spaces between the end of the key and the beginning of the comment.

Key file syntax examples:

```
MD5 3N:}7LtY<Uz+FG5y65c4 # MD5 hash algorithm
MD5 37R}sQ^~)S~F*HZY(/w\ # MD5 hash algorithm
MD5 Mv4[@;x$f:D"_5_l>]t{ # MD5 hash algorithm
MD5 ':CHFQ^DvQ0JlAjhP\4, # MD5 hash algorithm
MD5 &`!~)4Oem@Xz|M{Hb&bY # MD5 hash algorithm
```

Users Rights

Introduction

Users rights contain the following elements: **User**, **Group**, **Object**, **Operation**, **User Rights**, **Access rights**. These elements allow you to manage users accounts and users access rights to control the access on the global projects.

- A **User** is a person or a service with specific **User Rights**.
- A **Group** is a **Persona** or a **Function**. It is predefined or added. Each **Group** provides accesses thanks to **Object**.
- An **Object** is composed by predefined accesses thanks to **Operation**.
- An **Operation** is the elementary action possible.
- **User Rights** are the possible **Access rights**: **VIEW**, **MODIFY**, **EXECUTE** and **ADD-REMOVE** for the dedicated operation.

For more information, refer to the EcoStruxure Machine Expert Programming Guide (see EcoStruxure Machine Expert, Programming Guide).

Login and passwords

Login and password are not set by default. This table describes how to log in:

Server/feature	First connection or connection after reset to default / reset origin / reset origin device	User Rights enabled	Connection after User Rights disabled
EcoStruxure Machine Expert	You must first create your login and your password. NOTE: The login and the password that you create during the first connection have administrator privileges. NOTE: For information on lost login and passwords, see Troubleshooting , page 78.	Login: configured login Password: configured password	No login or password required.
Web server	No login possible	Login: configured login Password: configured password	Login: Anonymous Password: no password required.
FTP server	No login possible	Login: configured login Password: configured password	Login: Anonymous Password: Anonymous
OPC-UA	No login possible	Login: configured login Password: configured password	Login: Anonymous Password: Anonymous
Change Device Name feature	No login possible	Login: configured login Password: configured password	No login or password required.

WARNING

UNAUTHORIZED DATA AND/OR APPLICATION ACCESS

- Secure access to the FTP/Web/OPC-UA server(s) using User Rights.
- If you disable User Rights, disable the server(s) to prevent any unwanted or unauthorized access to your application and/or data.

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

NOTE: Anonymous login can be restored by disabling the user rights in **User Management** page of the Web server, page 135.

NOTE: The following characters are supported by the controller:

- login: **a...z A...Z 0...9 – = [] \ ; ' , . / @ # \$ % ^ & * () _ + { } | : “ < > ? ` ~**
- password: **a...z A...Z 0...9 – = [] \ ; ' , . / @ # \$ % ^ & * () _ + { } | : “ < > ? ` ~ and space**

The length is limited to 60 characters.

User

The **User** must be defined by the **Administrator** and must be linked with one or several **Groups**.

Default groups

This table indicates the name and description of the predefined default **groups**:

NOTE: Administrator can define a new **Group** if needed.

Group Name	Group Description
Administrator	- Manages all the user rights. - Is created at first connection.
Persona	
Persona Designer/Programmer	Group dedicated to the design of the application.
Persona Operator	Group dedicated to the usage of the application.
Persona Web Designer	Group dedicated to the management of the Web server.
Persona Communication	Group dedicated to the management of communication features.
Persona Maintenance	Group dedicated to the maintenance of the application.
Function	
Function External Media	Group to allow the usage of External Command (from SD Card).
Function File Access	Group to allow permissions on files tab.
Function FTP	Group to allow usage of FTP.
Function Symbol Configuration	Group to allow access to Symbol Configuration .
Function Web Access	Group to allow command on Web server.
Function Monitor	Group to allow monitoring of IEC variables.
Function OPC UA	Group to allow access to OPC UA server.
Function Variable	Group to allow read/write of IEC variables.

Object Names

This table indicates the name and description of the predefined objects:

Object name	Object Description
Device	Object related to the connection of the controller through EcoStruxure Machine Expert.
ExternalCmd	Object related to script command.
FrmUpdate	Object related to the commands Update Boot , Clone and CloneCheck .
FTP	Object related to FTP access (connection, upload and download on ftp server).
Logger	Object related to the message logger.
OPC-UA	Object related to OPC UA server (connection, read and write variables).
PlcLogic	Object related to the application on the controller.
Settings	Object related to the settings of the controller (nodename...).
UserManagement	Object related to User rights Management.
Web	Object related to the access of the Web server.
FileSystem	Object related to the file access (when accessing through the controller Files tab).

Operation Functions

This list indicates the name of the possible predefined operations:

- SD Card command
 - Script Command: Reboot
 - Script Command: SET_NODE_NAME
 - Script Command: FIREWALL_INSTALL
 - Script Command: Delete
 - Script Command: Download
 - Script Command: Upload
 - Script Command: UpdateBoot
 - Script Command: CloneCheck (update controller Users Rights from SD card)
 - Clone operation (clone controller contents to empty SD card)
- FTP server command
 - Connection to FTP server
 - List Directory
 - Change Directory
 - Create Folder
 - Rename Folder
 - Suppress Folder
 - Create File
 - Rename File
 - Suppress File
 - Download File
 - Upload File
- OPC UA server command:
 - Connection to OPC UA server
 - Read Variable
 - Write Variable
- Web server command:
 - Connection to Web server
 - List Variables
 - Read Variable
 - Write Variable
 - Set Time
 - Access to File System
 - Save File
 - Access to logger
 - Change Password
 - Reject/Trust Certificate (Also needs device.settings User Rights Modify)

- EcoStruxure Machine Expert Command
 - Reset Origin Device
 - Login
 - Set Node Name
 - Update Logger
 - Create Application
 - Download application
 - Pass RUN / STOP
 - Reset (Cold / Warm / Origin)
 - Delete Application
 - Create Boot Application
 - Save Retain Variables
 - Restore Retain Variables
 - Add Group
 - Remove Group
 - Add User
 - Remove User
 - Read User Rights
 - Import User Rights
 - Export User Rights

Access Rights

For each **Group** linked with an **Object**, **User Rights** are predefined with specifics **Access Rights**.

This table indicates the **Access Rights**:

Access Rights	Access Rights Description (depends on the Object. See Predefined Access Rights Needed by Object and Associated Operations, page 77).
VIEW	Allow to read only parameters and applications.
MODIFY	Allow to write, modify and download parameters and applications.
ADD_REMOVE	Allow to add and remove files, scripts and folders.
EXECUTE	Allow to execute and start applications and scripts.

Predefined Access Rights for Group Persona

For each **Group**, several **Objects** are predefined with preset **Access Rights**

Group: Administrator	
Object name	Access Rights
Device	VIEW / MODIFY / ADD_REMOVE / EXECUTE
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW
OPC-UA	VIEW / MODIFY
PlcLogic	VIEW / MODIFY / ADD_REMOVE / EXECUTE
Settings	VIEW / MODIFY
UserManagement	VIEW / MODIFY
Web	VIEW / MODIFY / EXECUTE
FileSystem	VIEW / MODIFY / ADD_REMOVE

Group: Designer / Programmer persona	
Object name	Access Rights
Device	VIEW / ADD_REMOVE
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW
OPC-UA	VIEW / MODIFY
PlcLogic	VIEW / MODIFY / ADD_REMOVE / EXECUTE
Settings	VIEW / MODIFY
UserManagement	VIEW
Web	VIEW / MODIFY / EXECUTE
FileSystem	VIEW / MODIFY / ADD_REMOVE

Group: Operator persona	
Object name	Access Rights
Device	VIEW
Logger	VIEW
PlcLogic	VIEW / MODIFY / EXECUTE
Settings	VIEW
UserManagement	VIEW
Web	VIEW / MODIFY / EXECUTE

Group: Designer / Web designer persona	
Object name	Access Rights
Device	VIEW
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW
OPC-UA	VIEW
PlcLogic	VIEW
Settings	VIEW

Group: Designer / Web designer persona	
Object name	Access Rights
UserManagement	VIEW
Web	VIEW / MODIFY / EXECUTE
FileSystem	VIEW / MODIFY / ADD_REMOVE

Group: Communication expert persona	
Object name	Access Rights
Device	VIEW
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW
OPC-UA	VIEW / MODIFY
PlcLogic	VIEW / MODIFY / EXECUTE
Settings	VIEW
UserManagement	VIEW
Web	VIEW / MODIFY / EXECUTE
FileSystem	VIEW / MODIFY / ADD_REMOVE

Group: Maintenance persona	
Object name	Access Rights
Device	VIEW
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW
OPC-UA	VIEW
PlcLogic	VIEW / EXECUTE
Settings	VIEW
UserManagement	VIEW
Web	VIEW / MODIFY / EXECUTE
FileSystem	VIEW / MODIFY / ADD_REMOVE

Predefined Access Rights for Group Function

For each **Group**, several **Objects** are predefined with predefined **Access Rights**

Group: Function External Media ⁽¹⁾	
Object name	Access Rights
ExternalCmd	VIEW / MODIFY / ADD_REMOVE / EXECUTE
FrmUpdate	VIEW / MODIFY / ADD_REMOVE / EXECUTE
(1) Enabling the objects in the group External Media will allow the access rights regardless of the user. That is to say, that the rights governing SD cards are global and are not confined to defined users.	

Group: Function File Access	
Object name	Access Rights
Logger	VIEW
FileSystem	VIEW / MODIFY / ADD_REMOVE

Group: Function FTP Access	
Object name	Access Rights
FTP	VIEW / MODIFY / ADD_REMOVE
Logger	VIEW

Group: Function Symbol Configuration Access	
Object name	Access Rights
Logger	VIEW
OPC-UA	VIEW / MODIFY
PlcLogic	VIEW / MODIFY / ADD_REMOVE / EXECUTE
Web	VIEW / MODIFY / EXECUTE

Group: Function Web Access	
Object name	Access Rights
Logger	VIEW
Web	VIEW / MODIFY / EXECUTE

Group: Function Monitor Access	
Object name	Access Rights
Logger	VIEW
OPC-UA	VIEW
PlcLogic	VIEW
Web	VIEW

Group: Function OPC UA Access	
Object name	Access Rights
Logger	VIEW
OPC-UA	VIEW / MODIFY

Group: Function Variable Access	
Object name	Access Rights
Logger	VIEW
OPC-UA	VIEW
PlcLogic	VIEW / MODIFY / ADD_REMOVE / EXECUTE
Web	VIEW

Predefined Access Rights Needed by Object and Associated Operations

Object Name	Access Rights			
	ADD_REMOVE	MODIFY	VIEW	EXECUTE
Device	Reset origin device	Set node name	Login	–
ExternalCmd	Delete	Download	Upload	Delete Reboot Set Node Name Firewall install Clone Check
FrmUpdate	Updateboot	–	Clone	Clone Check
FTP	–	Create folder Rename Folder Suppress folder Create file Rename File Suppress file Upload file	Connection to FTP Server List directory Change directory Create folder Rename Folder Suppress folder Create file Rename File Suppress file Download file Upload file	–
Logger	–	–	Update logger	–
OPC-UA	–	Write Variable	Connection OPC-UA Read Variable	–
PlcLogic	Create application Download application Delete application Create Boot application	Write Variable	Read Variable Save retain variables	Pass Run / Stop Reset Restore Retains Var
Settings	–	Reject / Trust Certificate Set Node Name	–	–
UserManagement	–	Add Group Remove Group Add User Remove User Edit User Rights Import User Rights Reset Origin Device	Read User Rights Export User Rights	–
Web	–	Set Variables Set Time Save File Change Password	Connection to Web server Monitor Variables Access Files System Change Password	Execute Command
FileSystem	–	–	–	–

Symbol Rights

The Symbol Rights tab (see [Tabs Description](#), page 60) allows you to configure user group access to the symbol sets. It consists in a customizable set of symbols allowing to separate functions and associate them with a user right. If supported by the target device, you can combine different symbol sets from the symbols of the application in the symbol configuration editor. The information about the symbol sets is downloaded to the controller. Then you can define the user group that has access to each symbol set.

Troubleshooting

The only way to gain access to a controller that has user access-rights enabled and for which you do not have the password(s) is by performing an Update Firmware operation. This clearing of User Rights can only be accomplished by using a SD card to update the controller firmware. In addition, you may clear the User Rights in the controller by running a script (refer to [Reset the User Rights to Default](#), page 234). This effectively removes the existing application from the controller memory, but restores the ability to access the Controller.

Embedded Inputs and Outputs Configuration

Configuring the Fast I/Os

Embedded I/Os Configuration

Overview

The embedded I/O function allows configuration of the controller inputs and outputs.

The TM262• controllers provide:

- 4 fast inputs.
- 4 fast outputs.

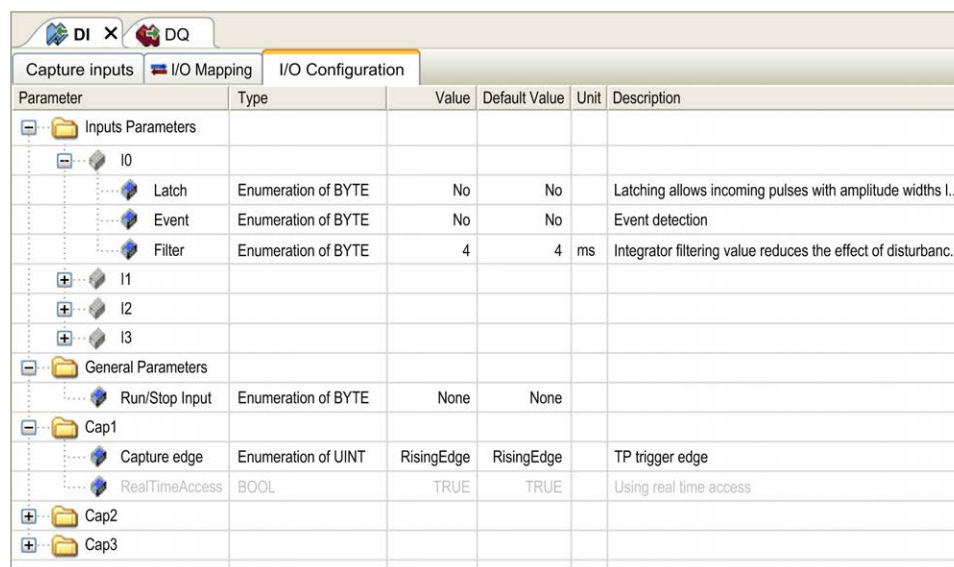
Accessing the I/O Configuration Window

Follow these steps to access the I/O configuration window:

Step	Description
1	Double-click DI (digital inputs) or DQ (digital outputs) in the Devices tree . Refer to Devices tree, page 21.
2	Select the I/O Configuration tab.

Configuration of Digital Inputs

This figure shows the **I/O Configuration** tab for digital inputs:



Parameter	Type	Value	Default Value	Unit	Description
Inputs Parameters					
I0					
Latch	Enumeration of BYTE	No	No		Latching allows incoming pulses with amplitude widths l...
Event	Enumeration of BYTE	No	No		Event detection
Filter	Enumeration of BYTE	4	4	ms	Integrator filtering value reduces the effect of disturbanc...
I1					
I2					
I3					
General Parameters					
Run/Stop Input	Enumeration of BYTE	None	None		
Cap1					
Capture edge	Enumeration of UINT	RisingEdge	RisingEdge		TP trigger edge
RealTimeAccess	BOOL	TRUE	TRUE		Using real time access
Cap2					
Cap3					

NOTE: For more information on the **I/O Mapping** tab, refer to the EcoStruxure Machine Expert Programming Guide (see EcoStruxure Machine Expert, Programming Guide).

Digital Input Configuration Parameters

For each digital input, you can configure the following parameters:

Parameter	Value	Description	Constraint
Filter	0.000 ms	Reduces the effect of noise on a controller input.	Configure Filter to 0.000 if you do not want to filter the signal.
	0.001 ms		
	0.002 ms		
	0.005 ms		
	0.01 ms		
	0.05 ms		
	0.1 ms		
	0.5 ms		
	1 ms		
	4 ms*		
	12 ms		
Latch	No*	Allows incoming pulses with amplitude widths shorter than the controller scan time to be captured and recorded.	Available if Event disabled.
	Yes		
Event	No*	Event detection	Available if Latch disabled. When Both edges is selected, and the input state is TRUE before the controller is powered on, the first falling edge is ignored.
	Rising edge		
	Falling edge		
	Both edges		
Run/Stop Input	None*	The Run/Stop input can be used to run or stop the controller application.	Select one of the inputs to use as the Run/ Stop Input.
	I0...I3		
* Parameter default value			

NOTE: The selection is grey and inactive if the parameter is unavailable.

Run/Stop Input

This table presents the different states:

Input states	Result
State 0	Stops the controller and ignores external Run commands. FSP LED is red ON.
A rising edge	From the STOPPED state, initiate a start-up of an application in RUNNING state, if no conflict with Run/Stop switch position.
State 1	The application can be controlled by: - EcoStruxure Machine Expert (Run/Stop) - A hardware Run/Stop switch - Application (Controller command) - Network command (Run/Stop command) Run/Stop command is available through the Web server command.

NOTE: Run/Stop input is managed even if the option **Update I/O while in stop** is not selected in **Controller Device Editor (PLC settings tab)**, page 62.

Inputs assigned to configured expert functions cannot be configured as Run/Stop inputs.

For further details about controller states and states transitions, refer to [Controller State Diagram](#), page 42.

⚠ WARNING

UNINTENDED MACHINE OR PROCESS START-UP

- Verify the state of security of your machine or process environment before applying power to the Run/Stop input.
- Use the Run/Stop input to help prevent the unintentional start-up from a remote location.

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

Capture Input

Capture Inputs tab allows you to select captures, exclusively for motion applications, and manage them in the **I/O Configuration** tab.

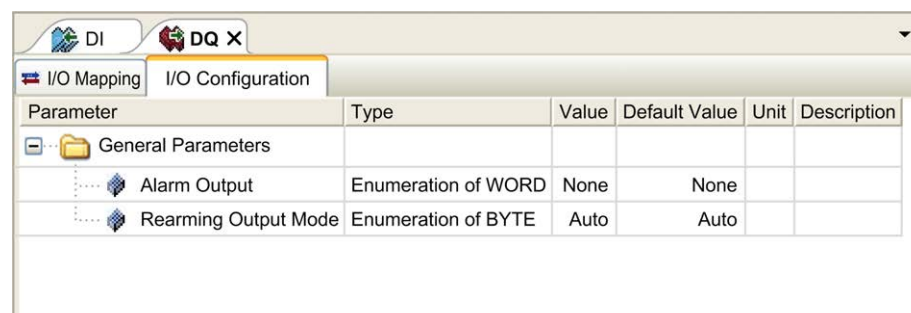
For each capture, you can configure the following parameters:

Parameter	Type	Value	Description	Constraint
Capture edge	UINT Enum	Falling edge Rising edge Both edges	Configure the edge on which the encoder position is captured.	Enable the capture positions in Capture Inputs tab. Do not use with the function blocks from the M262 Encoder Library .
RealTimeAccess	BOOL	TRUE	Using real time access.	Enable the capture positions in Capture Inputs tab. Do not use with the function blocks from the M262 Encoder Library .

For more information on motion applications and function blocks, such as **MC_TouchProbe** and **MC_AbortTrigger**, refer to [M262 Synchronized Motion Control Library Guide](#).

Configuration of Digital Outputs

This figure shows the **I/O Configuration** tab for digital outputs:



NOTE: For more information on the **I/O Mapping** tab, refer to the [EcoStruxure Machine Expert Programming Guide](#).

Digital Output Configuration Parameters

This table presents the function of the different parameters:

Parameter	Function
General Parameters	
Alarm Output	Select an output to be used as alarm output, page 82.
Rearming Output Mode	Select the rearming output mode, page 82.

NOTE: The selection is grey and inactive if the parameter is unavailable.

Alarm Output

This output is set to logical 1 when the controller is in the RUNNING state and the application program is not stopped at a breakpoint.

The alarm output is set to 0 when a task is stopped at a breakpoint to signal that the controller has stopped executing the application and when the controller is stopped.

NOTE: Outputs assigned to configured expert functions cannot be configured as the alarm output.

Rearming Output Mode

Fast outputs of the Modicon M262 Logic/Motion Controller use push/pull technology. In case of detected error (short-circuit or over temperature), the output is put in the default value and the condition is signaled by status bit and PLC_R_IO_STATUS. This is also signaled by %IX1.0.

Two behaviors are possible:

- **Automatic rearming:** as soon as the detected error is corrected, the output is set again according to the current value assigned to it and the diagnostic value is reset.
- **Manual rearming:** when an error is detected, the status is memorized and the output is forced to the default value until user manually clears the status (see I/O mapping channel).

In the case of a short-circuit or current overload, the common group of outputs automatically enters into thermal protection mode (all outputs in the group are set to 0), and are then periodically rearmed (each second) to test the connection state. However, you must be aware of the effect of this rearming on the machine or process being controlled.

⚠ WARNING
UNINTENDED MACHINE START-UP Inhibit the automatic rearming of outputs if this feature is an undesirable behavior for your machine or process. Failure to follow these instructions can result in death, serious injury, or equipment damage.

NOTE: Automatic rearming of outputs can be inhibited through the configuration.

Hardware Encoder Interface

Hardware Encoder Interface

Introduction

The controller has a specific hardware encoder interface that can support:

- Incremental encoder
- SSI absolute encoder

Incremental Mode Principle Description

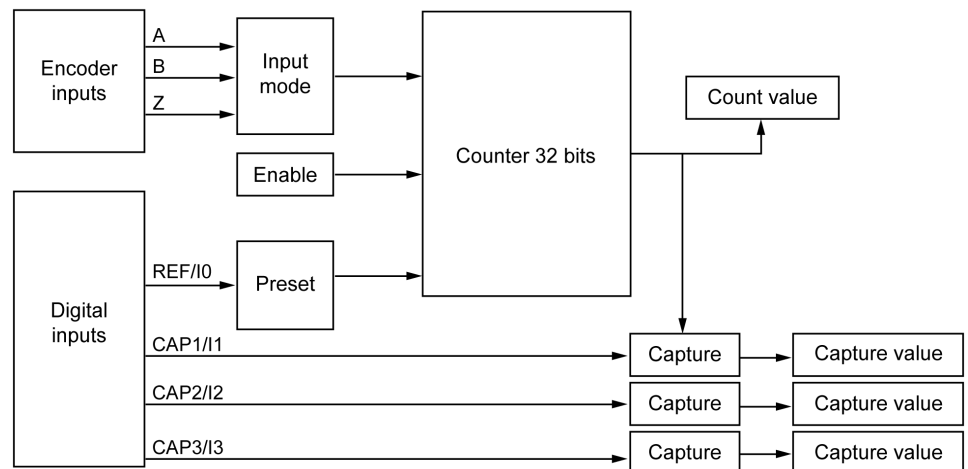
The incremental mode behaves like a standard up/down counter, using pulses and counting these pulses.

Positions must be preset and counting must be initialized to implement and manage the incremental mode.

The counter value can be stored in the capture register by configuring an external event.

Incremental Mode Principle Diagram

The following diagram provides an overview of the encoder in incremental mode:



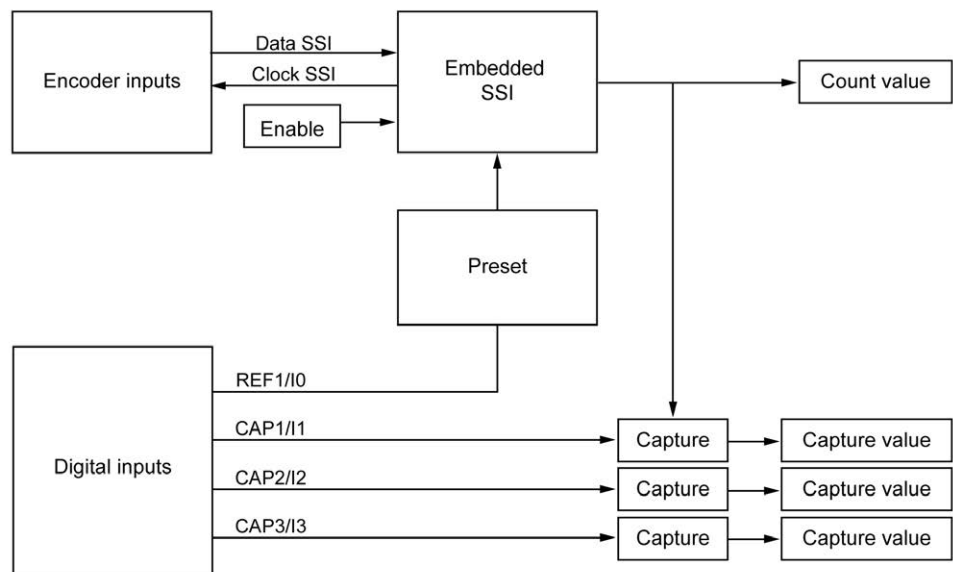
SSI Mode Principle Description

The SSI (Synchronous Serial Interface) mode allows the connection of an absolute encoder.

The position of the absolute encoder is read by an SSI link.

SSI Mode Principle Diagram

The following diagram provides an overview of the encoder in SSI mode:



I/O mapping

This variable is used by the library to identify the encoder, incremental or SSI, to which the function block applies.

Adding an Encoder

Introduction

In order to use the encoder interface, the Modicon M262 Logic/Motion Controller has a specific hardware encoder interface that can support:

- Incremental encoder
- SSI encoder

For more information on usable function blocks, refer to Modicon M262 Logic/Motion Controller - Encoder Library Guide (see Modicon M262 Logic/Motion Controller, Encoder Library Guide).

Adding an Encoder

To add an encoder to your controller, select the encoder in the **Hardware Catalog**. Drag and drop it to the **Devices tree** on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Incremental Encoder Configuration

To configure the incremental encoder, double-click the encoder node in the **Devices tree**.

This table describes the incremental encoder configuration parameters:

Parameter	Type	Value	Default Value	Unit	Description
Power supply					
Voltage selection	BYTE Enum	None 5 V 24 V	None	–	–
Power supply monitor	BYTE Enum	Enabled Disabled	Disabled	–	Enable the power supply monitor
General					
Input Mode	BYTE Enum	Normal Quadrature x 1 Normal Quadrature x 2 Normal Quadrature x 4 Reverse Quadrature x 1 Reverse Quadrature x 2 Reverse Quadrature x 4	Normal Quadrature x 1	–	Select the period measurement interval
Counting Inputs					
A Input					
Filter	BYTE Enum	0.000 0.001 0.002 0.005 0.05 0.01 0.08 0.5 1 4 12	0.002	ms	Set the filtering value to reduce the bounce effect on the input
B Input					
Filter	BYTE Enum	0	0.002	ms	Set the filtering value to reduce the bounce effect on the input
Preset Input					
Z Input					
Filter	BYTE Enum	0.000 0.001 0.002 0.005 0.05 0.01 0.08 0.5 1 4 12	0.002	ms	Set the filtering value to reduce the bounce effect on the input

SSI Encoder Configuration

To configure the **SSI Encoder**, double-click the encoder node in the **Devices tree**.

This table describes the SSI encoder configuration parameters:

Parameter	Type	Value	Default Value	Unit	Description
Power supply					
Voltage selection	BYTE Enum	None 5 V 24 V	None	–	–
Power supply monitor	BYTE Enum	Disabled	Disabled	–	Enable the power supply monitor
Synchronous Serial Interface (SSI)					
Transmission Speed	BYTE Enum	100 250 500	250	KHz	Select the speed of data transmission
Number of bits per frame	USINT (8...64)	8	8	–	Set the number of bits per frame (header + data bits + status + parity)
Number of data bits	USINT (8...32)	8	8	–	Set the number of bits to count turn + bits to count points per turn
Number of data bits / turn	USINT (8...16)	8	8	–	Set the number of data bits to count points per turn
Number of status bits	USINT (0...4)	0	0	–	Set the number of bits to be reserved for the status
Parity	BYTE Enum	None	None	–	Select the parity
Resolution reduction	USINT (0...17)	0	0	–	Set the resolution code
Binary coding	BYTE Enum	Binary	Binary	–	Select the binary coding mode

Motion Functions

You can configure specific elements, exclusively for motion applications. For more information, refer to **Motion Functions** tab, page 86.

Encoder Motion Functions

Introduction

The encoder **Motion Functions** tab allows you to configure specific elements, exclusively for motion applications.

NOTE: These **Motion Functions** must not be used with the **M262Encoder** library when check boxes for **Axis**, **Scaling**, **Filter** and/or **DeadTimeCompensation** are enabled.

Configuring the Motion Functions

This table describes the procedure to configure the motion functions

Step	Action
1	Double-click the encoder node in the Devices tree .
2	Open the Motion Functions tab.
3	Enable check boxes for Axis , Scaling , Filter and/or DeadTimeCompensation . Result: The configuration parameters are displayed in the Incremental Encoder configuration tab or in the SSI Encoder configuration tab.

Incremental/SSI Encoder

This table describes the motion functions configuration parameters of the incremental encoder or of the SSI encoder:

Parameter	Type	Value	Default Value	Description
Scaling				
IncrementResolution	DINT	1...2,147,483,647	131072	IncrementResolution
PositionResolution	LREAL	1.0...1.7976931348623158e+308	360.0	PositionResolution
GearIn	UDINT	1...4,294,967,295	1	GearIn
GearOut	UDINT	1...4,294,967,295	1	GearOut
InvertDirection	BOOL	FALSE	FALSE	Invert movement direction of the axis
Filter				
AverageDuration	UDINT	0...1024	0	Filter duration in Sercos cycles
DeadTimeCompensation				
Delay	LREAL	-100.0...100.0	0	Delay of feedback movement values (position/velocity/acceleration) in milliseconds. This delay will be compensated by the system.

NOTE: The **DeadTimeCompensation delay**, without a **filter**, results in a very high signal deviation of the feedback velocity and can result in unintended behavior of a coupled slave axis.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Use a filter whenever you use a delay for DeadTimeCompensation.

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

Symbol Configuration Editor

Overview

The symbol configuration functionality allows you to configure external access to variables. The symbols and the variables can then be monitored in the Web server and be accessed by external applications, such as Vijeo-Designer or OPC server.

To configure symbols for an application, double-click the **Symbol Configuration** node in the **Tools tree**. The **Symbol configuration** editor view opens.

The editor contains a table. Depending on the set filter, it shows the available variables, or just those already selected for the symbol configuration. For this purpose, the concerned info pop-ups with POU's definition or libraries are listed in the **Symbols** column. You can expand them in order to show the corresponding variables.

NOTE: The number of variables you can configure is not limited.

The restrictions depend on the different monitor platforms:

Platform	Maximum Limit
Web server	16 000 bytes
OPC UA	10 000 variables
HMI	Depends on RAM of each model

Elements of the Toolbar

Element	Description	
View button	The View button allows you to set the following filters to reduce the number of displayed variables:	
	Unconfigured from Project	Even variables not yet added to the symbol configuration, but available for this purpose in the project, are displayed.
	Unconfigured from Libraries	Also variables from libraries, not yet added to the symbol configuration, but available for this purpose in the project, are displayed.
	Symbols exported via attribute	This setting is effective only when the unconfigured variables are displayed (see the two filters described above). It has the effect that also those variables are listed, which are already selected for getting symbols by {attribute 'symbol' := 'read'} within their declaration. Such symbols are displayed grayed. The Attribute column shows which access right is set for the variable by the pragma. Refer to the following description of the Access Rights column (see EcoStruxure Machine Expert, Programming Guide).
Build button	The Build button allows you to build the project. The build run refreshes the variables view in the Symbol Configuration editor.	

Element	Description	
Settings button	The Settings button allows you to activate the following options:	
	Support OPC UA features	This function is not supported.
	Include comments in XML	This has the effect that comments assigned to variables are also exported to the symbol file.
	Include node flags in XML	This has the effect that the flags that contain the namespace are also exported to the symbol file. They provide additional information on the origin of the node in the namespace when OPC UA is active.
	Configure comments and attributes...	Opens the Comments and Attributes dialog box that allows you to configure the contents of the symbol configuration and the XML file.
	Configure synchronisation with IEC tasks...	Opens the Options tab of the Properties dialog box of the selected controller. Refer to the description of the Access variables in sync with IEC tasks option in the EcoStruxure Machine Expert Menu Commands Online Help (see EcoStruxure Machine Expert, Menu Commands, Online Help). NOTE: Do not activate the option Configure synchronisation with IEC tasks... for motion and real time-critical application because the delayed start of IEC tasks leads to a higher jitter. For further information, refer to the paragraph <i>Additional Information on the Option Configure synchronisation with IEC tasks...</i> (see EcoStruxure Machine Expert, Programming Guide).
	Compatibility Layout	Select this option to calculate the data output in the same way as in SoMachine / SoMachine Motion versions prior to V4.3. Do not use this layout together with exported STRUCTs that use the attributes <i>pack_mode</i> or <i>relative_offset</i> . The data layout created for the client is adapted as far as possible to the layout created by the compiler.
Optimized Layout	Select this option to calculate the data output in an optimized form, independent of the internal compiler layout. The optimization impacts only variables of a structured type and function blocks. No gaps with padding bytes are generated for members that are not published, for example, because they are deactivated in the Symbol Configuration . For internal members, for example, function blocks implementing interfaces, no gaps are created either. This option is by default selected for projects with EcoStruxure Machine Expert. The setting is preserved after a Project update .	
Tools button	Save XML Scheme File...	Opens the dialog box for saving a file in the file system. Allows you to create an XSD (XML Schema Definition) format of the symbol file, for use in external programs.

Description of the Table

Columns of the **Symbol configuration** table:

Column	Description
Symbols	The column shows a list of POUs. You can select variables to be exported. If you select variables of a structured data type, all members of the structure will be exported. You may also select only particular member variables in the dialog box Symbol Configuration for Data Type. Click the browse ... button in the Members column to open this dialog box. For further information, refer to the description of the Members column.
Access Rights	To modify the access rights for a selected item, click the Access Rights column. Each mouse-click switches the symbol within the following definitions: : read and write : write-only : read-only - none
Maximal	Shows the maximum access rights.
Type	Shows the data type of the variable. Variables of an alias data type are displayed as shown in the following example of a variable: <pre>myVar : MY_INT, , where MY_INT is an alias declared as: TYPE MY_INT : INT; END_TYPE.</pre> In this case, the Type column shows MY_INT : INT.
Members	Click the ... button in the Members column to open the Symbol Configuration for Data Type dialog box. It allows you to select only particular member variables. In case of nested types, this dialog box again provides a button to open another Symbol Configuration for Data Type dialog box. This selection applies to all instances of this data type for which symbols are exported. If not all members of a structured type are exported, then an asterisk (*) is displayed in the check boxes of the members to indicate that all exportable members of that type are exported.
Comment	Shows any comments which have been added in the declaration of the variable.

NOTE: With the POU property **Link Always**, an uncompiled POU can be forced to be downloaded to the controller. If this property is set in the **Build** tab of the **Properties** dialog box of the selected POU, then all variables declared in this POU are available, even if those variables are not referenced by other code. Alternatively, you can use the pragma {attribute linkalways} (see EcoStruxure Machine Expert, Programming Guide) to make not compiled variables available in the symbol configuration.

Variables which are configured to be exported but which are not valid in the application, for example because their declaration has been removed, are shown in red. This also applies to the concerned POU or library name.

By default, a symbol file is created with a code generation run. This file is transferred to the device with the next download. If you want to create the file without performing a download, use the command **Generate code**, by default available in the **Build** menu.

NOTE: Variables of a global variable list (GVL) will only be available in the symbol configuration if at least one of them is used in the programming code.

Comments and attributes Dialog Box

The **Comments and attributes** dialog box opens upon clicking **Settings > Configure comments and attributes**. It contains the following elements:

Element	Description
Symbol table contents	
Enable extended OPC UA information	This function is not supported.
Include comments	
Include attributes	
Also include comments and attributes for type nodes	
XML symbol file contents	
Include namespace node flags	Namespace node flags provide additional information about the origin of a node in the namespace. Node flags are available in the symbol table when OPC UA is activated. Deselect this option to prevent namespace node flags from being inserted in the XML file if your parser cannot process them.
Include comments	Select this option to save comments in the XML file. In SoMachine / SoMachine Motion versions prior to V4.4, this includes the setting Prefer docu comments .
Include attributes	Select this option to save attributes in the XML file.
Also include comments and attributes for type nodes	This option is only available if the option Include comments or the option Include attributes is activated. If this option is selected, the information for type nodes is also included (user-defined types, such as STRUCT and ENUM elements). If this option is not selected, comments and attributes are only available for directly exported variables.
Select comments	
These parameters are only available if one of the Include comments options is activated.	
Include docu comments: ///They start with triple slash and are usually /// formatted in ReST (library documentation)	Select the options to determine the kind of comments that are saved in the symbol configuration.
Include normal comments: (*IEC/ Pascal style comments *) // C++-Style comments with double-slash	
Always include both types of comments	
Prefer docu comments, fallback to normal ones	
Prefer normal comments, fallback to docu comments	
Filter Attributes (case insensitive)	
These parameters are only available if one of the Include attributes options is activated.	
Include all attributes (“foo”, “bar”, “foo.bar”)	Select the options to determine the attributes that are saved in the symbol configuration.
Match simple identifiers (“foo”, “bar”)	
Include attributes starting with:	
Filter Attributes with regular expression	

Additional Information on Configure synchronisation with IEC tasks...

To achieve synchronous, consistent access, the runtime system postpones the processing of read or write request of the symbolic client until no IEC task is executed. As soon as this gap is found, the restart of IEC tasks is postponed until the requested values have been copied to the variables list.

This option is useful for permanently running systems without production clocking, for example, if process values are to be written cyclically in fixed time intervals (such as 60 s).

NOTE: Do not activate the option **Configure synchronisation with IEC tasks...** for motion and real time-critical application because the delayed start of IEC tasks leads to a higher jitter.

If you intend to use the **Configure synchronisation with IEC tasks...** option, consider the following points when you define the variable lists that are read and written:

- Configure synchronous and consistent access only for those variables for which it is necessary.
- Create separate lists for consistent variables and for variables that may be inconsistent.
- Create several small lists containing consistent variables instead of one large list.
- Define the time intervals for cyclically reading values as large as possible.

The option **Configure synchronisation with IEC tasks...** is available at two different locations in EcoStruxure Machine Expert:

- In the **Symbol configuration** editor, as an option of the **Settings** button. (If a symbol configuration is available in the application.)
- In the **Options** tab of the **Properties** dialog box of the selected controller.

NOTE: In order for the setting to take effect, perform a **Download** or **Online Change** of the applications on the controller and update the boot applications.

Controller Cybersecurity

Introduction

To help keep your Schneider Electric products secure and protected, implement the cybersecurity best practices as indicated in the [Cybersecurity Best Practices and Cybersecurity Guidelines for EcoStruxure Machine Expert, Modicon and PacDrive Controllers and Associated Equipment](#) document provided on the Schneider Electric website.

Certificate Management

By default, the following certificates are displayed in the controller Web server in Maintenance: Certificates Submenu, page 139:

- TM262-XX-OPCUA is used for OPC UA
- TM262-XX is used for HTTP/FTP/WebVisualisation
- Nodename is used for communication with EcoStruxure Machine Expert

Security Settings Configuration with Cybersecurity Admin Expert Software

Introduction

CAE (Cybersecurity Admin Expert) is a software-based tool for building and managing the security configuration and policy of the Operational Technology (OT) within the communication network of control systems. It allows centralized administration of user accounts, roles and permissions for devices such as: network devices (switches, firewalls), PCs and IED/Protection relays. CAE is used for multiple purposes:

- Creating a cybersecurity and security policy
- Configuring the security of devices
- Managing the system definition
- Retrieving security logs of a whole substation, plant or industrial environment

CAE is a Schneider Electric software you can download from <https://www.se.com/ww/en/all-products>.

Prior to apply any modification to the CAE settings, refer to the [Cybersecurity Admin Expert User Manual](#).

The M262 Logic/Motion Controller device model encompasses two features:

- Role-Based Access Control (RBAC)
- Device Specific Settings (DSS)

Role-Based Access Control (RBAC)

This feature consists in controlling access to a system's resources based on the roles and permissions of the users. The list of permission covers common use cases, as displayed in the following graphic:

The screenshot displays the 'ROLES' configuration page for the 'M262' role. The interface includes a sidebar with a list of roles (C264, C5, CAE, Control Expert, EasergyP, EasergyT300, EcoSUI, EnerlinX, GTWPACIS, iFLS, M262) and a main panel for editing the selected role. The 'Model Details' section shows the role name as 'M262', with empty fields for 'Firmware' and 'Description'. The 'Permissions' section lists various permissions with toggle switches: Application Management (on), Application View (on), FTP Read Access (on), FTP Write Access (on), OPC UA Server (on), User Management (off), Web Read Access (on), and Web Write Access (on). An 'Add permission' button is also present.

The following table describes each permission, the concerned M262 objects and the corresponding access rights:

CAE Permissions	M262 Object Name	M262 Access Rights
Application Management	Device	USERDB_RIGHT_ALL
	Device.PlcLogic	USERDB_RIGHT_ALL
	Device.PlcLogic.Application	USERDB_RIGHT_ALL
	Device.Settings	USERDB_RIGHT_ALL
	Device.OPC	USERDB_RIGHT_ALL
	Device.ExternalCmd	USERDB_RIGHT_ALL
	"/"	USERDB_RIGHT_ALL
Application View	Device	USERDB_RIGHT_VIEW
	Device.PlcLogic	USERDB_RIGHT_VIEW
	Device.PlcLogic.Application	USERDB_RIGHT_VIEW
	Device.Settings	USERDB_RIGHT_VIEW
	Device.OPC	USERDB_RIGHT_VIEW
	Device.ExternalCmd	USERDB_RIGHT_VIEW
	"/"	USERDB_RIGHT_VIEW
FTP Read Access	Device.FTP	USERDB_RIGHT_VIEW
FTP Write Access	Device.FTP	USERDB_RIGHT_ALL
User Management	Device.UserManagement	USERDB_RIGHT_ALL
Web Read Access	Device.WEB	USERDB_RIGHT_VIEW
Web Write Access	Device.WEB	USERDB_RIGHT_ALL

Roles and Rights

Up to 20 users are supported by M262 controllers. Each user may have several roles. The following table describes the default rights for each user role:

Role	Access Rights
ENGINEER	Application Management Application View FTP Read Access FTP Write Access OPC UA Server Web Read Access Web Write Access
INSTALLER	OPC UA Server Web Read Access
OPERATOR	Application Management FTP Read Access FTP Write Access OPC UA Server Web Read Access
SECADM	User Management
VIEWER	Application View FTP Read Access Web Read Access

Device Specific Settings (DSS)

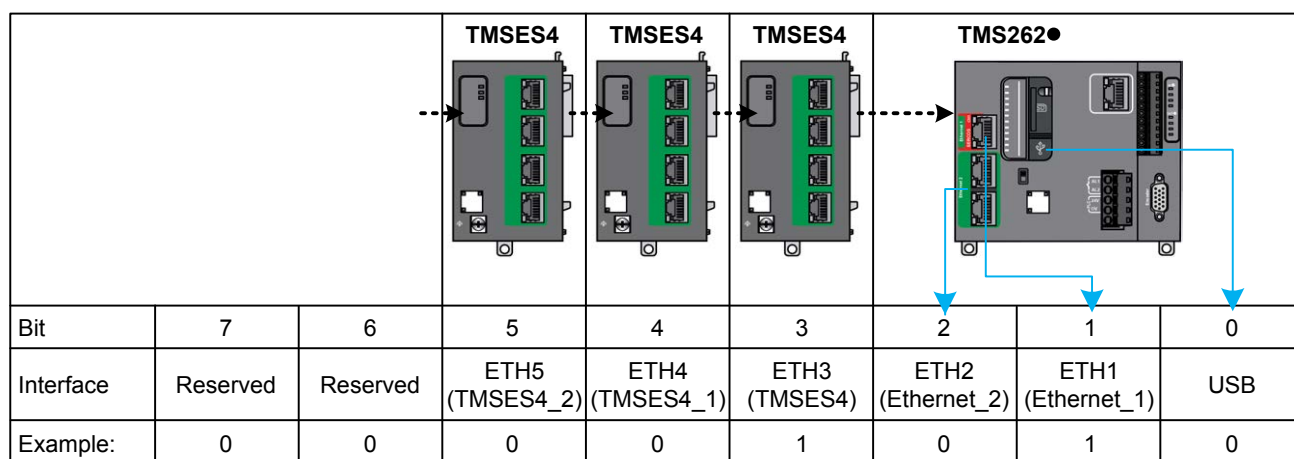
This parameter allows you to configure the device specific settings. This table describes the M262 Logic/Motion Controller **MODELS > Specific Settings** parameters:

Key	Type	Default Value	Description
Discovery Protocols	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol DPWS and NetManage in each communication port: - TCP port: 5357 - UDP ports: 3702, 5353, 27126, 27127 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾. NOTE: Disabling these protocols prevents the device to be discovered by the software CAE.
EtherNet/IP	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable EtherNet/IP in each communication port: - TCP port: 44818 - UDP ports: 2222, 44818 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾.
FTP Server	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP ports: 20, 21 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾.
Machine Expert Protocol	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - UDP ports: 1740 to 1743 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾.

Key	Type	Default Value	Description
Modbus Server	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP port: 502 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .
OPC UA Server	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP port: 4840 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .
Remote Connection (Fast TCP)	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP port: 11740 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .
Clone Application Enabled	BOOL	TRUE	Enable or disable the cloning of the device via the SD card.
SD Card Script Execution Enable	BOOL	TRUE	Enable or disable the execution of scripts via the SD card. Refer to Operation Functions, page 72.
Secure Web Server (HTTPS)	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP ports: 80, 443 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ . NOTE: Disabling this protocol prevents the device to receive data from the software CAE.
SNMP Agent	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - UDP ports: 161, 162 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .

Key	Type	Default Value	Description
TFTP Server	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - UDP port: 69 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .
WebVisualisation Protocol	INTEGER	Decimal: 255 Binary: 1111 1111	Allows you to enable or disable the protocol in each communication port: - TCP port: 8080, 8089 Bit value: 0 = Disable, 1 = Enable - Bit 0: USB - Bit 1: ETH1 - Bit 2: ETH2 - Bit 3 to 5: TMSSES4 1 to 3 - Bit 6 to 7: reserved For further information, refer to the example below ⁽¹⁾ .

(1)



In this example, the chosen protocol is allowed on ETH1 and the first TMSSES4. It is blocked on the other interfaces. The related binary value 00001010 corresponds to 10 in decimal, so the related parameter value must be set to 10.

Operating Modes

The control of the device security settings via the CAE is enabled by default on the M262 Logic/Motion Controller. To disable CAE, refer to *Post Configuration Presentation*, page 215.

Once the connection between the CAE and the controller is accepted, the CAE is allowed to send the RBAC or the DSS configurations. After receiving a valid RBAC configuration, the existing users and groups are deleted, and new groups and users are created based on the RBAC configuration.

If you have used CAE to manage security, and then modify the security settings with EcoStruxure Machine Expert, groups and/or user accounts may be deleted, and inconsistencies may occur.

⚠ WARNING

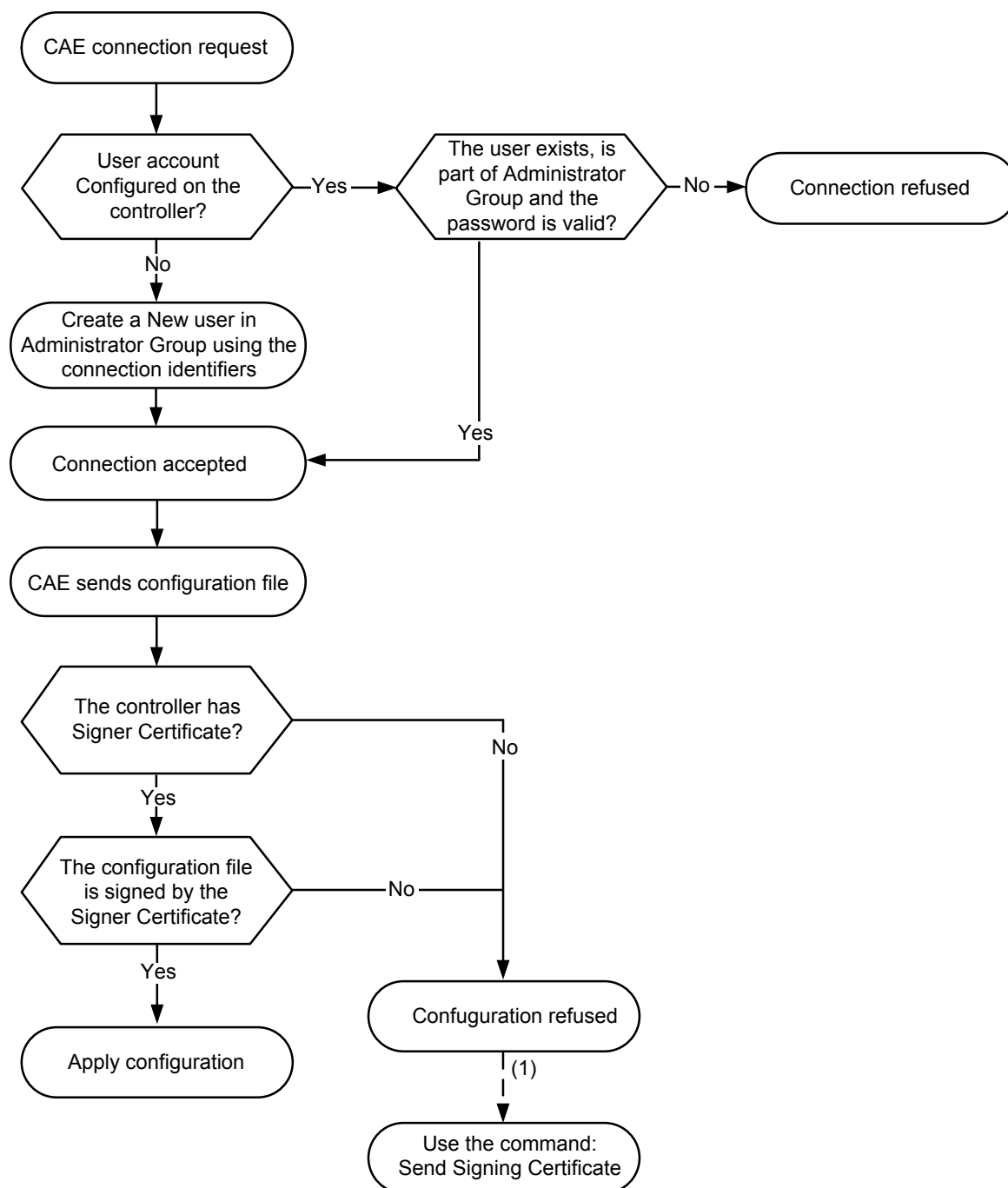
LOSS OF DATA

Do not create the user accounts and groups with EcoStruxure Machine Expert if the security settings are managed by the Cybersecurity Admin Expert (CAE) software.

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

Any DSS configuration received from the CAE is applied immediately.

The following diagram describes the connection between the CAE tool and the controller:



(1) In the case of the first CAE certificate has not been sent yet, use the command **Send Signing Certificate**. Subsequent operation will be proceed automatically.

CAE Options Supported by the M262 Logic/Motion Controller

This table describes the different CAE commands you can use with the M262 Logic/Motion Controller:

Commands	Description
 Discover device(s) over the network	Displays the controller in the list of the discovered devices.
Send Signing Certificate	Saves the CAE signing certificate inside the file system. Any configuration received is rejected if the signing certificate is not provisioned. Only one signing certificate is supported.
Send Security Configuration	Sends the RBAC and DSS configuration files and apply the configuration.
Log on	Connects manually the controller when automatic connection did not succeed.
Send DSS	Sends the DSS and to apply the configuration, once it has been validated by the CAE tool, using the CAE signing certificate.
Reset	Resets the security configuration to default. RBAC (users, roles, permissions) and DSS are reset.
Locate	Locates the device by flashing its LED.
Certificate Management > Whitelist	Adds or removes certificate(s) in the whitelist (allowlist).
Certificate Management > Signers	Adds or removes the CAE certificate that can be used to verify the configuration signature. Only one Signer certificate is supported.
Certificate Management > Trusted Chain Certificates	Add a root or intermediate certificate to the controller trusted list. Manages root CA and intermediate CA certificates, so that the controller can verify the certificates chain of trust.
PKI Management > Get CSR	Allows the controller to generate and send the CSR (certificate sign request) for OPC UA certificate.
PKI Management > Send signed device certificate	Allows you to replace the self signed certificate by the certificate signed by the Certificate Authority (CA) and provided to CAE. The certificate requires a Reset Cold , a Reset Warm or a Reboot of the application to apply.

If a command is not active (in grey in the software), refer to the [Cybersecurity Admin Expert User Manual](#).

This table describes the Public Key Infrastructure (PKI) shared between the M262 Logic/Motion Controller and the CAE. It provides the folder list and their usage.

M262 File System Folders	Description
/usr/pki/cae/castore	Stores working certificate received from CAE.
/usr/pki/cae/csr	Stores the signed certificate request.

Expansion Modules Configuration

Overview

This chapter describes how to configure the TMS and TM3 expansion modules for the Modicon M262 Logic/Motion Controller.

TM3 I/O Configuration General Description

Introduction

In your project, you can add I/O expansion modules to your M262 Logic/Motion Controller to increase the number of digital and analog inputs and outputs over those native to the controller (embedded I/O).

You can add either TM3 I/O expansion modules to the controller, and further expand the number of I/O via TM3 transmitter and receiver modules to create remote I/O configurations. Special rules apply in all cases when creating local and remote I/O expansions (refer to Maximum Hardware Configuration (see Modicon M262 Logic/Motion Controller, Hardware Guide)).

The I/O expansion bus of the M262 Logic/Motion Controller is created when you assemble the I/O expansion modules to the controller.

I/O Expansion Bus Errors

If the controller cannot communicate with one or more I/O expansion modules contained in the program configuration, and those modules are not configured as optional modules (refer to [Optional I/O Expansion Modules](#), page 109), the controller detects it as an I/O expansion bus error. The unsuccessful communication may be detected during the startup of the controller or during runtime, and there may be any number of causes. Causes of communication exceptions on the I/O expansion bus include, among other things, disconnection of or physically missing I/O modules, electromagnetic radiation beyond published environmental specifications, or otherwise inoperative modules.

NOTE: In fallback mode, the TM3 bus waits for controller communications for approximately 200 milliseconds before applying fallback values unless the controller sends out a bus reset, setting the output values to their initialization values after two consecutive bus task cycles. If the two bus task cycles exceeds the TM3 bus timeout, output modules first apply their fallback values and then apply initialization values when the bus reset is sent.

If an I/O expansion bus error is detected:

- The system status LED **I/O** of the controller is illuminated in red indicating an I/O error.
- When EcoStruxure Machine Expert is in online mode, a red triangle appears next to the TM3 expansion module or modules in error and next to the **IO_Bus** node on the **Devices tree** window.

The following diagnostic information is also available:

- Bit 0 and bit 1 of the `PLC_R.i_lwSystemFault_1` system variable are set to 0.
- The `PLC_R.i_wIOStatus1` and `PLC_R.i_wIOStatus2` system variables are set to `PLC_R_IO_BUS_ERROR`.
- The `TM3_MODULE_R[i].i_wModuleState` system variable, where `[i]` identifies the TM3 expansion module in error, is set to `TM3_BUS_ERROR`.
- The `TM3_GetModuleBusStatus` function block returns the `TM3_ERR_BUS` error code (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

Refer to PLC_R (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide) and TM3_MODULE_R structures for details on system variables.

Active I/O Expansion Bus Error Handling

This figure shows the select mode tab:

Parameter	Type	Value	Default Value	Unit	Description
Optional module	Enumeration of BYTE	No	No		
Functional Mode	Enumeration of BYTE	1	1		1 = Normal Mode 2 = Fallback Mode (TM3 DIO module with SV >= 2.0)

In the TM3 configuration, you can use the **Normal Mode (1)** or the **Fallback Mode (2)**.

The *TM3_BUS_W.q_wIOBusErrPassiv* system variable is set to *ERR_ACTIVE* by default to specify the use of active I/O error handling. The application can set this bit to *ERR_PASSIVE* to use passive I/O error handling instead.

By default, when the controller detects a TM3 module in bus communication error, it sets the bus to a "bus off" condition whereby the TM3 expansion module outputs image value are set to 0 or in Fallback value depending the mode used, and the inputs image value are set to 0. A TM3 expansion module is considered to be in bus communication error when an I/O exchange with the expansion module has been unsuccessful for at least two consecutive bus task cycles. When a bus communication error occurs, the *TM3_MODULE_R[i].i_wModuleState* system variable, where *[i]* is the expansion module number in error, is set to *TM3_BUS_ERROR*. The other bits are set to *TM3_OK*.

NOTE: In **Fallback Mode**, TM3 I/O Bus will wait for controller communications for approximately 200 milliseconds before applying fallback values unless the controller send out a bus reset, setting all output values to their initialization value after two consecutive bus task cycles. If the two bus task cycles exceeds the TM3 I/O Bus timeout, outputs modules will first apply their fallback values and then apply initialization values when the bus reset is sent.

Normal I/O expansion bus operation can only be restored after eliminating the source of the error and performing one of the following:

- Power cycle
- New application download
- Restarting the I/O Bus by setting the *TM3_BUS_W.q_wIOBusRestart* system variable to 1. The bus is restarted only if no expansion modules are in error (*TM3_MODULE_R[i].i_wModuleState = TM3_BUS_ERROR*). Refer to Restarting the I/O Expansion Bus, page 105.
- Issuing a **Reset Warm** or **Reset Cold** command with EcoStruxure Machine Expert, page 50.

This table describes the behavior of the modules connected to the controller, or to the Receiver module, according to their state and type:

Module Type	Modules Without Fallback Management	Modules With Fallback Management
Reset Cold	Default value ⁽¹⁾	
Reset Warm	Default value ⁽¹⁾	
Reset Origin (Empty controller)	Initialization value ⁽²⁾	
STOP	Default value ⁽¹⁾	
Detected communication error on TM3 bus	Initialization value ⁽²⁾	
Controller in HALT state	Default value ⁽¹⁾	
TM3 Transmitter-Receiver cable gets cut	Initialization value ⁽²⁾	
(1): Value set in EcoStruxure Machine Expert configuration screen.		
(2): Module I/O value state after receiving a reset command from the bus.		

Passive I/O Expansion Bus Handling

This figure shows the select mode tab:

Parameter	Type	Value	Default Value	Unit	Description
Optional module	Enumeration of BYTE	No	No		
Functional Mode	Enumeration of BYTE	1	1		1 = Normal Mode 2 = Fallback Mode (TM3 DIO module with SV > = 2.0)

In the TM3 configuration, you can use the **Normal Mode (1)** or the **Fallback Mode (2)**.

In **Normal Mode (1)**, the application can set the system variable *TM3_BUS_W.q_wIOBusErrPassiv* to *ERR_PASSIVE* to use passive I/O error handling. This error handling is provided to afford compatibility with previous firmware versions.

When passive I/O error handling is in use, the controller attempts to continue data bus exchanges with the modules during bus communication errors. While the expansion bus error persists, the controller attempts to re-establish communication on the bus with incommunicative modules, depending on the type of I/O expansion module:

- For TM3 I/O expansion modules, the value of the I/O channels is maintained (**Keep current values**) for approximately 10 seconds while the controller attempts to re-establish communication. If the controller cannot re-establish communications within that time, the affected TM3 I/O expansion outputs are set to 0.

In **Fallback Mode (2)**, the application can set the system variable *TM3_BUS_W.q_wIOBusErrPassiv* to *ERR_PASSIVE* to use passive I/O error handling. This error handling is provided to afford compatibility with previous firmware versions.

When passive I/O error handling is in use, the controller attempts to continue data bus exchanges with the modules during bus communication errors. While the expansion bus error persists, the controller attempts to re-establish communication on the bus with incommunicative modules, depending on the type of I/O expansion module:

- For TM3 I/O expansion modules, the value of the I/O channels is maintained (**Keep current values**) for approximately 200 milliseconds while the controller attempts to re-establish communication. If the controller cannot re-establish communications within that time, the affected TM3 I/O expansion outputs are set to fallback value.

In either case, the controller continues to solve logic and, if your controller is so equipped, the embedded I/O continues to be managed by the application

(“managed by application program, page 48”) while it attempts to re-establish communication with the incommunicative I/O expansion modules. If the communication is successful, the I/O expansion modules resume to be managed by the application. If communication with the I/O expansion modules is unsuccessful, you must resolve the reason for the unsuccessful communication, and then cycle power on the controller system, or issue a **Reset Warm** or **Reset Cold** command with EcoStruxure Machine Expert, page 50.

The value of the incommunicative I/O expansion modules input image is maintained and the output image value is set by the application.

Further, if the incommunicative I/O module(s) disturb the communication with unaffected modules, the unaffected modules are also considered to be in error and the *TM3_MODULE_R[i].i_wModuleState* system variable (where *[i]* is the expansion module number) is set to *TM3_BUS_ERROR*. However, with the ongoing data exchanges that characterize the Passive I/O Expansion Bus Error Handling, the unaffected modules apply the data sent, and do not apply the fallback values as for the incommunicative module.

Therefore, you must monitor within your application the state of the bus and the error state of the module(s) on the bus, and take the appropriate action necessary given your particular application.

Refer to *Controller States Description*, page 45 for more information on the actions taken upon startup of the controller when an I/O expansion bus error is detected.

This table describes the behavior of the modules connected to the controller, or to the Receiver module, according to their state and type:

Module Type	Modules Without Fallback Management		Modules With Fallback Management	
	Modules Connected to Controller	Modules Connected to Receiver module	Modules Connected to Controller	Modules Connected to Receiver module
Reset Cold	Default value ⁽¹⁾		Default value ⁽¹⁾	
Reset Warm	Default value ⁽¹⁾		Default value ⁽¹⁾	
Reset Origin (Empty controller)	Initialization value ⁽²⁾		Initialization value ⁽²⁾	
STOP	Default value ⁽¹⁾		Default value ⁽¹⁾	
Detected communication error on TM3 bus	Maintain during 10 seconds then initialization value ⁽²⁾		Configured fallback value	
Controller in HALT state	Default value ⁽¹⁾		Default value ⁽¹⁾	
TM3 Transmitter-Receiver cable gets cut	Value controlled by the application	Initialization value ⁽²⁾	Value controlled by the application	Initialization value ⁽²⁾
(1): Value set in EcoStruxure Machine Expert configuration screen.				
(2): Module I/O value state after receiving a reset command from the bus.				

Restarting the I/O Expansion Bus

When active I/O error handling is being applied, that is, embedded and TM3 outputs set to 0 or fallback value when a bus communication error is detected, the application can request a restart of the I/O expansion bus while the controller is still running (without the need for a Cold Start, Warm Start, power cycle, or application download).

The *TM3_BUS_W.q_wIoBusRestart* system variable is available to request restarts of the I/O expansion bus. The default value of this bit is 0. Provided at least one TM3 expansion module is in error (*TM3_MODULE_R[i].i_wModuleState* set to *TM3_BUS_ERROR*), the application can set *TM3_BUS_W.q_wIoBusRestart* to 1 to request a restart of the I/O expansion bus. On detection of a rising edge of this bit, the controller reconfigures and restarts the I/O expansion bus if all of the following conditions are met:

- The *TM3_BUS_W.q_wIoBusErrPassiv* system variable is set to *ERR_ACTIVE* (that is, I/O expansion bus activity is stopped)

- Bit 0 and bit 1 of the *PLC_R.i_lwSystemFault_1* system variable are set to 0 (I/O expansion bus is in error)
- The *TM3_MODULE_R[i].i_wModuleState* system variable is set to *TM3_BUS_ERROR* (at least one expansion module is in bus communication error)

If the *TM3_BUS_W.q_wIoBusRestart* system variable is set to 1 and any of the above conditions is not met, the controller takes no action.

Match Software and Hardware Configuration

The I/O that may be embedded in your controller is independent of the I/O that you may have added in the form of I/O expansion. It is important that the logical I/O configuration within your program matches the physical I/O configuration of your installation. If you add or remove any physical I/O to or from the I/O expansion bus or, depending on the controller reference, to or from the controller (in the form of cartridges), then you must update your application configuration. This is also true for any field bus devices you may have in your installation. Otherwise, there is the potential that the expansion bus or field bus no longer function while the embedded I/O that may be present in your controller continues to operate.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update the configuration of your program each time you add or delete any type of I/O expansions on your I/O bus, or you add or delete any devices on your field bus.

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

Presentation of the Optional Feature for I/O Expansion Modules

I/O expansion modules can be marked as optional in the configuration. The **Optional module** feature provides a more flexible configuration by the acceptance of the definition of modules that are not physically attached to the controller. Therefore, a single application can support multiple physical configurations of I/O expansion modules, allowing a greater degree of scalability without the necessity of maintaining multiple application files for the same application.

You must be fully aware of the implications and impacts of marking I/O modules as optional in your application, both when those modules are physically absent and present when running your machine or process. Be sure to include this feature in your risk analysis.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Include in your risk analysis each of the variations of I/O configurations that can be realized marking I/O expansion modules as optional, and in particular the establishment of TM3 Safety modules (TM3S...) as optional I/O modules, and make a determination whether it is acceptable as it relates to your application.

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

NOTE: For more details about this feature, refer to *Optional I/O Expansion Modules*, page 109.

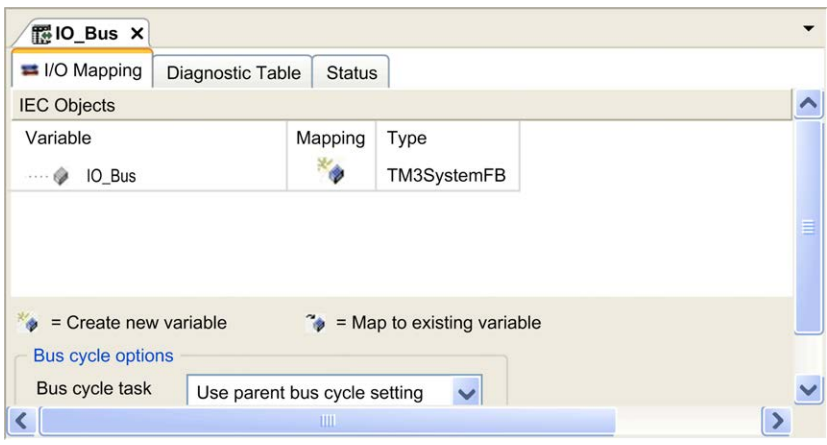
TM3 I/O Bus Configuration

Overview

TM3 I/O bus configuration enables you to select the task that drives TM3 physical exchanges. It can also override the configuration defined in the **PLC settings**, page 62 bus cycle task.

Configuring the I/O Bus

Follow these steps to configure the TM3 I/O bus:

Step	Action
1	In the Devices tree, double-click IO_Bus. Result: The IO_Bus editor tab displays: 
2	Set the Bus cycle task from the list to either of the following: • Use parent bus cycle setting (default) Sets the task for bus exchange as defined in the PLC settings. • MAST Sets the Master task for bus exchange irrespective of the task defined in the PLC settings.

TMS Expansion Module Configuration

Introduction

The Modicon M262 Logic/Motion Controller supports the TMS communication expansion modules.

TMS expansion modules connect to the left side of the controller and are dedicated to Ethernet and CANopen high speed communication. You can configure your TMS expansion modules in the EcoStruxure Machine Expert **Devices Tree**.

NOTE: The TMSES4 expansion module is not a standalone Ethernet switch.

For further information about the TMS expansion modules configuration, refer to the Modicon TMS Expansion Modules Programming Guide.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Adding an Expansion Module

To add an expansion module to your controller, select the expansion module in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Compatibility of the TMS Communication Expansion Modules

You can connect:

- 1 TMSCO1 for TM262L01MESE8T and TM262M05MESS8T
- 3 TMSES4 or 2 TMSES4 and 1 TMSCO1 for the other references

The TMSCO1 must be the leftmost module connected to the controller.

TM3 Expansion Module Configuration

Introduction

The Modicon M262 Logic/Motion Controller supports the following expansion modules:

- TM3 expansion modules:
 - Digital I/O modules
 - Analog I/O modules
 - Expert I/O modules
 - Safety modules
 - Transmitter and receiver modules

For further information about the TM3 expansion modules configuration, refer to the TM3 Expansion Modules Configuration Programming Guide.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Adding an Expansion Module

To add an expansion module to your controller, select the expansion module in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Optional I/O Expansion Modules

Presentation

I/O expansion modules can be marked as optional in the configuration. The **Optional module** feature provides a more flexible configuration by the acceptance of the definition of modules that are not physically attached to the controller. Therefore, a single application can support multiple physical configurations of I/O expansion modules, allowing a greater degree of scalability without the necessity of maintaining multiple application files for the same application.

Without the **Optional module** feature, when the controller starts up the I/O expansion bus (following a power cycle, application download or initialization command), it compares the configuration defined in the application with the physical I/O modules attached to the I/O bus. Among other diagnostics made, if the controller determines that there are I/O modules defined in the configuration that are not physically present on the I/O bus, an error is detected and the I/O bus does not start.

With the **Optional module** feature, the controller ignores the absent I/O expansion modules that you have marked as optional, which then allows the controller to start the I/O expansion bus.

The controller starts the I/O expansion bus at configuration time (following a power cycle, application download, or initialization command) even if optional expansion modules are not physically connected to the controller.

The TM3 I/O expansion modules can be marked as optional.

NOTE: TM3 Transmitter/Receiver modules (the TM3XTRA1 and the TM3XREC1) cannot be marked as optional.

You must be fully aware of the implications and impacts of marking I/O modules as optional in your application, both when those modules are physically absent and present when running your machine or process. Be sure to include this feature in your risk analysis.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Include in your risk analysis each of the variations of I/O configurations that can be realized marking I/O expansion modules as optional, and in particular the establishment of TM3 Safety modules (TM3S...) as optional I/O modules, and make a determination whether it is acceptable as it relates to your application.

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

Marking an I/O Expansion Module as Optional

To add an expansion module and mark it as optional in the configuration:

Step	Action																																																																														
1	Add the expansion module to your controller .																																																																														
2	In the Devices tree , double-click the expansion module.																																																																														
3	Select the I/O Configuration tab.																																																																														
4	In the Optional module line, select Yes in the Value column: <table><tr><th>Parameter</th><th>Type</th><th>Value</th><th>Default Value</th><th>Unit</th><th>Description</th></tr><tr><td> Optional module</td><td>Enumeration of BYTE</td><td> Yes</td><td>No</td><td></td><td></td></tr><tr><td> Outputs</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td> QW0</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td> Type</td><td>Enumeration of BYTE</td><td>Not used</td><td>Not used</td><td></td><td>Range mode</td></tr><tr><td> Minimum</td><td>INT(-32768...32766)</td><td>-32768</td><td>-32768</td><td></td><td>Minimum value</td></tr><tr><td> Maximum</td><td>INT(-32767...32767)</td><td>32767</td><td>32767</td><td></td><td>Maximum value</td></tr><tr><td> QW1</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td> Type</td><td>Enumeration of BYTE</td><td>Not used</td><td>Not used</td><td></td><td>Range mode</td></tr><tr><td> Minimum</td><td>INT(-32768...32766)</td><td>-32768</td><td>-32768</td><td></td><td>Minimum value</td></tr><tr><td> Maximum</td><td>INT(-32767...32767)</td><td>32767</td><td>32767</td><td></td><td>Maximum value</td></tr><tr><td> Diagnostic</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td> Status Enabled</td><td>Enumeration of BYTE</td><td>Yes</td><td>Yes</td><td></td><td></td></tr></table> Modifiable by programming = Yes = No	Parameter	Type	Value	Default Value	Unit	Description	Optional module	Enumeration of BYTE	Yes	No			Outputs						QW0						Type	Enumeration of BYTE	Not used	Not used		Range mode	Minimum	INT(-32768...32766)	-32768	-32768		Minimum value	Maximum	INT(-32767...32767)	32767	32767		Maximum value	QW1						Type	Enumeration of BYTE	Not used	Not used		Range mode	Minimum	INT(-32768...32766)	-32768	-32768		Minimum value	Maximum	INT(-32767...32767)	32767	32767		Maximum value	Diagnostic						Status Enabled	Enumeration of BYTE	Yes	Yes		
Parameter	Type	Value	Default Value	Unit	Description																																																																										
Optional module	Enumeration of BYTE	Yes	No																																																																												
Outputs																																																																															
QW0																																																																															
Type	Enumeration of BYTE	Not used	Not used		Range mode																																																																										
Minimum	INT(-32768...32766)	-32768	-32768		Minimum value																																																																										
Maximum	INT(-32767...32767)	32767	32767		Maximum value																																																																										
QW1																																																																															
Type	Enumeration of BYTE	Not used	Not used		Range mode																																																																										
Minimum	INT(-32768...32766)	-32768	-32768		Minimum value																																																																										
Maximum	INT(-32767...32767)	32767	32767		Maximum value																																																																										
Diagnostic																																																																															
Status Enabled	Enumeration of BYTE	Yes	Yes																																																																												

Shared Internal ID Codes

Controllers and bus couplers identify expansion modules by a simple internal ID code. This ID code is not specific to each reference, but identifies the logical structure of the expansion module. Therefore, different references can share the same ID code.

You cannot have two modules with the same internal ID code declared as optional without at least one mandatory module placed between them.

This table groups the module references sharing the same internal ID code:

Modules sharing the same internal ID code
TM3DI16K, TM3DI16, TM3DI16G
TM3DQ16R, TM3DQ16RG, TM3DQ16T, TM3DQ16TG, TM3DQ16TK, TM3DQ16U, TM3DQ16UG, TM3DQ16UK
TM3DQ32TK, TM3DQ32UK
TM3DI8, TM3DI8G, TM3DI8A
TM3DQ8R, TM3DQ8RG, TM3DQ8T, TM3DQ8TG, TM3DQ8U, TM3DQ8UG
TM3DM8R, TM3DM8RG
TM3DM24R, TM3DM24RG
TM3SAK6R, TM3SAK6RG
TM3SAF5R, TM3SAF5RG
TM3SAC5R, TM3SAC5RG
TM3SAFL5R, TM3SAFL5RG
TM3AI2H, TM3AI2HG
TM3AI4, TM3AI4G
TM3AI8, TM3AI8G
TM3AQ2, TM3AQ2G
TM3AQ4, TM3AQ4G
TM3AM6, TM3AM6G
TM3TM3, TM3TM3G
TM3TI4, TM3TI4G
TM3TI4D, TM3TI4DG
TM3TI8T, TM3TI8TG
TM3XFHSC202, TM3XFHSC202G
TM3XHSC202, TM3XHSC202G

Optional Modules Diagnostic

The following diagnostic information is available: **TM3_MODULE_R[i].i_wModuleState** system variable, where **[i]** identifies the absent TM3 optional expansion module, is set to **TM3_MISSING_OPT_MOD**.

Ethernet Configuration

Introduction

This chapter describes how to configure the Ethernet network interface of the Modicon M262 Logic/Motion Controller.

Ethernet Features, Functions and Services

Presentation

Ethernet Features, Functions and Services

The controller supports the following services:

- Modbus TCP Server, page 119
- Modbus TCP Client, page 119
- DHCP Server, page 156
- Web Server, page 122
- FTP Server, page 120
- SNMP, page 121
- Controller as Target Device On EtherNet/IP, page 156
- Controller as Slave Device On Modbus TCP, page 175
- IEC VAR ACCESS, page 113
- WebVisualisation, page 118
- OPC UA Server, page 201

TM262• Specific Considerations

The TM262• have two different Ethernet networks. Each one gets its own and unique IP and MAC address.

The two Ethernet networks are called Ethernet 1 and Ethernet 2:

- Ethernet 1 is a separate 100 Mbit/s Ethernet port and dedicated to the Sercos communication for the TM262M•.
- Ethernet 2 is a dual 1000 Mbit/s port Ethernet switch.

For example, you can:

- Connect your PC to the Ethernet 1
- Use a Modbus TCP I/O scanner with the Ethernet 2.

The Network Variables List (NVL) communication works on the Ethernet 1 port and Ethernet 2 port, only if both the Ethernet 1 port and Ethernet 2 port have a valid IP address and are connected to a device.

In addition, the TM262• allows you to connect your computer to the controller with a USB cable and to access the same services as with an Ethernet connection . See [Connecting the Controller to a PC](#), page 221.

Ethernet Protocols

The controller supports the following protocols:

- IP (Internet Protocol) V4, V6

- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- ARP (Address Resolution Protocol)
- ICMP (Internet Control Messaging Protocol)
- IGMP (Internet Group Management Protocol)

Communication Libraries

The communication libraries can be used in EcoStruxure Machine Expert. Refer to Communication Libraries folder of EcoStruxure Machine Expert online help.

Connections

This table shows the maximum number of connections:

Connection Type	Maximum
Modbus Server	8 connections
Modbus Client	8 connections
Modbus TCP I/O Scanner	64 channels
EtherNet/IP Scanner	64 connections
FTP Server	8 connections
Web server	10 concurrent users
Machine Expert Protocol (EcoStruxure Machine Expert software, trace, WebVisualisation, HMI devices)	8
OPC UA Server	4 connections
OPC UA Client	5 connections

Each connection based on TCP manages its own set of connections as follows:

1. When a client tries to open a connection that exceeds the poll size, the controller closes the oldest connection.
2. If all connections are busy (exchange in progress) when a client tries to open a new one, the new connection is denied.
3. The server connections stay open as long as the controller stays in operational states (*RUNNING*, *STOPPED*, *HALT*).
4. The server connections are closed when leaving operational states (*RUNNING*, *STOPPED*, *HALT*), except in case of power outage (because the controller does not have time to close the connections).

Connections can be closed when the originator of the connection requests to close the connection it had previously opened.

Services Available

With an Ethernet communication, the **IEC VAR ACCESS** service is supported by the controller. With the **IEC VAR ACCESS** service, data can be exchanged between the controller and an HMI.

The **NetWork variables** service is also supported by the controller. With the **NetWork variables** service, data can be exchanged between controllers.

NOTE: For more information, refer to the EcoStruxure Machine Expert Programming Guide.

IP Address Configuration

Introduction

There are different ways to assign the IP address to the added Ethernet interface of the controller:

- Address assignment by DHCP server based on the Network Name of the Ethernet interface
- Address assignment by BOOTP server based on the MAC address of the Ethernet interface
- Fixed IP address
- Post configuration file, page 215. If a post configuration file exists, this assignment method has priority over the others.

The IP address can also be changed dynamically through the:

- Communication Settings, page 61 tab in EcoStruxure Machine Expert
- **changelPAddress** function block, page 266

NOTE: If the attempted addressing method is unsuccessful, the link uses a default IP address, page 116 derived from the MAC address.

Carefully manage the IP addresses because each device on the network requires a unique address. Having multiple devices with the same IP address can cause unintended operation of your network and associated equipment.

WARNING

UNINTENDED EQUIPMENT OPERATION

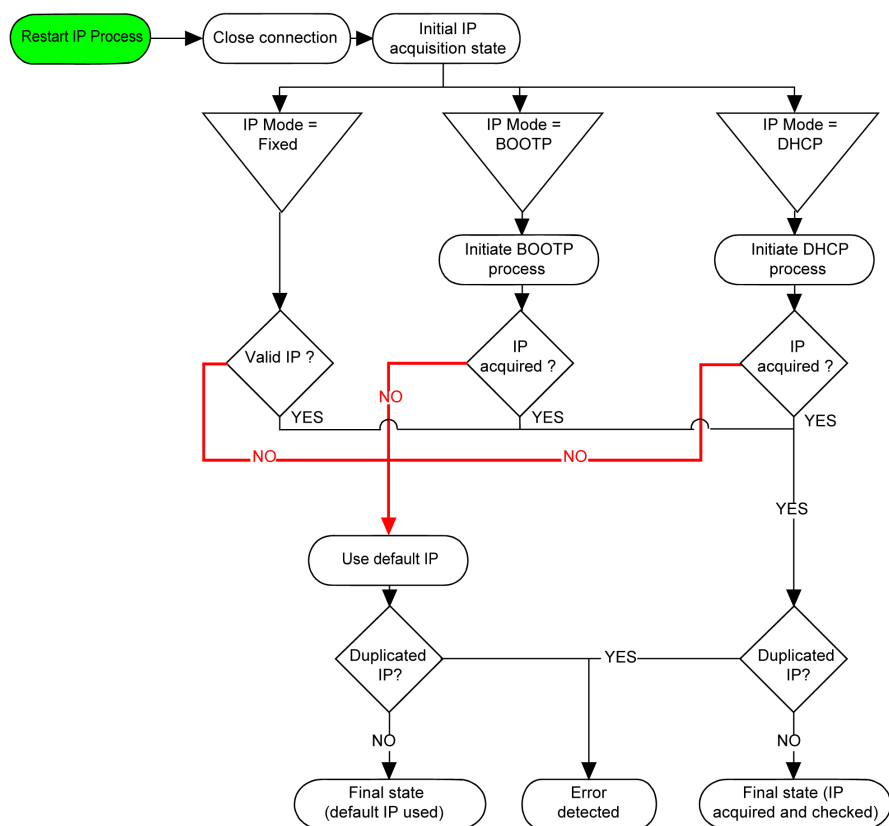
- Verify that there is only one master controller configured on the network or remote link.
- Verify that all devices have unique addresses.
- Obtain your IP address from your system administrator.
- Confirm that the IP address of the device is unique before placing the system into service.
- Do not assign the same IP address to any other equipment on the network.
- Update the IP address after cloning any application that includes Ethernet communications to a unique address.

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

NOTE: Verify that your system administrator maintains a record of assigned IP addresses on the network and subnetwork, and inform the system administrator of any configuration changes performed.

Address Management

This diagram shows the different types of address systems for the controller:



NOTE: If a device programmed to use the DHCP or BOOTP addressing methods is unable to contact its respective server, the controller uses the default IP address. It repeats its request constantly.

The IP process restarts in the following cases:

- Controller reboot
- Ethernet cable reconnection
- Application download (if IP parameters change)
- DHCP or BOOTP server detected after a prior addressing attempt was unsuccessful.

Ethernet Configuration

In the **Devices tree**, double-click **Ethernet_1** or **Ethernet_2**

Configured Parameters

Network Name

☒ IP Address by DHCP
☐ IP Address by BOOTP
☐ fixed IP Address

IP Address

Subnet Mask

Gateway Address

Ethernet Protocol

Transfer Rate

Security Parameters

Protocol inactive

Modbus Server
SNMP protocol
WebVisualisation protocol

Protocol active

Discovery protocol
FTP Server
Machine Expert protocol
Remote connection (Fast TCP)
Secured Web Server (HTTPS)

>>
<<

Ring topology options

Ring topology

Note: If you are in online mode, you see the two windows. You cannot edit them. If you are in offline mode, you see the **Configured Parameters** window and, for **Ethernet_2** the **Ring topology options** window. You can edit them.

This table describes the configured parameters:

Configured Parameters	Description
Interface Name	Name of the network link. Visible in online mode.
Network Name	Used as device name to retrieve IP address through DHCP, maximum 15 characters. NOTE: The network name modification is applied at next power ON.
IP Address by DHCP	IP address is obtained by DHCP server.
IP Address by BOOTP	IP address is obtained by BOOTP server. MAC address is located on the front of the controller.
Fixed IP Address	IP address, Subnet Mask, and Gateway Address are defined by the user.
Ethernet Protocol	Protocol type used: Ethernet 2
Transfer Rate	Speed and Duplex are in auto-negotiation mode.

Default IP Address

The default IP addresses are:

- 10.10.x.y. for Ethernet_1

- 10.11.x.y. for Ethernet_2

When TM262• is not configured, TMSES4 boots and automatically gets its default IP address:

- 10.12.x.z for the first module
- 10.13.x.z for the second module
- 10.14.x.z for the third module

x represents the 5th and y or z represent the 6th bytes of interface MAC address. For example, with a MAC address of 00:80:F4:4E:02:5D, the IP address will be 10.12.2.93

NOTE: The IP addresses must not be in the same IP network.

The MAC address of the Ethernet port can be retrieved on the label placed on the front side of the controller. The MAC address of the TMSES4 port can be calculated with the controller MAC address port Ethernet_2.

NOTE: For EcoStruxure Machine Expert versions prior to V1.2.4, the MAC address is determined by the value on the left side of the controller. See the EcoStruxure Machine Expert - Compatibility and Migration User Guide (see EcoStruxure Machine Expert Compatibility and Migration, User Guide).

The default subnet masks are:

- 255.255.0.0 for Ethernet_1
- 255.255.0.0 for Ethernet_2

NOTE: A MAC address is written in hexadecimal format and an IP address in decimal format. Convert the MAC address to decimal format.

Example of conversion:

Port	MAC address	IP address
Ethernet_1	MAC@Eth1:00.80.F4.4E.24.10	10.10.36.16
Ethernet_2	MAC@Eth2:00.80.F4.4E.24.0B	10.11.36.11
TMS_1	MAC@TMS:00.80.F4.50.24.0B	10.12.36.11
TMS_2	MAC@TMS:00.80.F4.50.24.0C	10.13.36.12
TMS_3	MAC@TMS:00.80.F4.50.24.0D	10.14.36.13

NOTE: The TMSES4 MAC address is calculated as follows: $MAC@TMS\_x = MAC@Ethernet2 + 0x20000 + (x-1)$.

Prohibited IP Addresses

USB Network address (192.168.200.0) and TMS Network address (192.168.2.0) are prohibited.

Subnet Mask

The subnet mask is used to address several physical networks with a single network address. The mask is used to separate the subnetwork and the device address in the host ID.

The subnet address is obtained by retaining the bits of the IP address that correspond to the positions of the mask containing 1, and replacing the others with 0.

Conversely, the subnet address of the host device is obtained by retaining the bits of the IP address that correspond to the positions of the mask containing 0, and replacing the others with 1.

Example of a subnet address:

IP address	192 (11000000)	1 (00000001)	17 (00010001)	11 (00001011)
Subnet mask	255 (11111111)	255 (11111111)	240 (11110000)	0 (00000000)
Subnet address	192 (11000000)	1 (00000001)	16 (00010000)	0 (00000000)

NOTE: The device can communicate only on its subnetwork when there is no gateway.

Gateway Address

The gateway allows a message to be routed to a device that is not on the same network.

If there is no gateway, the gateway address is 0.0.0.0.

The default gateway address can be defined on any interface. You can only configure the default gateway on one interface. The traffic to unknown networks is sent through this interface. Please refer to [IP Routing](#), page 65 if you need to configure more than one interface.

Security Parameters

This table describes the different security parameters:

Security Parameters	Description	Default settings
Discovery protocol	This parameter deactivates Discovery protocol . When deactivated, Discovery requests are ignored.	Active
FTP Server	This parameter deactivates the FTP server of the controller. When deactivated, FTP requests are ignored.	Active
Machine Expert protocol	This parameter deactivates the Machine Expert protocol on Ethernet interfaces. When deactivated, any Machine Expert request from any device is rejected. Therefore, no connection is possible on Ethernet from a PC with EcoStruxure Machine Expert, from an HMI target that wants to exchange variables with this controller, from an OPC server, or from Controller Assistant.	Active
Modbus Server	This parameter deactivates the Modbus server of the controller. When deactivated, any Modbus request to the controller is ignored.	Inactive
Remote connection (Fast TCP)	This parameter deactivates the remote connection. When deactivated, Fast TCP requests are ignored.	Active
Secured Web Server (HTTPS)	This parameter deactivates the secured Web server of the controller. When deactivated, HTTPS requests to the controller secured Web server are ignored.	Active
SNMP protocol	This parameter deactivates the SNMP server of the controller. When deactivated, SNMP requests are ignored.	Inactive
WebVisualisation protocol	This parameter deactivates the WebVisualisation pages of the controller. When deactivated, HTTP requests to the controller WebVisualisation protocol are ignored.	Inactive

Ring Topology Options

This parameter is only available on the Ethernet_2 network.

This table describes the **Ring topology** options:

Options	Description
No ring	If selected, verify that no ring is wired.
Root	First device of the ring topology.
Participant	One of the devices in the ring topology.

Each device in the ring topology must support the Rapid Spanning Tree Protocol (RSTP).

You can have up to 40 devices in the ring topology.

Modbus TCP Client/Server

Introduction

Unlike Modbus serial link, Modbus TCP is not based on a hierarchical structure, but on a client/server model.

The Modicon M262 Logic/Motion Controller implements both client and server services so that it can initiate communications to other controllers and I/O devices, and respond to requests from other controllers, SCADA, HMIs, and other devices.

Without any configuration, the embedded Ethernet port of the controller supports Modbus server.

The Modbus client/server is included in the firmware and does not require any programming action from the user. Due to this feature, it is accessible in RUNNING, STOPPED and EMPTY states.

Modbus TCP Client

The Modbus TCP client supports the following function blocks from the PLCCommunication library without any configuration:

- ADDM
- READ_VAR
- SEND_RECV_MSG
- SINGLE_WRITE
- WRITE_READ_VAR
- WRITE_VAR

For further information, refer to the Function Block Descriptions (see EcoStruxure Machine Expert, Modbus and ASCII Read/Write Functions, PLCCommunication Library Guide).

Modbus TCP Server

The Modbus server supports the Modbus requests:

Function Code Dec (Hex)	Subfunction Dec (Hex)	Function
1 (1)	–	Read digital outputs (%Q)
2 (2)	–	Read digital inputs (%I)
3 (3)	–	Read holding register (%MW)
6 (6)	–	Write single register (%MW)
8 (8)	–	Diagnostic
15 (F)	–	Write multiple digital outputs (%Q)
16 (10)	–	Write multiple registers (%MW)
23 (17)	–	Read/write multiple registers (%MW)
43 (2B)	14 (E)	Read device identification

Diagnostic Request

This table contains the data selection code list:

Data Selection Code (hex)	Description
00	Reserved
01	Basic Network Diagnostics
02	Ethernet Port Diagnostic
03	Modbus TCP/Port 502 Diagnostics
04	Modbus TCP/Port 502 Connection Table
05 - 7E	Reserved for other public codes
7F	Data Structure Offsets

FTP Server

Introduction

Any FTP client installed on a computer that is connected to the controller (Ethernet port), without EcoStruxure Machine Expert installed, can be used to transfer files to and from the data storage area of the controller.

NOTE: Schneider Electric adheres to industry best practices in the development and implementation of control systems. This includes a "Defense-in-Depth" approach to secure an Industrial Control System. This approach places the controllers behind one or more firewalls to restrict access to authorized personnel and protocols only.

WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT UNAUTHORIZED MACHINE OPERATION

- Evaluate whether your environment or your machines are connected to your critical infrastructure and, if so, take appropriate steps in terms of prevention, based on Defense-in-Depth, before connecting the automation system to any network.
- Limit the number of devices connected to a network to the minimum necessary.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.
- Monitor activities within your systems.
- Prevent subject devices from direct access or direct link by unauthorized parties or unauthenticated actions.
- Prepare a recovery plan including backup of your system and process information.

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

NOTE: Make use of the security-related commands (see EcoStruxure Machine Expert, Menu Commands, Online Help) which provide a way to add, edit, and remove a user in the online user management of the target device where you are currently logged in.

FTP Access

Access to the FTP server is controlled by User Rights when they are enabled in the controller. For more information, refer to **Users Rights** description, page 69.

To access the FTP server you must first connect to the controller with EcoStruxure Machine Expert or Controller Assistant and activate the user rights or create the user for the first login.

NOTE: FTPS (explicit over TLS FTP) is set by default. Simple FTP (non secure) access is not possible at first connection. Set the parameter 1106 to 0 in the post configuration and reboot the controller to allow Simple FTP connection.

FTP Client

The M262 Logic/Motion Controller supports an FTP client library to allow you to query FTP servers. For details, refer to the *FtpRemoteFileHandling – Library Guide*.

Files Access

See File Organization, page 26.

SNMP

Introduction

The Simple Network Management Protocol (SNMP) is used to provide the data and services required for managing a network.

The data is stored in a Management Information Base (MIB). The SNMP protocol is used to read or write MIB data. Implementation of the Ethernet SNMP services is minimal, as only the compulsory objects are handled.

SNMP Server

This table presents the supported standard MIB-2 server objects:

Object	Description	Access	Value
sysDescr	Text description of the device	Read	SCHNEIDER M262 Fast Ethernet TCP/IP
sys-Name	Node administrative name	Read/Write	Controller reference

The size of these character strings is limited to 50 characters.

The values written are saved to the controller via SNMP client tool software. The Schneider Electric software for this is ConneXview. ConneXview is not supplied with the controller or bus coupler. For more details, refer to www.se.com.

SNMP Client

The M262 Logic/Motion Controller supports an SNMP client library to allow you to query SNMP servers. For details, refer to the *EcoStruxure Machine Expert SnmpManager, Library Guide*.

Web Server

Introduction

The Web server is a tool that allows you to remotely monitor a controller and its application, perform various maintenance activities including modifications to data and configuration parameters, and change the state of the controller.

As standard equipment, the controller provides an embedded Web server with a predefined, built-in website. You can use the website for module setup and control, as well as application diagnostics and monitoring. These pages are ready for use with a Windows Web browser or mobile device. No configuration or programming is required.

The Web server can be accessed by the web browsers listed below:

- Google Chrome (version 87 or greater)
- Mozilla Firefox (version 62 or greater)
- Microsoft Internet Explorer (version 11 or greater)

The Web server can be accessed by the mobile device web browsers listed below:

- iOS Safari
- Android Chrome

HTTP (non secured connections) requests are redirected to HTTPS (secure connections).

The Web server is limited to 10 concurrent users, page 113.

The Web server has access to your application for reading and writing data and controlling the state of the controller. By enabling the Web server, you enable these functions. You can disable the Web server on an interface by deselecting the Web server active parameter in the Ethernet Configuration tab, page 116.

If there are security concerns over these functions, you must, at a minimum, assign a secure password to the Web server or disable the Web server to prevent unauthorized access to the application. Care must be taken to ensure that the immediate physical environment of the machine and process is in a state that will not present safety risks to people or property before exercising control remotely.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Define a secure password for the Web server and do not allow unauthorized or otherwise unqualified personnel to use this feature.
- Ensure that there is a local, competent, and qualified observer present when operating on the controller from a remote location.
- You must have a complete understanding of the application and the machine/process it is controlling before attempting to adjust data, stopping an application that is operating, or starting the controller remotely.
- Take the precautions necessary to assure that you are operating on the intended controller by having clear, identifying documentation within the controller application and its remote connection.

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

Web Server Access

Access to the Web server is controlled by User Rights when they are enabled in the controller. For more information, refer to **Users and Groups**, page 60.

To access the Web server you must first connect to the controller with EcoStruxure Machine Expert or Controller Assistant and modify the default user password.

⚠ WARNING

UNAUTHORIZED DATA ACCESS

- Secure access to the FTP/Web server using User Rights.
- If you disable User Rights, disable the FTP/Web server to prevent any unwanted or unauthorized access to data in your application.

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

In order to change the password, go to **Users and Groups** tab of the device editor. For more information, refer to the EcoStruxure Machine Expert Programming Guide.

NOTE: The only way to gain access to a controller that has user access-rights enabled and for which you do not have the password(s) is by performing an Update Firmware operation. This clearing of User Rights can only be accomplished by using a SD card to update the controller firmware. In addition, you may clear the User Rights in the controller by running a script (refer to [Reset the User Rights to Default](#), page 234). This effectively removes the existing application from the controller memory, but restores the ability to access the controller.

Home Page Access

To access the website home page, type the IP address of the controller into the browser.

This figure shows the Web server site login page:



The screenshot shows a web browser window. The address bar contains the URL `https://85.15.1.51/login.htm`. Below the address bar, the page title is `85.15.1.51/login.htm`. The main content area displays a login form with two input fields: `User:` and `Password:`. A `Login` button is positioned below the password field.

This figure shows the home page of the Web server site once you have logged in:



NOTE: Schneider Electric adheres to industry best practices in the development and implementation of control systems. This includes a "Defense-in-Depth" approach to secure an Industrial Control System. This approach places the controllers behind one or more firewalls to restrict access to authorized personnel and protocols only.

⚠ WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT UNAUTHORIZED MACHINE OPERATION

- Evaluate whether your environment or your machines are connected to your critical infrastructure and, if so, take appropriate steps in terms of prevention, based on Defense-in-Depth, before connecting the automation system to any network.
- Limit the number of devices connected to a network to the minimum necessary.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.
- Monitor activities within your systems.
- Prevent subject devices from direct access or direct link by unauthorized parties or unauthenticated actions.
- Prepare a recovery plan including backup of your system and process information.

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

Home Page Access Menu

The Home Page Access menu bar lets you access the main Web server pages.

The Web server contains the following pages:

Menu	Page	Description
Home	Home, page 123	Home page of the controller Web server page. Provides access to the tabs: • Monitoring • Diagnostics • Maintenance • Machine Assistant

Home page menu descriptions:

Menu	Submenu	Description
Monitoring	Data Parameters, page 127	Lets you display and modify controller variables.
	IO Viewer, page 128	Shows the module with module I/O values.
	Oscilloscope, page 128	Displays 2 variables in the form of a recorder-type time chart.
Diagnostics	Controller, page 129	Displays controller status.
	Ethernet, page 130	Displays Ethernet diagnostic.
	TM3 Expansion, page 131	Displays expansion module status.
	TMS Expansion, page 132	Displays expansion module status.
	TMSES4, page 133	Displays TMSES4 status.
	Scanner Status, page 133	Displays serial line status.
	EtherNet/IP Status, page 134	Displays Ethernet status.
Maintenance	Post configuration , page 134	Lets you access the post configuration file saved on the controller.
	User Management, page 135	Lets you change user password and customize login message. Possible in secure mode (HTTPS) only. • Change password (of current user): lets you change user password. • Users account management: allows you to remove all passwords from the controller and reset user accounts to their default state. • Clone management: lets you include or exclude user access rights when cloning a controller. • System use notification: lets you customize a message which will be displayed at login.
	Firewall, page 136	Lets you modify the firewall configuration.
	System Log Files, page 137	Lets you access log files generated by the controller.
	Message Logger, page 137	Lets you access controller messages.
	Run/Stop Controller, page 138	Lets you send Run and Stop commands to the controller.
	SelfAwareness, page 138	Lets you access temperature, memory usage, processor load and devices information.
	Certificates, page 139	Lets you customize certificates owned by a Modicon M262 Logic/Motion Controller.
	Date / Time, page 139	Lets you set the date, time, time zone and optional daylight saving time.
	SCEP, page 140	Lets you access to the configuration of the SCEP server.
Machine Assistant	List View	Displays the configuration in list view.
	Graphic view	Displays the configuration in graphic view.
	Scan, page 261	Lets you scan the devices configured.
	Clear, page 261	Lets you clear the scan.
	load .semtd file, page 264	Lets you upload a .semtd file after scan.
	Export scan results, page 264	Lets you export the scan results in your local SD Card.
	Log out	Lets you log out.

The Web server allows you to remotely monitor a controller and its application, and to perform various maintenance activities including modifications to data and configuration parameters, and change the state of the controller. Ensure that the immediate physical environment of the machine and process is in a state that will not present safety risks to people or property before exercising control remotely.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Configure and install the RUN/STOP input for the application, if available for your particular controller, so that local control over the starting or stopping of the controller can be maintained regardless of the remote commands sent to the controller.
- Define a secure password for the Web server and do not allow unauthorized or otherwise unqualified personnel to use this feature.
- Ensure that there is a local, competent, and qualified observer present when operating on the controller from a remote location.
- You must have a complete understanding of the application and the machine/process it is controlling before attempting to adjust data, stopping an application that is operating, or starting the controller remotely.
- Take the precautions necessary to assure that you are operating on the intended controller by having clear, identifying documentation within the controller application and its remote connection.

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

Monitoring Menu

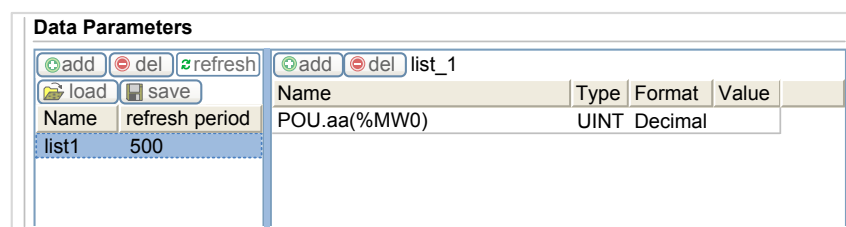
Monitoring: Data Parameters

Monitoring Web Server Variables

To monitor Web server variables, you must select the variables in the *Symbol Configuration Editor*, page 88.

Monitoring: Data Parameters Submenu

The **Data Parameters** submenu allows you to display and modify variable values:



Element	Description
Add	Adds a list description or a variable
Del	Deletes a list description or a variable
Refresh period	Refreshing period of the variables contained in the list description (in ms)
Refresh	Enables I/O refreshing: • Gray button: refreshing disabled • Orange button: refreshing enabled NOTE: Without enabling Refresh, when a variable's value is changed in the table the modification is directly sent to the controller.
Load	Loads saved lists from the controller internal Flash to the Web server page
Save	Saves the selected list description in the controller (<i>/usr/web</i> directory)

NOTE: The IEC objects (%MX, %IX, %QX) are not directly accessible. To access IEC objects you must first group their contents in located registers (refer to Relocation Table, page 30).

Monitoring: IO Viewer Submenu

You must add the I/Os in the **Symbol Configuration Editor** to get them visible in the **IO Viewer**. Refer to *Symbol Configuration Editor*, page 88.

The **IO Viewer** submenu allows you to display the I/O values:

IO Viewer

Period (ms)

Mapping	Address	Type	Format	Value

Element	Description
Refresh	Enables I/O refreshing: - Gray button: refreshing disabled - Orange button: refreshing enabled
Period (ms)	I/O refreshing period in ms
<<	Goes to previous I/O list page
>>	Goes to next I/O list page

Monitoring: Oscilloscope Submenu

The **Oscilloscope** submenu can display up to 2 variables in the form of a recorder time chart:

Oscilloscope

Item0:
min:
max:

Item1:
min:
max:
Period (ms)

Element	Description
Reset	Erases the memorization
Refresh	Starts/stops refreshing
Load	Loads parameter configuration of Item0 and Item1
Save	Saves parameter configuration of Item0 and Item1 in the controller
Item0	Variable to be displayed
Item1	Variable to be displayed
Min	Minimum value of the variable axis
Max	Maximum value of the variable axis
Period(ms)	Page refresh period in milliseconds

Diagnostic Menu

Diagnostic: Controller Submenu

The **Controller** submenu displays information about the controller:

Diagnostics
Controller
Ethernet
TM3 Expansion
TMS Expansion
TMSES4
Scanner Status
EtherNet/IP Status

Controller

Reset Statistics

Identification	Status
Vendor 0x101a	Application status Running (2)
Vendor name Schneider Electric	Boot project status Same boot project (65535)
Product 0x811	IO Status 1 Power supply fault (4)
Product reference TM262-25	IO Status 2 Ok (FFFF)
Serial Number 130	Application signature B9394DC8
Node name TM262-25	Application signature 0
	Application signature 0
	Application signature 0
	Last stop cause Powerfail (15)
	Last application error Software watchdog of IEC-task expired (16)
	System Fault 1 TMS module fault
	System Fault 2 No error
	Last stop time Fri, 12 Oct 2018 10:45:31
	Last power-off time Fri, 12 Oct 2018 10:51:20
	Events counter 0
	SdCard None (0)
	USB Programming port Not connected (0)

Version	File
Firmware 4.0.0.36	File system free handle 1974
Boot 0.0.0.35	File system total bytes 1073741824 (1024 MB)
Hardware 0x20202	File system free bytes 1062559744 (1013 mb)
Coprocessor 0x80280100	

Extension bus	File
TM3 Bus status 0b00000000000000011 OK	

Diagnostics: Ethernet Submenu

The **Ethernet** submenu displays the Ethernet ports status and access to the remote ping service:

Remote Ping Service

Enter IP address to ping from Controller:

Statistics

Ethernet_1	
MAC address	00.80.F4.4E.00.5C
IP address	85.50.60.70
Subnet mask	255.0.0.0
Gateway address	0.0.0.0
Status	Link up (1)
Speed	100

Ethernet_2	
MAC address	00.80.F4.4E.00.5B
IP address	10.11.0.91
Subnet mask	255.255.0.0
Gateway address	0.0.0.0
Status	Link up (1)
Speed	0

Ethernet statistics	
Opened Top connections	8
Frames transmitted OK	86132098
Frames received OK	452354445
Buffers transmitted NOK	0
Buffers received NOK	178123357

Modbus statistics	
Messages transmitted OK	0
Messages received OK	0
Error messages	0
IpMaster connection status	Not connected (1)
IpMaster timeout event counter	0

Ethernet IP Adapter statistics	
IO Messages transmitted	0
IO Messages received	0
UCMM Request	0
UCMM Error	0
Class3 Request	0
Class3 Error	0
Assembly Instance Input	0
Assembly Instance Input size	0
Assembly Instance Output	0
Assembly Instance Output size	0

Diagnostics: TM3 Expansion Submenu

The **TM3 Expansion viewer** submenu shows the expansion modules status:

Expansion viewer | << < < 1 - 8 / 14 > >> >> |

Expansion 1	Expansion 2
Module ID -	Module ID -
Status Inactive (0)	Status Inactive (0)
Expansion 3	Expansion 4
Module ID -	Module ID -
Status Inactive (0)	Status Inactive (0)
Expansion 5	Expansion 6
Module ID -	Module ID -
Status Inactive (0)	Status Inactive (0)
Expansion 7	Expansion 8
Module ID -	Module ID -
Status Inactive (0)	Status Inactive (0)

Diagnostics: TMS Expansion Submenu

The **TMS Expansion viewer** submenu shows the expansion modules status:

Expansion viewer << < < 1 - 7 / 7 > >> >>	
Expansion 1	
Name	TMSES4
Major type	1
Sub. type	1
Version	1.0.0.3
Module status	Configured (2)
IP status	Ping Success (0)
Pix command status	Disabled (12)
Expansion 2	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)
Expansion 3	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)
Expansion 4	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)
Expansion 5	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)
Expansion 6	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)
Expansion 7	
Name	
Major type	0
Sub. type	0
Version	
Module status	Discovery (9)
IP status	Not Configured (10)
Pix command status	Disabled (12)

Diagnostics: TMSES4 Submenu

The **TMSES4 Devices viewer** submenu shows the status of the modules:

Devices viewer | << < < 1 - 3 / 3 > >> >> |

TMSES4 1

MAC address	00.80.F4.50.03.31
IP address	10.208.34.34
Subnet mask	255.255.254.0
Gateway address	10.208.34.1
Link Status	Link up (1)
IP Status	Data Exchange (2)

TMSES4 2

MAC address	00.80.F4.50.03.32
IP address	95.100.80.9
Subnet mask	255.255.0.0
Gateway address	0.0.0.0
Link Status	Link up (1)
IP Status	Data Exchange (2)

TMSES4 3

MAC address	00.80.F4.50.03.33
IP address	85.80.80.9
Subnet mask	255.0.0.0
Gateway address	0.0.0.0
Link Status	Link up (1)
IP Status	Data Exchange (2)

Diagnostics: Scanner Status Submenu

The **Scanner Status** submenu displays status of the Modbus TCP I/O Scanner (IDLE, STOPPED, OPERATIONAL) and the health bit of up to 64 Modbus slave devices:

Modbus TCP I/O Scanner

[Scanner Status](#)

Idle

[Connection Statistics](#)

Total transmissions sent: **0**

Number of Configured Connections: **0**

[Scanned Device Statuses](#)

No Scanned Devices Reported

☐ Not Configured
☒ Scanned
☒ Fault

For more information, refer to EcoStruxure Machine Expert Modbus TCP User guide.

Diagnostics: EtherNet/IP Status Submenu

The **EtherNet/IP Status** submenu displays the status of the EtherNet/IP Scanner (IDLE, STOPPED, OPERATIONAL) and the health bit of up to 64 EtherNet/IP target devices:

EIP I/O Scanner

Scanner Status

 Idle

Connection Statistics

Total transmissions sent: **0**
Number of Configured Connections: **0**

Scanned Device Statuses

No Scanned Devices Reported

☐ Not Configured

☒ Scanned

☒ Fault

For more information, refer to EcoStruxure Machine Expert EtherNet/IP User guide.

Maintenance Menu

Introduction

The Maintenance page provides access to the `/usr` folders of the controller non-volatile memory, page 26 and information for device maintenance purposes.

Step	Action
1	Click Load .
2	Modify the parameters, page 217.
3	Click Save . NOTE: The new parameters will be considered at next Post Configuration file reading, page 215.

Maintenance: Post Conf Submenu

The **Post Conf** submenu allows you to update the post configuration, page 215 file saved on the controller:

Post Conf

Load

Save

No Post Conf available

Maintenance: User Management Submenu

The **User Management** submenu displays a screen that allows you to access four different actions, restricted by using secure protocol (HTTPS):

- **Change password (of current user):**

allows you to change your password.

- **User accounts management:**

Allows you to manage user accounts management, removing the password and returning the user accounts on the controller to default settings.

Click **Disable** to deactivate the user rights on the controller. (Passwords are saved and are restored if you click **Enable**.)

Click **OK** on the window that appears to confirm. As a result:

- Users no longer have to set and enter a password to connect to the controller.
- FTP, HTTP, and OPC UA server connections accept anonymous user connections. See [Login and passwords table, page 70](#).

NOTE: The **Disable** button is only active if the user has administrator privileges.

Click **Enable** to restore the previous user rights saved on the controller.

Click **OK** on the window that appears to confirm. As a result, users have to enter the password previously set to connect to the controller. See [Login and passwords table, page 70](#)

NOTE: The **Enable** button only appears if the user rights were disabled and the user rights backup file is available on the controller.

Click **Reset to default** to return the user accounts on the controller to their default setting state.

Click **OK** on the window that appears to confirm.

NOTE: Connections to FTP, HTTP, and the OPC UA server are blocked until a new password is set.

- **Clone management:**

allows you to control whether user rights are copied and applied to the target controller when cloning a controller with an SD card, [page 237](#).

Click **Exclude users rights** to exclude copying user rights to the target controller when cloning a controller.

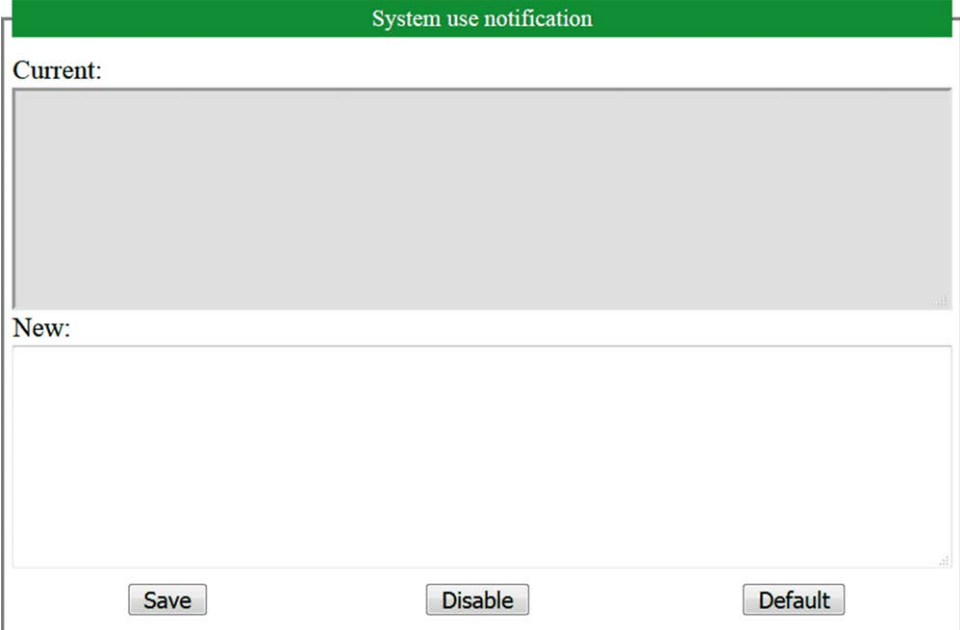
NOTE: By default, the users rights are excluded.

Click **Include users rights** to copy user rights to the target controller when cloning a controller. A popup prompts you to confirm copying the user rights. Click **OK** to continue.

NOTE: The **Exclude users rights** and **Include users rights** buttons are only active if the user is connected to the controller using a secure protocol.

- **System use notification:**

allows you to customize a message which will be displayed at login.



The image shows a window titled "System use notification" with a green header. It contains two text areas: "Current:" (with a grey placeholder) and "New:" (with a white placeholder). At the bottom, there are three buttons: "Save", "Disable", and "Default".

Maintenance: Firewall Submenu

The **Firewall** submenu allows you to modify the default firewall configuration file, page 142:



The image shows a window titled "Firewall". It contains two buttons, "Load" and "Save", followed by the text "No Firewall Conf available".

Maintenance: System Log Files Submenu

The **System Log Files** submenu provides access to log files generated by the controller:

System Log Files	
 FwLog.txt	8 kb FRI OCT 12 10:51:39 2018
 PlcLog_0.txt	104 kb FRI OCT 12 10:46:59 2018
 LoggerFile_11-10-2018_02h19m40s.mel	57 kb THU OCT 11 14:19:41 2018
 LoggerFile_11-10-2018_04h45m48s.mel	60 kb THU OCT 11 16:45:48 2018
 PlcLog_1.txt	104 kb FRI OCT 12 05:12:18 2018
 LoggerFile_11-10-2018_04h47m11s.mel	65 kb THU OCT 11 16:47:12 2018
 LoggerFile_11-10-2018_06h10m35s.mel	60 kb THU OCT 11 18:10:35 2018
 PlcLog_2.txt	104 kb FRI OCT 12 07:27:31 2018
 LoggerFile_11-10-2018_07h11m40s.mel	60 kb THU OCT 11 19:11:40 2018
 LoggerFile_11-10-2018_09h02m59s.mel	60 kb THU OCT 11 21:02:59 2018
 PlcLog.txt	24 kb FRI OCT 12 14:18:56 2018
 LoggerFile_11-10-2018_10h14m05s.mel	60 kb THU OCT 11 22:14:05 2018
 LoggerFile_12-10-2018_01h28m42s.mel	60 kb FRI OCT 12 01:28:42 2018
 LoggerFile_12-10-2018_02h30m44s.mel	60 kb FRI OCT 12 02:30:44 2018
 LoggerFile_12-10-2018_05h21m17s.mel	60 kb FRI OCT 12 05:21:17 2018
 LoggerFile_12-10-2018_06h23m39s.mel	60 kb FRI OCT 12 06:23:39 2018
 LoggerFile_12-10-2018_07h50m10s.mel	60 kb FRI OCT 12 07:50:11 2018
 LoggerFile_12-10-2018_08h38m01s.mel	59 kb FRI OCT 12 08:38:01 2018
 LoggerFile_12-10-2018_10h36m56s.mel	62 kb FRI OCT 12 10:36:56 2018
 LoggerFile_12-10-2018_10h37m19s.mel	64 kb FRI OCT 12 10:37:19 2018
 LoggerFile_12-10-2018_10h52m01s.mel	58 kb FRI OCT 12 10:52:01 2018

NOTE: A maximum of 300 log files can be stored in the **Message Logger**. When the maximum log file size is reached, previous logs must be deleted in order to continue saving new diagnostic information.

Maintenance: Message Logger Submenu

The **Message Logger** submenu displays latest controller log messages:

Message Logger							
				Load Save			
No.	Timestamp	Type	Object	Instance	Diag. code	Ext. diagnosis	Message

Maintenance: Run/Stop Controller Submenu

The **Run/Stop Controller** submenu allows you to manually stop and restart the controller:

Run/Stop Controller

Stop Controller

Identification	
Product reference	TM262-25
Serial Number	130
Node name	TM262-25
MAC address	00.80.F4.4E.00.5C
IP address	85.50.60.70
Subnet mask	255.0.0.0
Gateway address	0.0.0.0

Status	
Application status	Running (2)
Boot project status	Same boot project (65535)
Last stop cause	Powerfail (15)
Last application error	Software watchdog of IEC-task expired (16)
Last stop time	Fri, 12 Oct 2018 10:45:31
Last power-off time	Fri, 12 Oct 2018 10:51:20

Maintenance: SelfAwareness Submenu

The **SelfAwareness** submenu allows you to access temperature, memory usage, processor load and devices information:

Power On

Time (Seconds) 15241344
Count 2237

Power Faults

Bad Voltage 0
TMS Over-current 0

PLC Internal Temperatures (°C)

	Current	Max	
Power Supply Temp.	34	34	Reset
Ambient Temp.	28	28	Reset
TMS interface Temp.	35	35	Reset
CPU Board Temp.	35	35	Reset
CPU Internal Temp.	45	45	Reset

Memory Usage (bytes)

	Current	Max	
Allocated Mem:	195874344	204362952	Reset

Cpu Load (%)

☒ Communication core 25 Disable
☐ Application core 9 Disable

Chart – Communication core – 76s

NOTE: The sample rate is set at 4 seconds. Setting under 4 seconds increases the **Communication core** and **CPU Load**.

The controller internal ambient maximum temperature is 100 °C (212 °F). The external ambient maximum temperature can be found in the hardware guide of your controller.

Maintenance: Certificates Submenu

The following graphic shows the **Certificates** submenu:

The screenshot shows the 'Certificates' submenu with two main sections:

- 1: Own Certificate** (highlighted with a blue box):

	Current values	New values (updated after PLC reboot)
Country:	FR	
State:		
Locality:	Carros	
Organization:	Schneider-Electric	
Organization unit:	MachineSolutions	
Common name:	TM262-25	
DNS:		

 A 'Save' button is located at the bottom right of this section.
- 2: Client Certificates** (highlighted with a blue box):

Rejected	Trusted

 Navigation buttons '>>' and '<<' are located between the two columns.

1: Own Certificate allows you to modify the certificates owned by an M262 Logic/Motion Controller. The optional **DNS** value indicates the domain name for which the certificates is valid (OPC UA or HTTP/FTP).

NOTE: Any modification has an impact on OPC UA and HTTP/FTP certificates. See [Certificate Management](#), page 93.

NOTE: Any modification overwrites SCEP certificates and requires a new SCEP enrollment. See [Maintenance: Simple Certificate Enrollment Protocol \(SCEP\) Submenu](#), page 140.

2: Client Certificates allows you to determine which certificates are trusted by the M262 Logic/Motion Controller.

Maintenance: Date / Time Submenu

The **Date / Time** submenu displays the date, time, time zone and optional daylight saving time and allows you to manually change them:

The screenshot shows the 'Date / Time' submenu with the following sections:

- Read** (green header):

PLC Time	Fri Dec 03 2021 11:10:09 GMT+0100 (CET)
Local Time	Fri Dec 03 2021 11:10:07 GMT+0100 (Romance Standard Time)
- Update PLC** (green header):

Local Time

Date (yyyy-mm-dd)
Time (hh:mn:ss)

Time zone / Daylight Saving Time

+0100 +0200 Central European /Summer Time ▾

Relative correction

+/- 0..9999s

Maintenance: Simple Certificate Enrollment Protocol (SCEP) Submenu

The SCEP submenu allows you to communicate with a SCEP server. This section describes how to specify settings that allow the device to obtain certificates from a Certificate Authority (CA) using Simple Certificate Enrollment Protocol (SCEP).

SCEP Server Settings

SCEP Server URL

Certificate Revocation List URL

Certificate Authority Identifier (optional)

Certificate Authority

Get Certificate
Download Certificate
Trust Certificate

Certificate Revocation List

Get revocation list
Download revocation list

Certificate Authority Capabilities

Get capabilities

Diagnostic

Last command name:

None

Result: None

Message:

None:

Certificate to enroll

OPC UA ▼

Challenge password

Retry Period Minutes (1-60)

Retry Count (0-100)

Enroll
Check Status
Cancel

Current Enrollment Status

Idle

Certificate Status

Status: None

Message: None

The following table describes the **SCEP** submenu:

Element	Option	Description
SCEP Server Settings	SCEP Server URL	Allows you to specify the URL of the SCEP server to which the device should send certificate requests.
	Certificate Revocation List URL	Allows you to specify the URL of the Certificate Revocation List.
	Certificate Authority Identifier (Optional)	Allows you to choose which certificate is required if a Certificate Authority (CA) has multiple certificates.
Certificate Authority	Get Certificate	Allows you to obtain the certificate.
	Download Certificate	Allows you to download the certificate.
	Trust Certificate	Allows you to add the certificate to the trusted list of the device.
Certificate Revocation List	Get revocation list	Allows you to obtain the certification revocation list from the Certificate Authority (CA).
	Download revocation list	Displays the content of the Certificate Revocation List (CRL) received.
Certificate Authority Capabilities	Get capabilities	Allows you to request which functionality is available from the Certificate Authority (CA).

Element	Option	Description
Diagnostic	Last command name	Displays the last action executed, its result and a diagnostic message if necessary.
	Result	
	Message	
Certificate to enroll	Selection list	From the selection list, select one of the following options to configure the certificate to enroll: • OPC UA • HTTP (also used for FTP)
	Challenge password	Password used and provided by the Certificate Authority (CA) for router certificate enrollment and revocation.
	Retry Period	Specifies the delay, in minutes, between certificate request attempts.
	Retry Count	Specifies the number of times the device should resubmit a certificate request.
	Enroll	Allows you to start the enrollment process.
	Check Status	Allows you to verify the status of the enrollment process.
	Cancel	Allows you to cancel the enrollment process.
	Current Enrollment Status	Displays a message about the status of the enrollment process: • Idle • On going
	Certificate Status	Displays the status of the certificate and an associated message:
		Starting Enrollment process is starting
		Success Request pending for manual approval
		Pending • Request granted. Certificate will be applied on the next reboot or • Request granted. Certificate will be applied on the next reset cold, reset warm or application download
		Cancel Operation cancelled by the user
		Error Request rejected

This table describes the Public Key Infrastructure (PKI) shared between the M262 Logic/Motion Controller and the SCEP server. It provides the folder list and their usage:

M262 File System Folders	Description
/usr/pki/scep/castore	Stores working certificate received from SCEP server.
/usr/pki/scep/tmp	Stores temporary files.
/usr/pki/scep/csr	Stores the signed certificate request.

⚠ WARNING

UNAUTHENTICATED ACCESS AND SUBSEQUENT UNAUTHORIZED MACHINE OPERATION

- Evaluate whether your environment or your machines are connected to your critical infrastructure and, if so, take appropriate steps in terms of prevention, based on Defense-in-Depth, before connecting the automation system to any network.
- Limit the number of devices connected to a network to the minimum necessary.
- Isolate your industrial network from other networks inside your company.
- Protect any network against unintended access by using firewalls, VPN, or other, proven security measures.
- Monitor activities within your systems.
- Prevent subject devices from direct access or direct link by unauthorized parties or unauthenticated actions.
- Prepare a recovery plan including backup of your system and process information.

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

Firewall Configuration

There are three ways to manage the controller firewall configuration:

- Static configuration
- Dynamic changes
- Application settings

Script files are used in the static configuration and for dynamic changes.

Static Configuration

The static configuration is loaded at the controller boot.

The controller firewall can be statically configured by managing a default script file located in the controller. The path to this file is `/usr/Cfg/FirewallDefault.cmd`.

NOTE: The file name is case sensitive.

Dynamic Changes

After the controller boot, the controller firewall configuration can be changed by the use of script files.

There are two ways to load these dynamic changes using:

- A physical SD card, page 144.
- A function block, page 144 in the application.

Application Settings

See Ethernet Configuration, page 116.

Dynamic Changes Procedure

Using an SD Card

This table describes the procedure to execute a firewall script from an SD card:

Step	Action
1	Create a valid firewall script, page 146. For example, name the firewall script <i>FirewallMaintenance.cmd</i> .
2	Load the firewall script on the SD card. For example, load the firewall script in the <i>usr/Cfg</i> folder.
3	In the file <i>Sys/Cmd/Script.cmd</i> , add a code line with the command <code>Firewall_install "/pathname/FileName"</code> For example, the code line is <code>Firewall_install "/sd0/usr/Cfg/FirewallMaintenance.cmd"</code> NOTE: The file name is case sensitive.
4	Insert the SD card on the controller.

Using a Function Block in the Application

This table describes the procedure to execute a firewall script from an application:

Step	Action
1	Create a valid firewall script, page 146. For example, name the firewall script <i>FirewallMaintenance.cmd</i> .
2	Load the firewall script in the controller memory. For example, load the firewall script in the <i>usr/Syslog</i> folder with FTP.
3	Use an <code>ExecuteScript</code> function block. For more information, refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide). For example, the [SCmd] input is <code>'Firewall_install "/usr/Syslog/FirewallMaintenance.cmd"'</code> NOTE: The file name is case sensitive.

Firewall Behavior

Introduction

The firewall configuration depends on the action done on the controller and the initial configuration state. There are five possible initial states:

- There is no default script file in the controller.
- A correct script file is present.
- An incorrect script file is present.
- There is no default script file and the application has configured the firewall.
- A dynamic script file configuration has already been executed.

No Default Script File

If...	Then ...
Boot of the controller	Firewall is not configured. No protection is activated.
Execute dynamic script file	Firewall is configured according to the dynamic script file.
Execute dynamic incorrect script file	Firewall is not configured. No protection is activated.
Download application	Firewall is configured according to the application settings.

Default Script File Present

If...	Then ...
Boot of the controller	Firewall is configured according to the default script file.
Execute dynamic script file	The whole configuration of the default script file is deleted. Firewall is configured according to the dynamic script file.
Execute dynamic incorrect script file	Firewall is configured according to the default script file. The dynamic script file is not taken into account.
Download application	The whole configuration of the application is ignored. Firewall is configured according to the default script file.

Incorrect Default Script File Present

If...	Then ...
Boot of the controller	Firewall is not configured. No protection is activated.
Execute dynamic script file	Firewall is configured according to the dynamic script file.
Download application	Firewall is configured according to the application settings.

Application Settings with No Default Script File

If...	Then ...
Boot of the controller	Firewall is configured according to the application settings.
Execute dynamic script file	The whole configuration of the application settings is deleted. Firewall is configured according to the dynamic script file.
Execute dynamic incorrect script file	Firewall is configured according to the application settings. The dynamic script file is not taken into account.
Download application	The whole configuration of the previous application is deleted. Firewall is configured according to the new application settings.

Execute Dynamic Script File Already Executed

If...	Then ...
Boot of the controller	Firewall is configured according to the dynamic script file configuration (see note).
Execute dynamic script file	The whole configuration of the previous dynamic script file is deleted. Firewall is configured according to the new dynamic script file.
Execute dynamic incorrect script file	Firewall is configured according to the previous dynamic script file configuration. The dynamic incorrect script file is not taken into account.

If...	Then ...
Download application	The whole configuration of the application is ignored. Firewall is configured according to the dynamic script file.
NOTE: If an SD card containing a cybersecurity script is plugged into the controller, booting is blocked. First remove the SD card to correctly boot the controller.	

Firewall Script Commands

Overview

This section describes how script files (default script files or dynamic script files) are written so that they can be executed during the booting of the controller or during a specific command triggered.

NOTE: The MAC layer rules are managed separately and have more priority over other packet filter rules.

Script File Syntax

The syntax of script files is described in [Creating a Script](#), page 231.

General Firewall Commands

The following commands are available to manage the Ethernet firewall of the M262 Logic/Motion Controller:

Command	Description
Firewall Enable	Blocks the frames from the Ethernet interfaces. If no specific IP address or port is authorized, it is not possible to communicate on the Ethernet interfaces. NOTE: By default, when the firewall is enabled, the frames are rejected.
Firewall Disable	Firewall rules are not applied. Frames are not blocked.
Firewall Ethx Default Allow ⁽¹⁾	Frames are accepted by the controller on interface Ethx.
Firewall Ethx Default Reject ⁽¹⁾	Frames are rejected by the controller on interface Ethx. NOTE: By default, if this line is not present, it corresponds to the command <code>Firewall Eth1 Default Reject</code> .
(1) Where Ethx = - Eth0: USB port - Eth1: Ethernet_1 - Eth2: Ethernet_2 - Eth3: TMSSES4 - Eth4: TMSSES4_1 - Eth5: TMSSES4_2	

Specific Firewall Commands

The following commands are available to configure firewall rules for specific ports and addresses:

Command	Range	Description
Firewall Ethx Allow IP $\bullet.\bullet.\bullet.\bullet^{(1)}$	$\bullet = 0\ldots255$	Frames from the specified IP address are allowed on all port numbers and port types.
Firewall Ethx Reject IP $\bullet.\bullet.\bullet.\bullet^{(1)}$	$\bullet = 0\ldots255$	Frames from the specified IP address are rejected on all port numbers and port types.
Firewall Ethx Allow IPs $\bullet.\bullet.\bullet.\bullet$ to $\bullet.\bullet.\bullet.\bullet^{(1)}$	$\bullet = 0\ldots255$	Frames from the IP addresses in the specified range are allowed for all port numbers and port types. NOTE: Rules with specific IP address range will be converted to CIDR format in controller while they are established. Example: "Firewall Eth2 allows IPs 192.168.100.66 to 192.168.100.99 on TCP port 44818" is separated into 7: 192.168.100.66/31 192.168.100.68/30 192.168.100.72/29 192.168.100.80/28 192.168.100.96/27 192.168.100.128/26 192.168.100.192/29 Using of entire subnet IP ranges avoids firewall rules saturation.
Firewall Eth1 Reject IPs $\bullet.\bullet.\bullet.\bullet$ to $\bullet.\bullet.\bullet.\bullet^{(1)}$	$\bullet = 0\ldots255$	Frames from the IP addresses in the specified range are rejected for all port numbers and port types.
Firewall Eth1 Allow port_type port Y ⁽¹⁾	Y = (destination port numbers, page 151)	Frames with the specified destination port number are allowed.
Firewall Eth1 Reject port_type port Y ⁽¹⁾	Y = (destination port numbers, page 151)	Frames with the specified destination port number are rejected. NOTE: When IP forwarding is activated, rules with reject port only filter frames with current controller as destination. They are not applied for the frames routed by the current controller.
Firewall Eth1 Allow port_type ports Y1 to Y2 ⁽¹⁾	Y = (destination port numbers, page 151)	Frames with a destination port number in the specified range are allowed.
Firewall Eth1 Reject port_type ports Y1 to Y2 ⁽¹⁾	Y = (destination port numbers, page 151)	Frames with a destination port number in the specified range are rejected.
Firewall Eth1 Allow IP $\bullet.\bullet.\bullet.\bullet$ on port_type port Y ⁽¹⁾	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from the specified IP address and with the specified destination port number are allowed.
Firewall Ethx Reject IP $\bullet.\bullet.\bullet.\bullet$ on port_type port Y	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from the specified IP address and with the specified destination port number are rejected.
Firewall Ethx Allow IP $\bullet.\bullet.\bullet.\bullet$ on port_type ports Y1 to Y2	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from the specified IP address and with a destination port number in the specified range are allowed.
Firewall Ethx Reject IP $\bullet.\bullet.\bullet.\bullet$ on port_type ports Y1 to Y2	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from the specified IP address and with a destination port number in the specified range are rejected.
Firewall Ethx Allow IPs $\bullet1.\bullet1.\bullet1.\bullet1$ to $\bullet2.\bullet2.\bullet2.\bullet2$ on port_type port Y	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from an IP address in the specified range and with the specified destination port number are allowed.
Firewall Ethx Reject IPs $\bullet1.\bullet1.\bullet1.\bullet1$ to $\bullet2.\bullet2.\bullet2.\bullet2$ on port_type port Y ⁽¹⁾	$\bullet = 0\ldots255$ Y = (destination port numbers, page 151)	Frames from an IP address in the specified range and with the specified destination port number are rejected.

Command	Range	Description
Firewall Ethx Allow IPs •1.•1.•1.•1 to •2.•2.•2.•2 on port_type ports Y1 to Y2 ⁽¹⁾	• = 0...255 Y = (destination port numbers, page 151)	Frames from an IP address in the specified range and with a destination port number in the specified range are allowed.
Firewall Ethx Reject IPs •1.•1. •1.•1 to •2.•2.•2. •2 on port_type ports Y1 to Y2 ⁽¹⁾	• = 0...255 Y = (destination port numbers, page 151)	Frames from an IP address in the specified range and with a destination port number in the specified range are rejected.
Firewall Ethx Allow MAC ••:••:••:••:••:••: •• ⁽¹⁾	• = 0...F	Frames from the specified MAC address ••:••:~••:~••:~••:~•• are allowed. NOTE: When the rules to allow the MAC address are applied, only the listed MAC addresses can communicate with the controller, even if other rules are allowed.
Firewall Ethx Reject MAC ••:~••:~••:~••:~••: ••:~••:~••:~••:~•• ⁽¹⁾	• = 0...F	Frames with the specified MAC address ••:~••:~••:~••:~•• are rejected.
Firewall Ethx ⁽¹⁾ Established to port_type port Y	Y = 0...65535	Frames established from the controller with the protocols TCP/UDP to the specified destination port number are allowed.
(1) Where Ethx = - Eth0: USB port - Eth1: Ethernet_1 - Eth2: Ethernet_2 - Eth3: TMSSES4 - Eth4: TMSSES4_1 - Eth5: TMSSES4_2		

NOTE: When IP Forwarding is activated, rules with **Reject port** only filter frames with current controller as destination. They are not applied for the frames routed by the current controller.

Script Example

```
; Enable FireWall. All frames are rejected;
FireWall Enable;

; Allow frames on Eth1
FireWall Eth1 Default Allow;

; Block all Modbus Requests on all IP address
Firewall Eth1 Reject tcp port 502;

; Reject frames on Eth2
FireWall Eth2 Default Reject;

; Allow FTP active connection for IP address 85.16.0.17
FireWall Eth2 Allow IP 85.16.0.17 on tcp ports 20 to 21;
```

NOTE: IP addresses are converted to CIDR format.

For example:

"FireWall Eth2 Allow IPs 192.168.100.66 to 192.168.100.99 on tcp port 44818;" is separated into 7:

- 192.168.100.66/31
- 192.168.100.68/30
- 192.168.100.72/29
- 192.168.100.80/28
- 192.168.100.96/27
- 192.168.100.128/26
- 192.168.100.192/29

To prevent a firewall error, use the entire subnet configuration.

The following is an example of a firewall in white list mode. The example has all communication blocked by default and allows only the necessary services.

NOTE: This example is designed to show most of the commands available with the firewall. It should be adapted to your configuration and tested before implementation.

Commands	Comments
Firewall Enable	; Enable the firewall.
Eth1 Configuration	
Firewall Eth1 Default Reject	; Reject all frames on interface ETH1. ; In this example, ETH1 is connected to the Industrial Ethernet devices network and therefore can be relatively trusted.
Firewall Eth1 Allow TCP port 502	; Allow Modbus TCP server on interface ETH1. ; There is no authentication on Modbus so this should be allowed only on trusted networks.
Firewall Eth1 Established to TCP port 502	; Allow replies to communication established by the controller to TCP port 502. ; This is necessary when using PlcCommunication library to communicate using Modbus TCP protocol.
Firewall Eth1 Allow UDP port 2222	; Allow ETHIP scanner implicit exchanges replies to UDP port 2222 (ETHIP) on interface ETH1.
Firewall Eth1 Established to TCP port 44818	; Allow replies to communication established by the controller to TCP port 44818 (ETHIP) on interface ETH1. ; The last 2 commands allow the EtheNetIP Scanner to communicate with the industrial ethernet devices.
Eth2 Configuration	
Firewall Eth2 Default Reject	; Reject all frames on interface ETH2. This interface is connected to a network used mainly for commissioning.
Firewall Eth2 Allow TCP port 4840	; Allow OPC UA server on interface ETH2.
Firewall Eth2 Allow TCP port 443	; Allow Web server (https) on interface ETH2.
Firewall Eth2 Allow TCP port 8089	; Allow WebVisualisation (https) on interface ETH2.
Firewall Eth2 Allow TCP ports 20 to 21	; Allow FTP in active mode on interface ETH2.
Firewall Eth2 Allow IP 192.168.1.1 on UDP ports 27126 to 27127	; Allow the IP of the commissioning PC to discover and configure the IP address of the controller. ; This should be allowed only on a trusted network as IP can be changed even if the User Rights are configured.
Firewall Eth2 Allow IPs 192.168.1.1 to 192.168.1.2 on UDP port 1740	; Allow the IP of the commissioning PC and an HMI to communicate with the controller using Machine Expert protocol.
Firewall Eth2 Allow TCP port 11740	; Allow Fast TCP on interface ETH2. This allow to connect to the controller using TCP.
Firewall Eth2 Allow TCP port 2222	; Allow implicit communication with UDP port 2222 (ETHIP) on interface ETH2.
Firewall Eth2 Allow TCP port 44818	; Allow explicit communication to TCP port 44818 (ETHIP) on interface ETH2. The last 2 commands allow to use the controller as an EtherNetIP Adapter.
Firewall Eth2 Allow MAC 4C:CC:6A:A1:09:C8	; Allow the MAC address of the HMI.
Firewall Eth2 Allow MAC 00:0C:29:92:43:A8	; Allow the MAC address of the commissioning PC. Only the MAC addresses allowed can communicate with the controller.
Eth3 Configuration TMSSES4	
Firewall Eth3 Default Reject	; Reject frames on TMSSES4. This interface is connected to the Plant network and can access the web. It should be considered as untrusted.

Firewall Eth3 Established to TCP port 443

; Allow http client (for example to connect to Machine Advisor) on interface TMSES4.

Firewall Eth3 Allow TCP port 11740

; Allow Fast TCP on interface TMSES4. This allows to connect to the controller remotely. It must not be allowed unless User Rights are activated on the controller.

NOTE: Characters are limited to 200 per line, including comments.

Ports Used

Protocol	Destination Port Numbers
Machine Expert	UDP 1740, 1741, 1742, 1743 TCP 11740
FTP	TCP 21, 20
HTTP	TCP 80
HTTPS	TCP 443
Modbus	TCP 502
OPC UA	TCP 4840
Machine Expert Discovery	UDP 27126, 27127
Bonjour Discovery Protocol	UDP 5353
Web Services Dynamic Discovery	UDP 3702 TCP 5357
SNMP	UDP 161, 162
NVL	UDP Default value: 1202
EtherNet/IP	UDP 2222 TCP 44818
WebVisualisation	HTTP 8080 HTTPS 8089
TFTP	UDP 69 (used for FDR server only)
SafeLogger	UDP 35021, 45000
Machine Assistant	UDP 45001...45004

Industrial Ethernet

Introduction

This chapter describes how to add and configure the Industrial Ethernet.

Industrial Ethernet Presentation

Overview

Industrial Ethernet is the term used to represent the industrial protocols that use the standard Ethernet physical layer and standard Ethernet protocols.

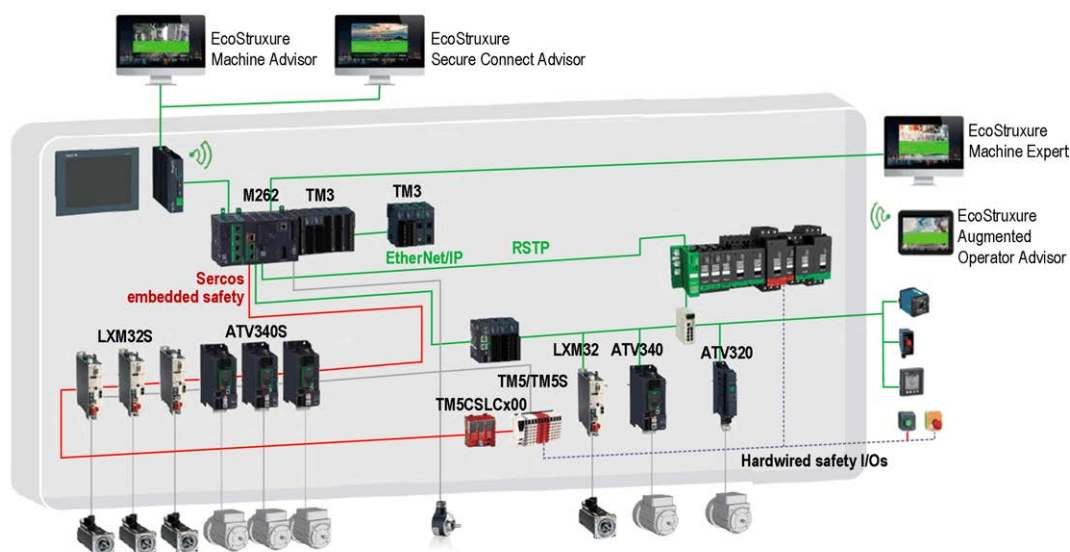
On an Industrial Ethernet network, you can connect:

- industrial devices (industrial protocols)
- non-industrial devices (other Ethernet protocols)

For more information, refer to Industrial Ethernet Overview User Guide (see EcoStruxure Machine Expert Industrial Ethernet Overview, User Guide).

Industrial Ethernet Architecture

This figure presents a typical Industrial Ethernet architecture:



This architecture is configurable with EcoStruxure Machine Expert.

Industrial Ethernet Description

M262 Logic/Motion Controller	
Features	Description
Topology	Daisy chain and Star via switches
Bandwidth	10/100 Mbit/s for Ethernet 1 port 10/100/1000 Mbit/s for Ethernet 2 port
EtherNet/IP Scanner	
Performance	Up to 64 EtherNet/IP target devices managed by the controller, monitored within a timeslot of: 40 ms on TM262L01MESE8T, TM262L10MESE8T, TM262M05MESS8T and TM262M15MESS8T 20 ms on TM262L20MESE8T, TM262M25MESS8T and TM262M35MESS8T
Number of connections	0...64
Number of input words	0...15360
Number of output words	0...15360
I/O communications	EtherNet/IP Scanner service Function block for configuration and data transfer
	Originator/Target
Modbus TCP IOScanner	
Performance	Up to 64 Modbus TCP slave devices managed by the controller, monitored within a timeslot of: 20 ms on TM262L01MESE8T, TM262L10MESE8T, TM262M05MESS8T and TM262M15MESS8T 10 ms on TM262L20MESE8T, TM262M25MESS8T and TM262M35MESS8T
Number of connections	0...6 on Ethernet_1 port of TM262M• 0...64 on other ports of TM262M• and on TM262L•
Number of input words	0...8000
Number of output words	0...8000
I/O communications	Modbus TCP IOScanner service Function block for data transfer
	Master/Slave
Sercos	
Performance	Refer to Performance Overview, page 19.

M262 Logic/Motion Controller	
Features	Description
Other services	FDT/DTM/EDS management
	FDR (Fast Device Replacement)
	DHCP server
	Security management (refer to Security Parameters, page 118 and Firewall Configuration, page 142)
	Modbus TCP server
	Modbus TCP client
	EtherNet/IP adapter (controller as a target on EtherNet/IP)
	EtherNet/IP Originator
	Modbus TCP server (controller as a slave on Modbus TCP)
	Web server, page 122
	FTP server, page 120
	NTP, page 67
	OPC UA, page 201
	SNMP, page 121
	IEC VAR ACCESS
Additional features	You can mix the Ethernet/IP and Modbus TCP server devices: 96 (64 EIP and 32 TCP) on TM262L01MESE8T, TM262L10MESE8T, TM262M05MESS8T and TM262M15MESS8T 128 (64 EIP and 64 TCP) on TM262L20MESE8T, TM262M25MESS8T and TM262M35MESS8T. Devices can be directly accessed for configuration, monitoring, and management purposes. Network transparency between control network and device network (controller can be used as a gateway). NOTE: Using the controller as a gateway can impact the performance of the controller.
Single Wire Architecture, page 180	Allows up to 6 Ethernet devices (EtherNet/IP, TCP/IP, and so on) to be added to the end of a cable containing Sercos devices. The last Sercos device acts as a gateway. No additional gateways or switches are required. The Ethernet frames are embedded within the Sercos frames.

EtherNet/IP Overview

EtherNet/IP is the implementation of the CIP protocol over standard Ethernet.

The EtherNet/IP protocol uses an Originator/Target architecture for data exchange.

Originators are devices that initiate data exchanges with Target devices on the network. This applies to both I/O communications and service messaging. This is the equivalent of the role of a client in a Modbus network.

Targets are devices that respond to data requests generated by Originators. This applies to both I/O communications and service messaging. This is the equivalent of the role of a server in a Modbus network.

EtherNet/IP Adapter is an end-device in an EtherNet/IP network. I/O blocks and drives can be EtherNet/IP Adapter devices.

The communication between an EtherNet/IP Originator and Target is accomplished using an EtherNet/IP connection.

Modbus TCP Overview

The Modbus TCP protocol uses a Client/Server architecture for data exchange.

The Modbus TCP explicit (non-cyclic) data exchanges are managed by the application.

Modbus TCP implicit (cyclic) data exchanges are managed by the Modbus TCP IOScanner. The Modbus TCP IOScanner is a service based on Ethernet that polls slave devices continuously to exchange data, status, and diagnostic information. This process monitors inputs and controls outputs of slave devices.

Clients are devices that initiate data exchange with other devices on the network. This applies to both I/O communications and service messaging.

Servers are devices that address any data requests generated by a Client. This applies to both I/O communications and service messaging.

The communication between the Modbus TCP IOScanner and the slave device is accomplished using Modbus TCP channels.

Sercos Overview

For more information on Sercos standard and configuration, refer to [Overview of the Sercos Standard](#), page 179.

Adding the Protocol Manager

The protocol manager must be present on the **Ethernet_1 (ETH1)** and **Ethernet_2 (ETH2)** nodes of the **Devices tree** to activate these functions and services:

- EtherNet/IP Scanner
- Generic TCP/UDP Manager
- Modbus TCP IO Scanner

When a Protocol manager is defined on an interface, this interface address must be **Fixed**. The post-configuration defined for this interface is not applied, if any.

The protocol manager is available by default under the **Ethernet_1 (ETH1)** and **Ethernet_2 (ETH2)** nodes. It is automatically added when a slave device is added on the **Ethernet_1 (ETH1)** or **Ethernet_2 (ETH2)** node.

To manually add the a function or service to the **Ethernet_1 (ETH1)** or **Ethernet_2 (ETH2)**, select the protocol manager in the **Hardware Catalog** and drag and drop it on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see [EcoStruxure Machine Expert, Programming Guide](#))
- Using the Contextual Menu or Plus Button (see [EcoStruxure Machine Expert, Programming Guide](#))

Adding the Sercos Master

The Sercos fieldbus must be present on the **Ethernet_1 (ETH1)** to activate the Sercos Master. It is automatically added when a slave device is added on the **Ethernet_1 (ETH1)** node.

To manually add **Sercos Master** to the **Ethernet_1 (ETH1)**, select **Sercos Master** in the **Hardware Catalog** and drag and drop it on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

DHCP Server

Overview

The DHCP server offers addresses to the devices connected on the Ethernet network. The DHCP server only delivers static addresses. A unique identified slave gets a unique address. DHCP slave devices are identified either by their MAC address or their DHCP device name. The DHCP server configuration table defines the relation between addresses and identified slave devices.

The DHCP server addresses are given with an infinite lease time. There is no need for the slave devices to refresh the leased IP address.

The synthesis of the DHCP server configuration is displayed on the **Ethernet Services** tab, page 64.

For more information, refer to IP Addressing Methods (see EcoStruxure Machine Expert Modbus TCP, User Guide).

Fast Device Replacement

Overview

The Fast Device Replacement (FDR) helps facilitate replacing and reconfiguring a network device. This function is available on the Ethernet 1 and Ethernet 2 ports of the M262 Logic/Motion Controller.

For more information, refer to Slave Device Replacement with FDR (see EcoStruxure Machine Expert Modbus TCP, User Guide).

Controller as a Target Device on EtherNet/IP

Introduction

This section describes the configuration of the M262 Logic/Motion Controller as an EtherNet/IP target device.

For further information about EtherNet/IP, refer to the www.odva.org website.

EtherNet/IP Target Configuration

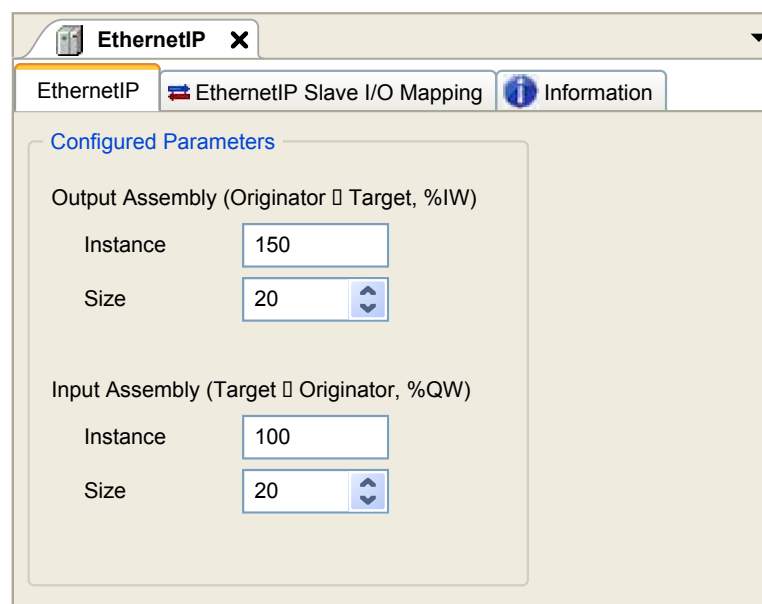
To configure your M262 Logic/Motion Controller as an EtherNet/IP target device, you must:

Step	Action
1	In the Hardware Catalog , select Devices & Modules > Communication > Ethernet IP > EtherNetIP .
2	Drag and drop it to the Devices tree on one of the highlighted nodes. For more information on adding a device to your project, refer to: • Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide) • Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

EtherNet/IP Parameters Configuration

To configure the EtherNet/IP parameters, double-click an Ethernet port > **EthernetIP** in the **Devices tree**.

This dialog box is displayed:



The EtherNet/IP configuration parameters are defined as:

- **Instance:**
Number referencing the input or output Assembly.
- **Size:**
Number of channels of an input or output Assembly.
The memory size of each channel is 2 bytes that stores the value of an %IWx or %QWx object, where x is the channel number.
For example, if the **Size** of the **Output Assembly** is 20, it represents that there are 20 input channels (IW0...IW19) addressing %IWy...%IW(y+20-1), where y is the first available channel for the Assembly.

Element		Admissible Controller Range	EcoStruxure Machine Expert Default Value
Output Assembly	Instance	150...189	150
	Size	2...120	20
Input Assembly	Instance	100...149	100
	Size	2...120	20

EDS File Generation

You can generate an EDS file to configure EtherNet/IP cyclic data exchanges.

To generate the EDS file:

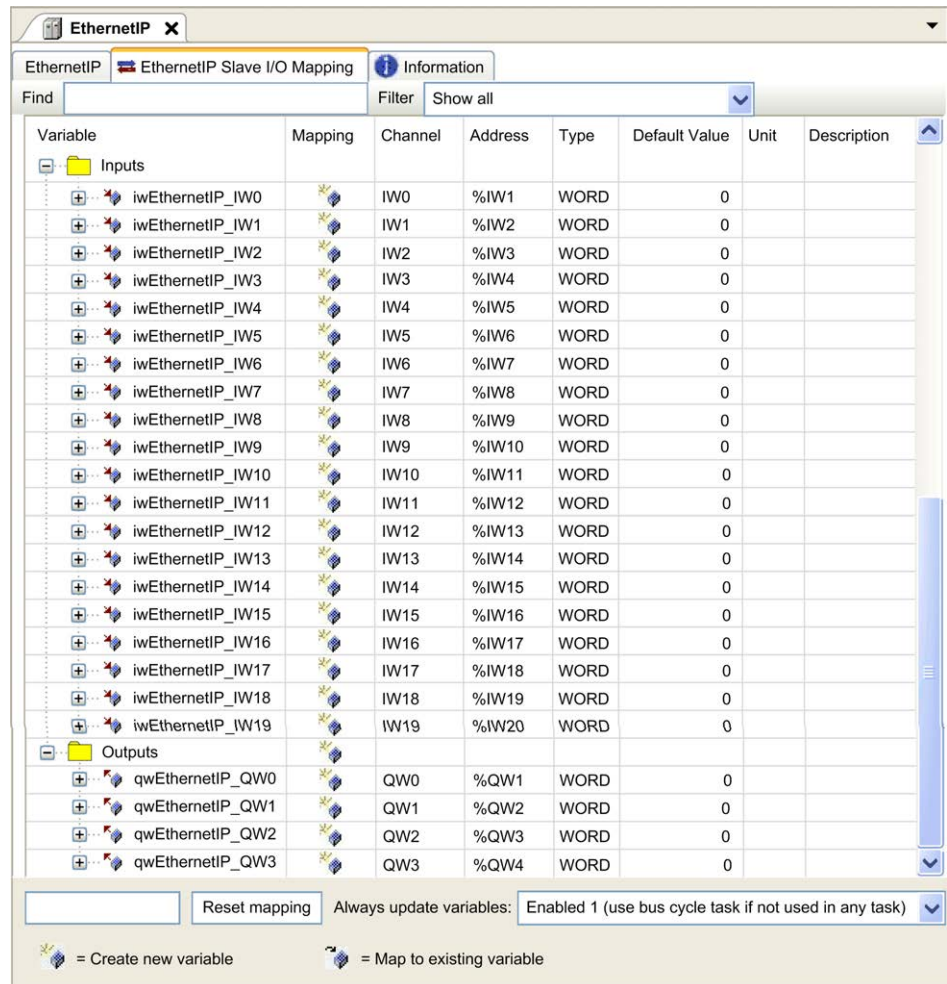
Step	Action
1	In the Devices tree , right-click the EthernetIP node and choose the Export as EDS command from the contextual menu.
2	Modify the default file name and location as required.
3	Click Save .

NOTE: The **Major Revision** and **Minor Revision** objects of the EDS file, defined in the file, are used to ensure uniqueness of the EDS file. The values of these objects do not reflect the actual controller revision level.

A generic EDS file for the M262 Logic/Motion Controller is also available on the Schneider Electric website. You must adapt this file to your application by editing it and defining the required Assembly instances and sizes.

EthernetIP Slave I/O Mapping Tab

Variables can be defined and named in the **EthernetIP Slave I/O Mapping** tab. Additional information such as topological addressing is also provided in this tab.



The table below describes the **EthernetIP Slave I/O Mapping** configuration:

Channel		Type	Default Value	Description
Input	IW0	WORD	-	Command word of controller outputs (%QW)
	IWxxx			
Output	QW0	WORD	-	State of controller inputs (%IW)
	QWxxx			

The number of words depends on the size parameter configured in EtherNet/IP Target Configuration, page 156.

Output means OUTPUT from Originator controller (= %IW for the controller).

Input means INPUT from Originator controller (= %QW for the controller).

Connections on EtherNet/IP

To access a target device, an Originator opens a connection which can include several sessions that send requests.

One explicit connection uses one session (a session is a TCP or UDP connection).

One I/O connection uses two sessions.

The following table shows the EtherNet/IP connections limitations:

Characteristic	Maximum
Explicit connections	8 (Class 3)
I/O connections	1 (Class 1)
Connections	8
Sessions	16
Simultaneous requests	32

NOTE: The M262 Logic/Motion Controller supports cyclic connections only. If an Originator opens a connection using a change of state as a trigger, packets are sent at the RPI rate.

Profile

The controller supports the following objects:

Object class	Class ID (hex)	Cat.	Number of Instances	Effect on Interface Behavior
Identity Object, page 160	01	1	1	Supports the reset service
Message Router Object, page 163	02	1	1	Explicit message connection
Assembly Object, page 164	04	2	2	Defines I/O data format
Connection Manager Object, page 166	06	–	1	–
TCP/IP Interface Object, page 167	F5	1	1	TCP/IP configuration
Ethernet Link Object, page 168	F6	1	1	Counter and status information
Interface Diagnostic Object, page 169	350	1	1	–
IOScanner Diagnostic Object, page 171	351	1	1	–
Connection Diagnostic Object, page 172	352	1	1	–
Explicit Connection Diagnostic Object, page 174	353	1	1	–
Explicit Connections Diagnostic List Object, page 174	354	1	1	–

Identity Object (Class ID = 01 hex)

The following table describes the class attributes of the Identity Object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Implementation revision of the Identity Object.
2	Get	Max Instance	UINT	01	The largest instance number.
6	Get	Max Class Attribute	UINT	01	The largest class attributes value.
7	Get	Max Instance Attribute	UINT	07	The largest instance attributes value.

The following table describes the Class Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all class attributes.
0E	Get Attribute Single	Returns the value of the specified attribute.

The following table describes the Instance Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all class attributes.
05	Reset ⁽¹⁾	Initializes EtherNet/IP component (controller reboot).
0E	Get Attribute Single	Returns the value of the specified attribute.

(1) Reset Service description:

When the Identity Object receives a Reset request, it:

- determines whether it can provide the type of reset requested
- responds to the request
- attempts to perform the type of reset requested

NOTE: The Reset Service only applies if the corresponding parameter has been activated in post configuration file. See [Post Configuration File Example](#), page 218.

The Reset common service has one specific parameter, Type of Reset (USINT), with the following values:

Value	Type of Reset
0	Reboots the controller NOTE: This is the default value if this parameter is omitted.
1	Not supported
2	Not supported
3...99	Reserved
100...199	Vendor specific
200...255	Reserved

The following table describes the Instance attributes:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Vendor ID	UINT	F3	Schneider Automation ID
2	Get	Device type	UINT	0E	Controller
3	Get	Product code	UINT	4102	Controller product code
4	Get	Revision	Struct of USINT, USINT	–	Product revision number of the controller ⁽¹⁾ . Equivalent to the 2 low bytes of the controller version.
5	Get	Status	WORD	–	Status word ⁽²⁾
6	Get	Serial number	UDINT	–	Serial number of the controller: XX + 3 LSB of MAC address.
7	Get	Product name	Struct of USINT, STRING	–	–

⁽¹⁾ Mapped in a WORD:

- MSB: minor revision (second USINT)
- LSB: major revision (first USINT)

Example: 0205 hex means revision V5.2.

⁽²⁾ Status word (Attribute 5):

Bit	Name	Description
0	Owned	Unused.
1	Reserved	–
2	Configured	TRUE indicates the device application has been reconfigured.
3	Reserved	–
4...7	Extended Device Status	• 0: Self-testing or undetermined • 1: Firmware update in progress • 2: At least one invalid I/O connection detected • 3: No I/O connections established • 4: Non-volatile configuration invalid • 5: Unrecoverable error detected • 6: At least one I/O connection in RUNNING state • 7: At least one I/O connection established, all in idle mode • 8: Reserved • 9...15: Unused
8	Minor Recoverable Fault	TRUE indicates the device detected an error, which, under most circumstances, is recoverable. This type of event does not lead to a change in the device state.
9	Minor Unrecoverable Fault	TRUE indicates the device detected an error, which, under most circumstances, is unrecoverable. This type of event does not lead to a change in the device state.
10	Major Recoverable Fault	TRUE indicates the device detected an error, which requires the device to report an exception and enter into the HALT state. This type of event leads to a change in the device state, but, under most circumstances, is recoverable.
11	Major Unrecoverable Fault	TRUE indicates the device detected an error, which requires the device to report an exception and enter into the HALT state. This type of event leads to a change in the device state, but, under most circumstances, is not recoverable.
12...15	Reserved	–

Message Router Object (Class ID = 02 hex)

The following table describes the class attributes of the Message Router object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Implementation revision number of the Message Router Object.
2	Get	Max Instance	UINT	02	The largest instance number.
3	Get	Number of Instance	UINT	01	The number of object instances.
4	Get	Optional Instance Attribute List	Struct of UINT, UINT []	02	The first 2 bytes contain the number of optional instance attributes. Each following pair of bytes represents the number of other optional instance attributes (from 100 to 119).
5	Get	Optional Service List	UINT	0A	The number and list of any implemented optional services attribute (0: no optional services implemented).
6	Get	Max Class Attribute	UINT	07	The largest class attributes value.
7	Get	Max Instance Attribute	UINT	02	The largest instance attributes value.

The following table describes the Class services:

Service Code (hex)	Name	Description
01	Get_Attribute_All	Returns the value of all class attributes.
0E	Get_Attribute_Single	Returns the value of the specified attribute.

The following table describes the Instance services:

Service Code (hex)	Name	Description
01	Get_Attribute_All	Returns the value of all class attributes.
0E	Get_Attribute_Single	Returns the value of the specified attribute.

The following table describes the Instance attributes:

Attribute ID (hex)	Access	Name	Data Type	Value	Description
1	Get	Implemented Object List	Struct of UINT, UINT []	–	Implemented Object list. The first 2 bytes contain the number of implemented objects. Each 2 bytes that follow represents another implemented class number. This list contains the following objects: • Identity • Message Router • Assembly • Connection Manager • Parameter • File Object • Modbus • Port • TCP/IP • Ethernet Link
2	Get	Number available	UINT	512	Maximum number of concurrent CIP (Class 1 or Class 3) connections supported.
3	Get	Number active	UINT	–	Numbers of connections currently used by system component.

Assembly Object (Class ID = 04 hex)

The following table describes the class attributes of the Assembly object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	02	Implementation revision of the Assembly Object.
2	Get	Max Instance	UINT	BE	The largest instance number.
3	Get	Number of Instances	UINT	03	The number of object instances.
4	Get	Optional Instance Attribute List	Struct of: UINT UINT []	01 04	The first 2 bytes contain the number of optional instance attributes. Each following pair of bytes represents the number of other optional instance attributes.
5	Get	Optional Service List	UINT	Not supported	The number and list of any implemented optional services attribute (0: no optional services implemented).
6	Get	Max Class Attribute	UINT	07	The largest class attributes value.
7	Get	Max Instance Attribute	UINT	04	The largest instance attributes value.

The following table describes the Class Services:

Service Code (hex)	Name	Description
0E	Get Attribute Single	Returns the value of the specified attribute.

The following table describes the Instance Services:

Service Code (hex)	Name	Description
0E	Get Attribute Single	Returns the value of the specified attribute.
10	Set Attribute Single	Modifies the value of the specified attribute.

Instances Supported

Output means OUTPUT from Originator controller (= %IW for the controller).

Input means INPUT from Originator controller (= %QW for the controller).

The controller supports 2 Assemblies:

Name	Instance	Data Size
Controller Output (% IW)	Configurable: must be between 100 and 149	2...40 words
Controller Input (% QW)	Configurable: must be between 150 and 189	2...40 words

NOTE: The Assembly object binds together the attributes of multiple objects so that information to or from each object can be communicated over a single connection. Assembly objects are static.

The Assemblies in use can be modified through the parameter access of the network configuration tool (RSNetWorx). The controller needs to recycle power to register a new Assembly assignment.

The following table describes the Instance attributes:

Attribute ID (hex)	Access	Name	Data Type	Value	Description
3	Get/Set	Instance Data	ARRAY of Byte	–	Data Set service only available for Controller output.
4	Get	Instance Data Size	UINT	4...80	Size of data in byte.

Access from a EtherNet/IP Scanner

When a EtherNet/IP Scanner needs to exchange assemblies with a M262 Logic/Motion Controller, it uses the following access parameters (*Connection path*):

- Class 4
- Instance xx where xx is the instance value (example: 2464 hex = instance 100).
- Attribute 3

In addition, a configuration assembly must be defined in the Originator.

For example: Class 4, Instance 3, Attribute 3, the resulting *Connection Path* is:

- 2004 hex
- 2403 hex
- 2c<xx> hex

Connection Manager Object (Class ID = 06 hex)

The following table describes the class attributes of the Assembly Object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Implementation revision of the Connection Manager Object.
2	Get	Max Instance	UINT	01	The largest instance number.
3	Get	Number of Instances	UINT	01	The number of object instances.
4	Get	Optional Instance Attribute List	Struct of: UINT UINT []	–	The number and list of the optional attributes. The first word contains the number of attributes to follow and each following word contains another attribute code. Following optional attributes include: - total number of incoming connection open requests - the number of requests rejected due to non-conforming format of the Forward Open - the number of requests rejected because of insufficient resources - the number of requests rejected due to parameter value sent with the Forward Open - the number of Forward Close requests received - the number of Forward Close requests with an invalid format - the number of Forward Close requests that could not be matched to an active connection - the number of connections that have timed out because the other side stopped producing, or a network disconnection occurred
6	Get	Max Class Attribute	UINT	07	The largest class attributes value.
7	Get	Max Instance Attribute	UINT	08	The largest instance attributes value.

The following table describes the Class Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all class attributes.
0E	Get Attribute Single	Returns the value of the specified attribute.

The following table describes the Instance Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all instance attributes.
0E	Get Attribute Single	Returns the value of the specified attribute.
4E	Forward Close	Closes an existing connection.
52	Unconnected Send	Sends a multi-hop unconnected request.
54	Forward Open	Opens a new connection.

The following table describes the Instance attributes:

Attribute ID (hex)	Access	Name	Data Type	Value	Description
1	Get	Open Requests	UINT	–	Number of Forward Open service requests received.
2	Get	Open Format Rejects	UINT	–	Number of Forward Open service requests which were rejected due to invalid format.
3	Get	Open Resource Rejects	ARRAY of Byte	–	Number of Forward Open service requests which were rejected due to lack of resources.
4	Get	Open Other Rejects	UINT	–	Number of Forward Open service requests which were rejected for reasons other than invalid format or lack of resources.
5	Get	Close Requests	UINT	–	Number of Forward Close service requests received.
6	Get	Close Format Requests	UINT	–	Number of Forward Close service requests which were rejected due to invalid format.
7	Get	Close Other Requests	UINT	–	Number of Forward Close service requests which were rejected for reasons other than invalid format.
8	Get	Connection Timeouts	UINT	–	Total number of connection timeouts that have occurred in connections controlled by this Connection Manager.

TCP/IP Interface Object (Class ID = F5 hex)

This object maintains link specific counters and status information for an Ethernet 802.3 communications interface.

The following table describes the class attributes of the TCP/IP Interface Object:

Attribute ID (hex)	Access	Name	Data Type	Value	Details
1	Get	Revision	UINT	4	Implementation revision of the TCP/IP Interface Object.
2	Get	Max Instance	UINT	2	The largest instance number.
3	Get	Number of Instances	UINT	2	The number of object instances.

The following table describes the Class Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all class attributes.
0E	Get Attribute Single	Returns the value of the specified attribute.

Instance Codes

Only instance 1 is supported.

The following table describes the Instance Services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all instance attributes.
0E	Get Attribute Single	Returns the value of the specified instance attribute.

The following table describes the Instance Attributes:

Attribute ID (hex)	Access	Name	Data Type	Value	Description
1	Get	Status	DWORD	Bit level	0: The interface configuration attribute has not been configured. 1: The interface configuration contains a valid configuration. 2...15: Reserved.
2	Get	Configuration Capability	DWORD	Bit level	0: BOOTP Client 2: DHCP Client 5: Configurable in EcoStruxure Machine Expert All other bits are reserved and set to 0.
3	Get	Configuration	DWORD	Bit level	0: The interface configuration is valid. 1: The interface configuration is obtained with BOOTP. 2: The interface configuration is obtained with DHCP. 3: reserved All other bits are reserved and set to 0.
4	Get	Physical Link	UINT	Path size	Number of 16 bits word in the element Path.
			Padded EPATH	Path	Logical segments identifying the physical link object. The path is restricted to one logical class segment and one logical instance segment. The maximum size is 12 bytes.
5	Get	Interface configuration	UDINT	IP Address	–
			UDINT	Network Mask	–
			UDINT	Gateway Address	–
			UDINT	Primary Name	–
			UDINT	Secondary Name	0: no secondary name server address has been configured.
			STRING	Default Domain Name	0: no Domain Name is configured.
6	Get	Host Name	STRING	–	ASCII characters. 0: no host name is configured.

Ethernet Link Object (Class ID = F6 hex)

This object provides the mechanism to configure a TCP/IP network interface device.

The following table describes the class attributes of the Ethernet Link object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	4	Implementation revision of the Ethernet Link Object.
2	Get	Max Instance	UINT	255	The largest instance number.
3	Get	Number of Instances	UINT	4	The number of object instances.

The following table describes the class services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all class attributes.
0E	Get Attribute Single	Returns the value of the specified attribute.

Instance Codes

Only instance 1 is supported.

The following table describes the instance services:

Service Code (hex)	Name	Description
01	Get Attribute All	Returns the value of all instance attributes.
0E	Get Attribute Single	Returns the value of the specified instance attribute.

The following table describes the instance attributes:

Attribute ID (hex)	Access	Name	Data Type	Value	Description
1	Get	Interface Speed	UDINT	–	Speed in Mbit/s (10 or 100)
2	Get	Interface Flags	DWORD	Bit level	0: link status 1: half/full duplex 2...4: negotiation status 5: manual setting / requires reset 6: local hardware error detected All other bits are reserved and set to 0.
3	Get	Physical Address	ARRAY of 6 USINT	–	This array contains the MAC address of the product. Format: XX-XX-XX-XX-XX-XX

EtherNet/IP Interface Diagnostic Object (Class ID = 350 hex)

The following table describes the class attributes of the EtherNet/IP Interface Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Increased by 1 on each new update of the object.
2	Get	Max Instance	UINT	01	Maximum instance number of the object.

The following table describes the instance attributes of the EtherNet/IP Interface Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Details
1	Get	Protocols supported	UINT	Protocol(s) supported (0=not supported, 1=supported): - Bit 0: EtherNet/IP - Bit 1: Modbus TCP - Bits 2...15: Reserved, 0
2	Get	Connection Diag	STRUCT of	
		Max CIP IO Connections opened	UINT	Maximum number of CIP I/O connections opened.
		Current CIP IO Connections	UINT	Number of CIP I/O connections currently opened.
		Max CIP Explicit Connections opened	UINT	Maximum number of CIP explicit connections opened.
		Current CIP Explicit Connections	UINT	Number of CIP explicit connections currently opened
		CIP Connections Opening Errors	UINT	Incremented on each unsuccessful attempt to open a CIP connection.
		CIP Connections Timeout Errors	UINT	Incremented when a CIP connection times out.
		Max EIP TCP Connections opened	UINT	Maximum number of TCP connections opened and used for EtherNet/IP communications.
		Current EIP TCP Connections	UINT	Number of TCP connections currently open and being used for EtherNet/IP communications.
3	Get Clear	IO Messaging Diag	STRUCT of	
		IO Production Counter	UDINT	Incremented each time a Class 0/1 CIP message is sent.
		IO Consumption Counter	UDINT	Incremented each time a Class 0/1 CIP message is received.
		IO Production Send Errors Counter	UINT	Incremented each Time a Class 0/1 message is not sent.
		IO Consumption Receive Errors Counter	UINT	Incremented each time a consumption is received that contains an error.
4	Get Clear	Explicit Messaging Diag	STRUCT of	
		Class3 Msg Send Counter	UDINT	Incremented each time a Class 3 CIP message is sent.
		Class3 Msg Receive Counter	UDINT	Incremented each time a Class 3 CIP message is received.
		UCMM Msg Send Counter	UDINT	Incremented each time a UCMM message is sent.
		UCMM Msg Receive Counter	UDINT	Incremented each time a UCMM message is received.
5	Get	Com Capacity	STRUCT of	
		Max CIP Connections	UINT	Maximum number of supported CIP connections.
		Max TCP Connections	UINT	Maximum number of supported TCP connections.
		Max Urgent priority rate	UINT	Maximum number of CIP transport class 0/1 Urgent priority message packets per second.
		Max Scheduled priority rate	UINT	Maximum number of CIP transport class 0/1 Scheduled priority message packets per second.
		Max High priority rate	UINT	Maximum number of CIP transport class 0/1 High priority message packets per second.
		Max Low priority rate	UINT	Maximum number of CIP transport class 0/1 Low priority message packets per second.
		Max Explicit Messaging rate	UINT	Max CIP transport class 2/3 or other EtherNet/IP messages packets per second

Attribute ID (hex)	Access	Name	Data Type	Details
6	Get	Bandwidth Diag	STRUCT of	
		Current sending Urgent priority rate	UINT	CIP transport class 0/1 Urgent priority message packets sent per second.
		Current reception Urgent priority rate	UINT	CIP transport class 0/1 Urgent priority message packets received per second.
		Current sending Scheduled priority rate	UINT	CIP transport class 0/1 Scheduled priority message packets sent per second.
		Current reception Scheduled priority rate	UINT	CIP transport class 0/1 Scheduled priority message packets received per second.
		Current sending High priority rate	UINT	CIP transport class 0/1 High priority message packets sent per second.
		Current reception High priority rate	UINT	CIP transport class 0/1 High priority message packets received per second.
		Current sending Low priority rate	UINT	CIP transport class 0/1 Low priority message packets sent per second.
		Current reception Low priority rate	UINT	CIP transport class 0/1 Low priority message packets received per second.
		Current sending Explicit Messaging rate	UINT	CIP transport class 2/3 or other EtherNet/IP message packets sent per second.
		Current reception Explicit Messaging rate	UINT	CIP transport class 2/3 or other EtherNet/IP message packets received per second.
7	Get	Modbus Diag	STRUCT of	
		Max. Modbus TCP Connections opened	UINT	Maximum number of TCP connections opened and used for Modbus communications.
		Current Modbus TCP Connections	UINT	Number of TCP connections currently opened and used for Modbus communications.
		Modbus TCP Msg Send Counter	UDINT	Incremented each time a Modbus TCP message is sent.
		Modbus TCP Msg Receive Counter	UDINT	Incremented each time a Modbus TCP message is received.

The following table describes the class services:

Service Code (hex)	Name	Description
01	Get_Attributes_All	Returns the value of all class attributes.
0E	Get_Attribute_Single	Returns the value of a specified attribute.
4C	Get_and_Clear	Gets and clears a specified attribute.

IOScanner Diagnostic Object (Class ID = 351 hex)

The following table describes the class attributes of the IOScanner Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	1	Increased by 1 on each new update of the object.
2	Get	Max Instance	UINT	1	Maximum instance number of the object.

The following table describes the instance attributes of the IOScanner Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Details
1	Get	IO Status Table	STRUCT of	
		Size	UINT	Size in bytes of the Status attribute.
		Status	ARRAY of UINT	I/O status. Bit n, where n is instance n of the object, provides the status of the I/O exchanged on the I/O connection: 0: The input or output status of the I/O connection is in error, or no device. 1: The input or output status of the I/O connection is correct.

The following table describes the class services:

Service Code (hex)	Name	Description
01	Get_Attributes_All	Returns the value of all class attributes.

IO Connection Diagnostic Object (Class ID = 352 hex)

The following table describes the class attributes of the IO Connection Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Increased by 1 on each new update of the object.
2	Get	Max Instance	UINT	01	Maximum instance number of the object. 0...n where n is the maximum number of CIP I/O connections. NOTE: There is an IO Connection Diagnostic object instance for both O->T and T->O paths.

The following table describes the instance attributes of the I/O Connection Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Details
1	Get Clear	IO Com Diag	STRUCT of	
		IO Production Counter	UDINT	Incremented each time a production is sent.
		IO Consumption Counter	UDINT	Incremented each time a consumption is received.
		IO Production Send Errors Counter	UINT	Incremented each time a production is not sent due to an error.
		IO Consumption Receive Errors Counter	UINT	Incremented each time a consumption is received that contains an error.
		CIP Connection TimeOut Errors	UINT	Incremented each time a connection times out.
		CIP Connection Opening Errors	UINT	Incremented on each unsuccessful attempt to open a connection.
		CIP Connection State	UINT	State of the CIP IO connection.
		CIP Last Error General Status	UINT	General status of the last error detected on the connection.
		CIP Last Error Extended Status	UINT	Extended status of the last error detected on the connection.
		Input Com Status	UINT	Communication status of the inputs.
		Output Com Status	UINT	Communication status of the outputs.
2	Get	Connection Diag	STRUCT of	
		Production Connection ID	UDINT	Connection ID for production.
		Consumption Connection ID	UDINT	Connection ID for consumption.
		Production RPI	UDINT	Requested Packet Interval (RPI) for productions, in μ s.
		Production API	UDINT	Actual Packet Interval (API) for productions.
		Consumption RPI	UDINT	RPI for consumptions.
		Consumption API	UDINT	API for consumptions.
		Production Connection Parameters	UDINT	Connection parameters for productions.
		Consumption Connection Parameters	UDINT	Connection parameters for consumptions.
		Local IP	UDINT	Local IP address for I/O communication.
		Local UDP Port	UINT	Local UDP port number for I/O communication.
		Remote IP	UDINT	Remote IP address for I/O communication.
		Remote UDP Port	UINT	Remote UDP port number for I/O communication.
		Production Multicast IP	UDINT	Multicast IP address for productions, or 0 if multicast is not used.
		Consumption Multicast IP	UDINT	Multicast IP address for consumptions, or 0 if multicast is not used.
		Protocols supported	UINT	Protocol(s) supported (0=not supported, 1=supported): • Bit 0: EtherNet/IP • Bit 1: Modbus TCP • Bit 2: Modbus Serial • Bits 3...15: Reserved, 0

Instance Attributes

The following table describes the class services:

Service Code (hex)	Name	Description
01	Get_Attributes_All	Returns the value of all class attributes.
0E	Get_Attribute_Single	Returns the value of the specified attribute.
4C	Get_and_Clear	Gets and clears a specified attribute.

Explicit Connection Diagnostic Object (Class ID = 353 hex)

The following table describes the class attributes of the Explicit Connection Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Increased by 1 at each new update of the object.
2	Get	Max Instance	UINT	0...n (maximum number of CIP IO connections)	Maximum instance number of the object.

The following table describes the instance attributes of the Explicit Connection Diagnostic object:

Attribute ID (hex)	Access	Name	Data Type	Details
1	Get	Originator Connection ID	UDINT	O to T Connection ID
2	Get	Originator IP	UDINT	–
3	Get	Originator TCP Port	UINT	–
4	Get	Target Connection ID	UDINT	T to O Connection ID
5	Get	Target IP	UDINT	–
6	Get	Target TCP Port	UINT	–
7	Get	Msg Send Counter	UDINT	Incremented each time a Class 3 CIP Message is sent on the connection
8	Get	Msg ReceiveCounter	UDINT	Incremented each time a Class 3 CIP Message is received on the connection.

Explicit Connections Diagnostic List Object (Class ID = 354 hex)

The following table describes the class attributes of the Explicit Connections Diagnostic List object:

Attribute ID (hex)	Access	Name	Data Type	Value (hex)	Details
1	Get	Revision	UINT	01	Increased by 1 at each new update of the object.
2	Get	Max Instance	UINT	0...n	n is the maximum number of concurrent list accesses supported.

The following table describes the instance attributes of the Explicit Connections Diagnostic List object:

Attribute ID (hex)	Access	Name	Data Type	Details
1	Get	Number of Connections	UINT	Total number of open Explicit connections.
2	Get	Explicit Messaging Connections Diagnostic List	ARRAY of STRUCT	Contents of instantiated Explicit Connection Diagnostic objects.
		Originator Connection ID	UDINT	Originator to Target connection ID.
		Originator IP	UDINT	Originator to Target IP address.
		Originator TCP Port	UINT	Originator to Target port number.
		Target Connection ID	UDINT	Target to Originator connection ID.
		Target IP	UDINT	Target to Originator IP address.
		Target TCP Port	UINT	Target to Originator port number.
		Msg Send Counter	UDINT	Incremented each time a Class 3 CIP message is sent on the connection.
		Msg Receive Counter	UDINT	Incremented each time a Class 3 CIP message is sent on the connection.

The following table describes the class services:

Service Code (hex)	Name	Description
08	Create	Creates an instance of the Explicit Connections Diagnostic List object.
09	Delete	Deletes an instance of the Explicit Connections Diagnostic List object.
33	Explicit_Connections_Diagnostic_Read	Explicit connections diagnostic read object.

Controller as a Slave Device on Modbus TCP

Overview

This section describes the configuration of the M262 Logic/Motion Controller as a **Modbus TCP Slave Device**.

Each M262 Logic/Motion Controller uses a Modbus server which does not need to be configured. The **Modbus TCP Slave Device** adds another Modbus server function to the controller. This server is addressed by the Modbus client application by specifying a configured Unit ID (Modbus address) in the range 1...247. The embedded Modbus server of the slave controller needs no configuration, and is addressed by specifying a Unit ID equal to 255. Refer to [Modbus TCP Configuration](#), page 176.

To configure your M262 Logic/Motion Controller as a **Modbus TCP Slave Device**, you must add **Modbus TCP Slave Device** functionality to your controller (see [Adding a Modbus TCP Slave Device](#) thereafter). This functionality creates a specific I/O area in the controller that is accessible with the Modbus TCP protocol. This I/O area is used whenever an external master needs to access the %IW and %QW objects of the controller. This **Modbus TCP Slave Device** functionality allows you to furnish to this area the controller I/O objects which can then be accessed with a single Modbus read/write registers request.

Inputs/outputs are seen from the slave controller: inputs are written by the master, and outputs are read by the master.

The **Modbus TCP Slave Device** can define a privileged Modbus client application, whose connection is not forcefully closed (embedded Modbus connections may be closed when more than 8 connections are needed).

The watchdog associated to the privileged connection allows you to verify whether the controller is being polled by the privileged master. If no Modbus request is received within the timeout duration, the diagnostic information *i_byMasterIpLost* is set to 1 (TRUE). For more information, refer to the Ethernet Port Read-Only System Variables (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

For further information about Modbus TCP, refer to the www.odva.org website.

Adding a Modbus TCP Slave Device

To add a Modbus TCP slave device, select **Modbus TCP Slave Device** in the **Hardware Catalog**.

Drag and drop it to the **Devices tree** on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Modbus TCP Configuration

To configure the Modbus TCP slave device, double-click **Ethernet_2 > ModbusTCP_Slave_Device** in the **Devices tree**.

This dialog box appears:

The screenshot shows a 'Configured Parameters' dialog box with the following settings:

- IPMaster Address:** 0 . 0 . 0 . 0
- Watchdog:** ☒ Watchdog: 2000 (ms)
- Slave Port:** 502
- Unit ID:** 247
- Holding Registers (%IW):** 10
- Input Registers (%QW):** 10

Element	Description
Configured Parameters	
IP Master Address	IP address of the Modbus master The connections are not closed on this address.
Watchdog	Watchdog in 500 ms increments NOTE: The watchdog applies to the IP master Address unless the address is 0.0.0.0.
Slave Port	Modbus communication port (502)
Unit ID	Sends the requests to the Modbus TCP slave device (1...247), instead of to the embedded Modbus server (255).
Holding Registers (%IW)	Number of %IW registers to be used in the exchange (2...40) (each register is 2 bytes)
Input Registers (%QW)	Number of %QW registers to be used in the exchange (2...40) (each register is 2 bytes)

Modbus TCP Slave Device I/O Mapping Tab

The I/Os are mapped to Modbus registers from the master perspective as follows:

- %IWs are mapped from register 0 to n-1 and are R/W (n = Holding register quantity, each %IW register is 2 bytes).
- %QWs are mapped from register n to n+m-1 and are read only (m = Input registers quantity, each %QW register is 2 bytes).

Once a **Modbus TCP Slave Device** has been configured, Modbus commands sent to its Unit ID (Modbus address) access the %IW and %QW objects of the controller instead of the regular Modbus words (accessed when the Unit ID is 255). This facilitates read/write operations by a Modbus TCP IOScanner application.

The **Modbus TCP Slave Device** responds to a subset of the Modbus commands, but does so in a way that differs from Modbus standards, and with the purpose of exchanging data with the external I/O scanner. The following Modbus commands are supported by the Modbus TCP slave device:

Function Code Dec (Hex)	Function	Comment
3 (3)	Read holding register	Allows the master to read %IW and %QW objects of the device
6 (6)	Write single register	Allows the master to write %IW objects of the device
16 (10)	Write multiple registers	Allows the master to write %IW objects of the device
23 (17)	Read/write multiple registers	Allows the master to read %IW and %QW objects of the device and write %IW objects of the device
Other	Not supported	–

NOTE: Modbus requests that attempt to access registers above n+m-1 are answered by the 02 - ILLEGAL DATA ADDRESS exception code.

To link I/O objects to variables, select the **Modbus TCP Slave Device I/O Mapping** tab:

Channel		Type	Description
Input	IW0	WORD	Holding register 0
	...	...	...
	IWx	WORD	Holding register x
Output	QW0	WORD	Input register 0
	...	...	...
	QWy	WORD	Input register y

The number of words depends on the **Holding Registers (%IW)** and **Input Registers (%QW)** parameters of the **Modbus TCP** tab.

NOTE: Output means OUTPUT from client/master controller (= %IW for the server/slave controller). Input means INPUT from client/master controller (= %QW for the server/slave controller).

Bus Cycle Options

In the **Modbus TCP Slave Device I/O Mapping** tab, select the **Bus cycle task** to use:

- **Use parent bus cycle setting** (the default),
- **MAST**
- **An existing task of the project:** you can select an existing task and associate it to the scanner. For more information about the application tasks, refer to the EcoStruxure Machine Expert Programming Guide (see EcoStruxure Machine Expert, Programming Guide).

NOTE: There is a corresponding **Bus cycle task** parameter in the I/O mapping editor of the device that contains the **Modbus TCP Slave Device**. This parameter defines the task responsible for refreshing the %IW and %QW registers.

Sercos Configuration

Introduction

This chapter describes how to configure the Sercos interface of the Modicon M262 Motion Controller.

Overview of the Sercos Standard

Introduction

The Sercos interface is a standardized interface (IEC 61491) for real-time communication between controllers, drives, servo drives, I/O devices, encoders, and other equipment requiring real-time services.

For motion control, the Sercos standard describes the internationally standardized digital interface for communication between a control unit and associated servo drives. It defines standardization of operating data, parameters, and scaling for machines with multiple drives that can be operated in torque, velocity, or position interface operation modes.

The main features of the Sercos interface are:

- Ring topology (redundancy)
- Master / slave system
- Baud rate 100 MBaud
- Minimum synchronization time of 1 ms (4 axes or 8 axes), 2 ms (16 axes), or 4 ms (24 axes)
- Synchronization (jitter < 1 μ s)

Data Exchange

Communication with Sercos interface is divided into two types:

- Cyclical communication:

The cyclical communication is used for exchanging real-time data (for example, position) and is executed once in every communication cycle (*CycleTime*). Certain specified data are transferred from the controller to all drives and from all drives to the controller in every cycle.

The exchange of information between the motion controller (Sercos master) and the servo drives (slaves), is accomplished via a message structure known as a telegram. There are three telegrams defined by IEC 61491:

- MST (Master Synchronization Telegram): An MST telegram is broadcast by the master at the beginning of each transmission cycle to synchronize the timing of the cycle.
 - MDT (Master Data Telegram): An MDT telegram is sent by the master once during each transmission cycle to transmit data (command values) to the servo drives (slaves).
 - AT (Acknowledge Telegram): AT telegrams are sent by the slaves to the master (feedback values).
- Non-cyclical communication with function blocks.

Non-cyclical communication is used to exchange data such as parameters for configuring communication, the drive parameters, status, and so on, where time is not a critical factor. The controller controls non-cyclical communication. All of the parameters in the system can be contacted using this channel, even parameters that are configured cyclically.

NOTE: The two types of communication can be used simultaneously.

IDN Description

IEC 61491 assigns identification numbers (IDNs) to all the operation data in a Sercos drive. Operation data includes parameters, interface procedure commands, and command and feedback values.

There are two categories of IDNs available:

- Standards IDNs (S): They are defined by the Sercos standard IEC 61491. Standards IDNs, if supported by a Sercos drive, behave the same, irrespective of the drive manufacturer;
- Proprietary IDNs (P): They are reserved for product-specific data that can be defined by the manufacturers of control units and servo drives.

Modicon M262 Logic/Motion Controller Sercos Configuration

Introduction

For more information about Modicon M262 Logic/Motion Controller Sercos configuration, refer to M262 Sercos For Modicon M262 Motion Controller - User Guide (see Sercos For Modicon M262 Motion Controller - User Guide).

Modicon M262 Motion Controller and Safety Controllers with Sercos

Introduction

Sercos fieldbus allows to connect Safety controllers. For more information, refer to the M262 Embedded Safety - Integration Guide (see M262 Embedded Safety - Integration Guide).

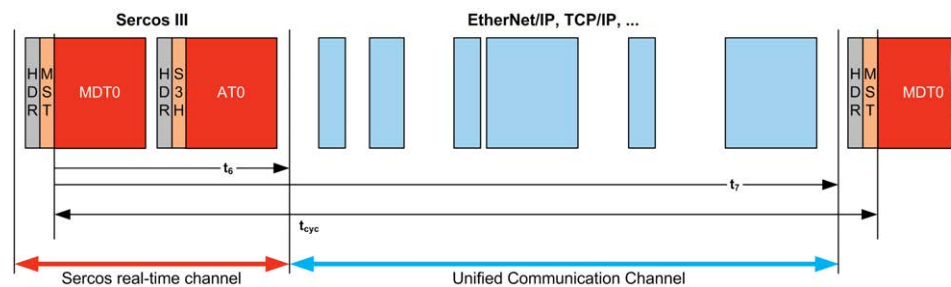
Single Wire Architecture

Overview

In addition to real-time and safety-critical data, the Sercos standard allows the transmission of Ethernet data over a common network infrastructure.

NOTE: The TM262M05MESS8T motion controller does not support the Single Wire Architecture.

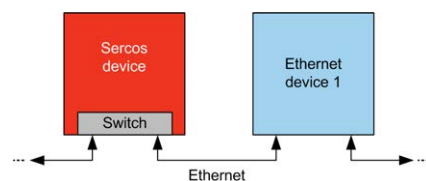
The EtherNet/IP or TCP/IP frames are embedded within the Sercos frame:



This Single Wire Architecture can be implemented using a single network cable connected to the controller. The Ethernet devices are added to the end of the cable after the Sercos devices.

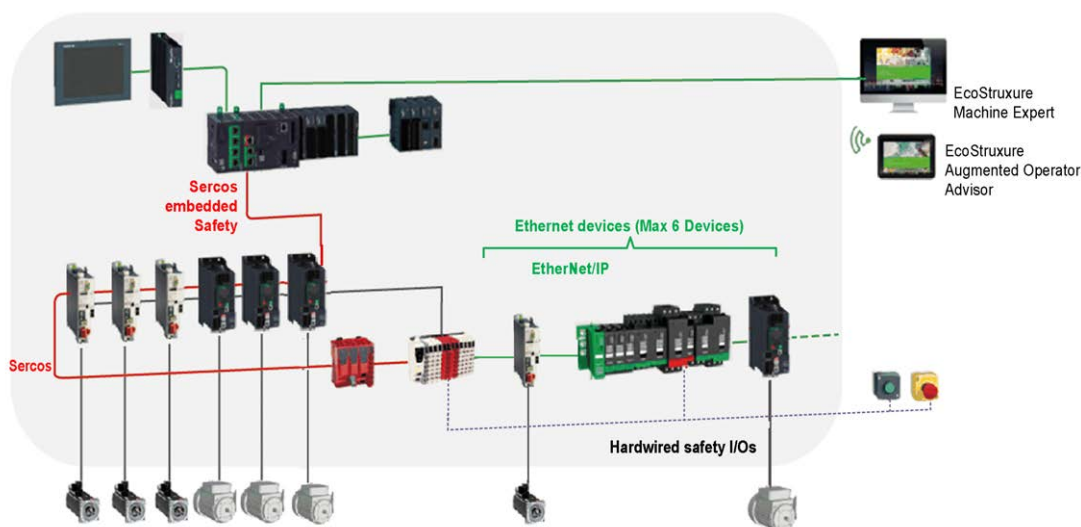
No additional cables or network components (gateways or switches) are required.

The last Sercos device on the cable acts as the gateway. It must have two Sercos connectors; one connected to the upstream Sercos devices, the other to the downstream Ethernet devices:



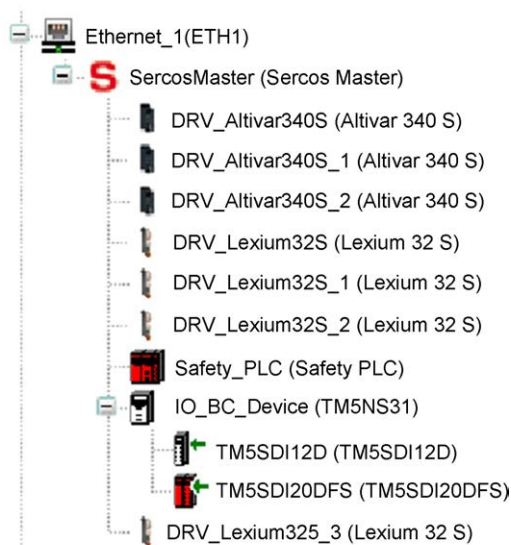
Up to 6 Ethernet devices can be added to the cable.

This figure shows an example Single Wire Architecture:

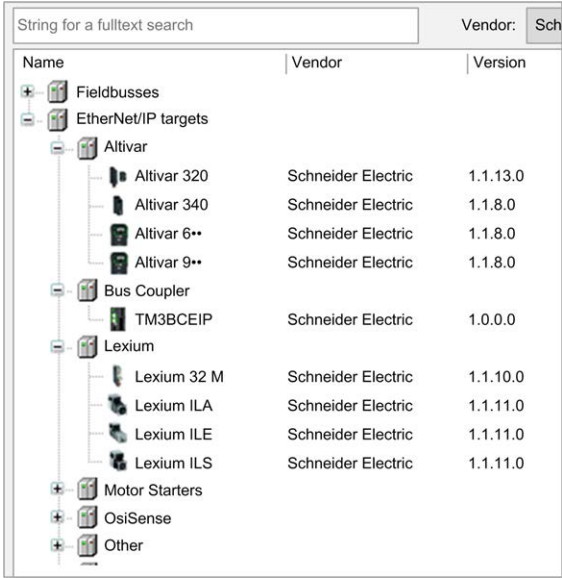


Single Wire Architecture in EcoStruxure Machine Expert

This figure shows the implementation of the example Single Wire Architecture in EcoStruxure Machine Expert:



To build this configuration:

Step	Action
1	Add the Sercos Master node and Sercos devices in the normal way.
2	Add up to a maximum of 6 Ethernet devices below the last Sercos device. Any of the Ethernet target devices available in the Device editor window can be added:  The screenshot shows the 'Device editor' window with a search bar and a list of devices. The list is organized into categories: Fieldbuses, EtherNet/IP targets, and Motor Starters. Under 'EtherNet/IP targets', there are sub-categories for 'Altivar' and 'Lexium'. The 'Altivar' category lists 'Altivar 320', 'Altivar 340', 'Altivar 6**', and 'Altivar 9**'. The 'Lexium' category lists 'Lexium 32 M', 'Lexium ILA', 'Lexium ILE', and 'Lexium ILS'. The 'Motor Starters' category lists 'OsiSense' and 'Other'. The 'Vendor' column for all devices is 'Schneider Electric'. The 'Version' column shows various versions, including 1.1.13.0, 1.1.8.0, 1.0.0.0, 1.1.10.0, 1.1.11.0, and 1.1.11.0.
3	Set the Sercos bus to the Phase 4 state to activate Ethernet communication. When commissioning Sercos devices, it may be necessary to downgrade the Sercos phase, for example, by adjusting the Communication Cycle Time parameter in the Sercos device). In this case, the Ethernet devices will enter a fallback state.

Serial Line Configuration

Introduction

This chapter describes how to configure the serial line communication of the Modicon M262 Logic/Motion Controller.

Serial Line Configuration

Introduction

The Serial Line configuration window allows you to configure the physical parameters of a serial line (baud rate, parity, and so on).

Serial Line Configuration

To configure a Serial Line, double-click **Serial line** in the **Devices tree**.

The following parameters must be identical for each serial device connected to the port.

Element	Description
Baud rate	Transmission speed in bits/s
Parity	Used for error detection
Data bits	Number of bits for transmitting data
Stop bits	Number of stop bits
Physical Medium	Specify the medium to use: • RS485 (using polarisation resistor or not) • RS232
Polarization Resistor	Polarization resistors are integrated in the controller. They are switched on or off by this parameter.

The serial line ports of your controller are configured for the Machine Expert protocol by default when new or when you update the controller firmware. The Machine Expert protocol is incompatible with that of other protocols such as Modbus Serial Line. Connecting a new controller to, or updating the firmware of a controller connected to, an active Modbus configured serial line can cause the other devices on the serial line to stop communicating. Make sure that the controller is not connected to an active Modbus serial line network before first downloading a valid application having the concerned port or ports properly configured for the intended protocol.

NOTICE

INTERRUPTION OF SERIAL LINE COMMUNICATIONS

Be sure that your application has the serial line ports properly configured for Modbus before physically connecting the controller to an operational Modbus Serial Line network.

Failure to follow these instructions can result in equipment damage.

This table indicates the maximum baud rate value of the managers:

Manager	Maximum Baud Rate (Bits/S)
Machine Expert Network Manager	115200
Modbus Manager	
ASCII Manager	
Modbus IOScanner	

Serial Line Diagnostic Table

To access the **Serial Line Diagnostic Table**, double click the **Serial Line** node in the **Tools tree** tab. Diagnostic information are accessible with the structure **SERDIAG_W_STRUCT**. For more information, refer to Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

Machine Expert Network Manager

Introduction

Use the Machine Expert Network Manager to exchange variables with a Magelis Advanced Panel with Machine Expert software protocol, or when the Serial Line is used for EcoStruxure Machine Expert programming.

Adding the Manager

To add a Machine Expert Network Manager to your controller, select the **Machine Expert - Network Manager** in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Configuring the Manager

There is no configuration for Machine Expert Network Manager.

Adding a Modem

To add a modem to the Machine Expert Network Manager, refer to [Adding a Modem to a Manager](#), page 197.

Modbus Manager

Introduction

The Modbus Manager is used for Modbus RTU or ASCII protocol in master or slave mode.

Adding the Manager

To add a Modbus manager to your controller, select the **Modbus Manager** in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

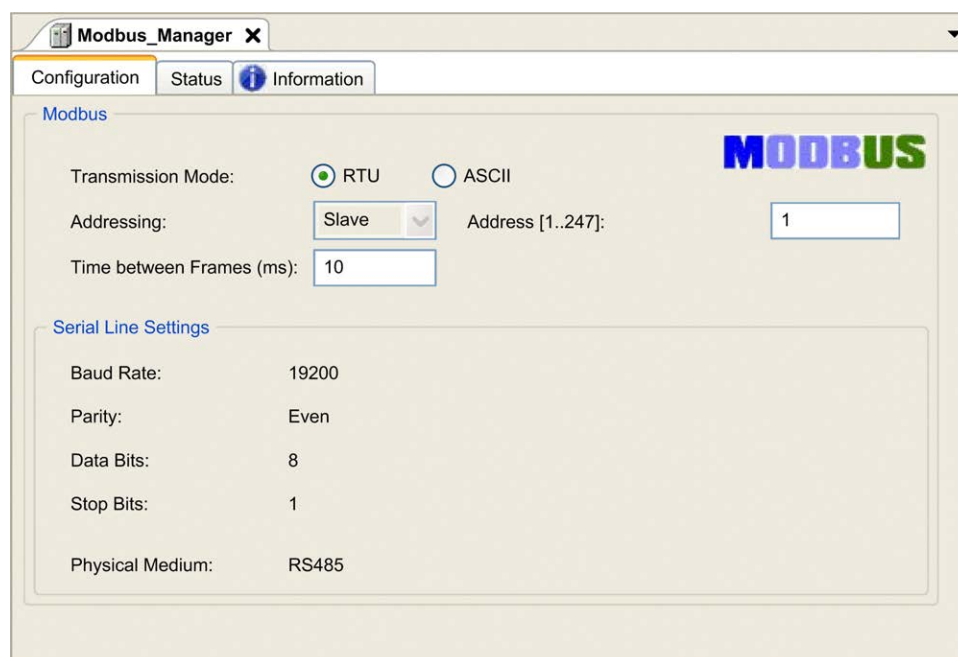
For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Modbus Manager Configuration

To configure the Modbus Manager of your controller, double-click **Modbus Manager** in the **Devices tree**.

The Modbus Manager configuration window is displayed as below:



Modbus_Manager X

Configuration Status Information

Modbus

Transmission Mode: ☒ RTU ☐ ASCII

Addressing: Slave Address [1..247]: 1

Time between Frames (ms): 10

Serial Line Settings

Baud Rate: 19200

Parity: Even

Data Bits: 8

Stop Bits: 1

Physical Medium: RS485

Set the parameters as described in this table:

Element	Description
Transmission Mode	Specify the transmission mode to use: - RTU: uses binary coding and CRC error-checking (8 data bits) - ASCII: messages are in ASCII format, LRC error-checking (7 data bits) Set this parameter identical for each Modbus device on the link.
Addressing	Specify the device type: - Master - Slave
Address	Modbus address of the device, when slave is selected.
Time between Frames (ms)	Time to avoid bus-collision. Set this parameter identical for each Modbus device on the link.
Serial Line Settings	Parameters specified in the Serial Line configuration window.

Modbus Master

When the controller is configured as a Modbus Master, the following function blocks are supported from the PLCCommunication Library:

- ADDM
- READ_VAR
- SEND_RECV_MSG
- SINGLE_WRITE
- WRITE_READ_VAR
- WRITE_VAR

For further information, see Function Block Descriptions (see EcoStruxure Machine Expert, Modbus and ASCII Read/Write Functions, PLCCommunication Library Guide) of the PLCCommunication Library.

Modbus Slave

When the controller is configured as Modbus Slave, the following Modbus requests are supported:

Function Code Dec (Hex)	Sub-Function Dec (Hex)	Function
1 (1 hex)	–	Read digital outputs (%Q)
2 (2 hex)	–	Read digital inputs (%I)
3 (3 hex)	–	Read multiple register (%MW)
6 (6 hex)	–	Write single register (%MW)
8 (8 hex)	–	Diagnostic
15 (F hex)	–	Write multiple digital outputs (%Q)
16 (10 hex)	–	Write multiple registers (%MW)
23 (17 hex)	–	Read/write multiple registers (%MW)
43 (2B hex)	14 (E hex)	Read device identification

This table contains the sub-function codes supported by the diagnostic Modbus request 08:

Sub-Function Code		Function
Dec	Hex	
10	0A	Clears Counters and Diagnostic Register
11	0B	Returns Bus Message Count
12	0C	Returns Bus Communication Error Count
13	0D	Returns Bus Exception Error Count
14	0E	Returns Slave Message Count
15	0F	Returns Slave No Response Count
16	10	Returns Slave NAK Count
17	11	Returns Slave Busy Count
18	12	Returns Bus Character Overrun Count

This table lists the objects that can be read with a read device identification request (basic identification level):

Object ID	Object Name	Type	Value
00 hex	Vendor name	ASCII String	Schneider Electric
01 hex	Product code	ASCII String	Controller reference
02 hex	Major / Minor revision	ASCII String	aa.bb.cc.dd (same as device descriptor)

The following section describes the differences between the Modbus memory mapping of the controller and HMI Modbus mapping. If you do not program your application to recognize these differences in mapping, your controller and HMI will not communicate correctly. Thus it will be possible for incorrect values to be written to memory areas responsible for output operations.

⚠ WARNING

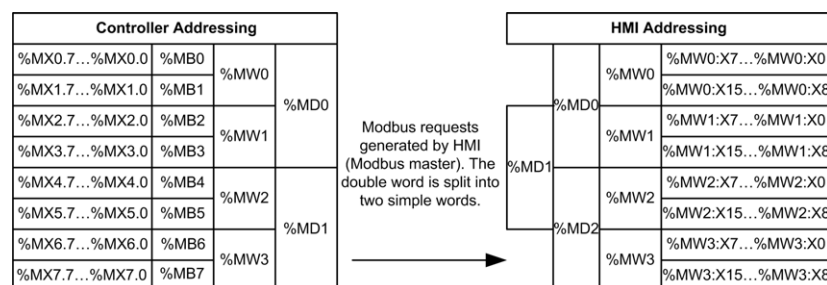
UNINTENDED EQUIPMENT OPERATION

Program your application to translate between the Modbus memory mapping used by the controller and that used by any attached HMI devices.

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

When the controller and the Magelis HMI are connected via Modbus (HMI is master of Modbus requests), the data exchange uses simple word requests.

There is an overlap on simple words of the HMI memory while using double words but not for the controller memory (see following diagram). In order to have a match between the HMI memory area and the controller memory area, the ratio between double words of HMI memory and the double words of controller memory has to be 2.



The following gives examples of memory match for the double words:

- %MD2 memory area of the HMI corresponds to %MD1 memory area of the controller because the same simple words are used by the Modbus request.
- %MD20 memory area of the HMI corresponds to %MD10 memory area of the controller because the same simple words are used by the Modbus request.

The following gives examples of memory match for the bits:

- %MW0:X9 memory area of the HMI corresponds to %MX1.1 memory area of the controller because the simple words are split in 2 distinct bytes in the controller memory.

Adding a Modem

To add a Modem to the Modbus Manager, refer to *Adding a Modem to a Manager*, page 197.

ASCII Manager

Introduction

The ASCII manager is used on a Serial Line, to transmit and/or receive data with a simple device.

Adding the Manager

To add an ASCII manager to your controller, select the **ASCII Manager** in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

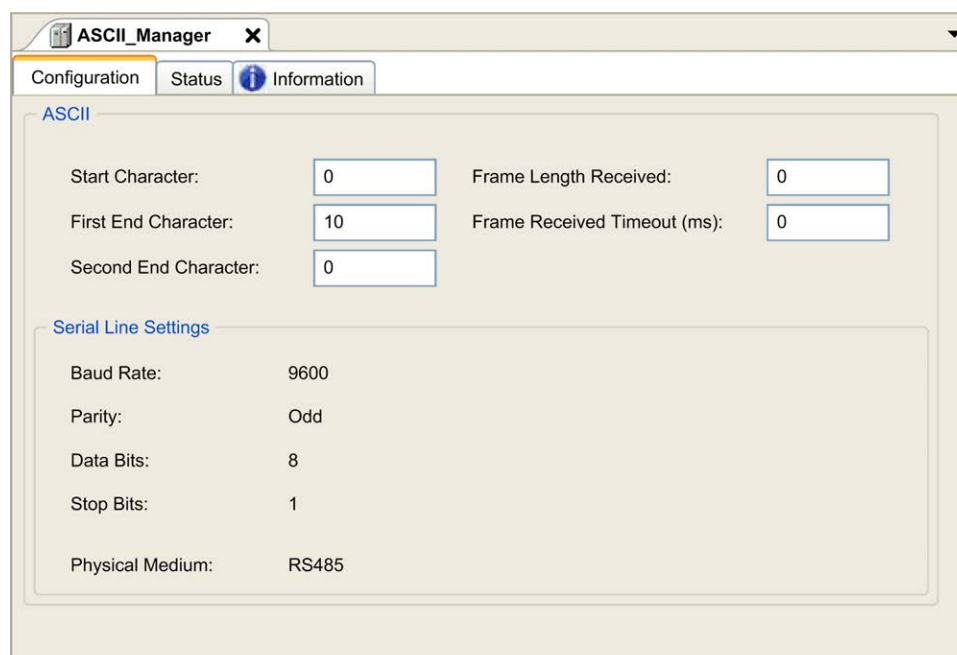
For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

ASCII Manager Configuration

To configure the ASCII manager of your controller, double-click **ASCII Manager** in the **Devices tree**.

The ASCII Manager configuration window is displayed as below:



ASCII Manager

Configuration | Status | Information

ASCII

Start Character: 0 Frame Length Received: 0

First End Character: 10 Frame Received Timeout (ms): 0

Second End Character: 0

Serial Line Settings

Baud Rate: 9600

Parity: Odd

Data Bits: 8

Stop Bits: 1

Physical Medium: RS485

Set the parameters as described in this table:

Parameter	Description
Start Character	If 0, no start character is used in the frame. Otherwise, in Receiving Mode , the corresponding character in ASCII is used to detect the beginning of a frame. In Sending Mode , this character is added at the beginning of the frame.
First End Character	If 0, no first end character is used in the frame. Otherwise, in Receiving Mode , the corresponding character in ASCII is used to detect the end of a frame. In Sending Mode , this character is added at the end of the frame.
Second End Character	If 0, no second end character is used in the frame. Otherwise, in Receiving Mode , the corresponding character in ASCII is used to detect the end of a frame. In Sending Mode , this character is added at the end of the frame.
Frame Length Received	If 0, this parameter is not used. This parameter allows the system to conclude an end of frame at reception when the controller received the specified number of characters. Note: This parameter cannot be used simultaneously with Frame Received Timeout (ms) .
Frame Received Timeout (ms)	If 0, this parameter is not used. This parameter allows the system to conclude the end of frame at reception after a silence of the specified number of ms.
Serial Line Settings	Parameters specified in the <i>Serial Line configuration window</i> , page 183.

NOTE: In the case of using several frame termination conditions, the first condition to be TRUE terminates the exchange.

Adding a Modem

To add a Modem to the ASCII manager, refer to *Adding a Modem to a Manager*, page 197.

Modbus Serial IOScanner

Introduction

The Modbus IOScanner is used to simplify exchanges with Modbus slave devices.

Add a Modbus IOScanner

To add a Modbus IOScanner on a Serial Line, select the **Modbus IOScanner** in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on one of the highlighted nodes.

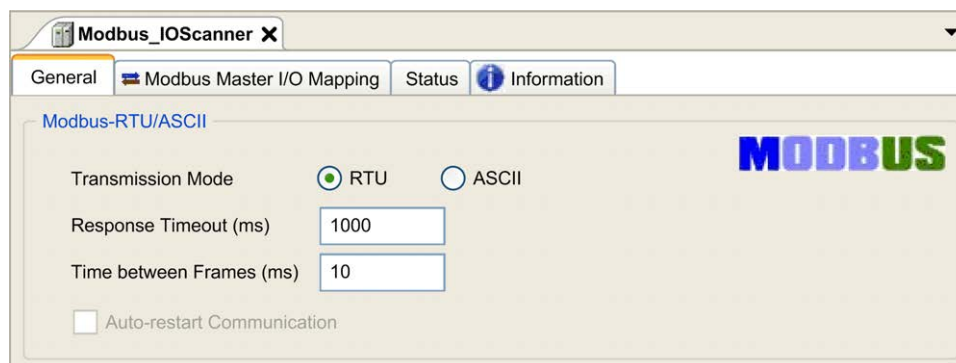
For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

Modbus IOScanner Configuration

To configure a Modbus IOScanner on a Serial Line, double-click **Modbus IOScanner** in the **Devices tree**.

The configuration window is displayed as below:



Set the parameters as described in this table:

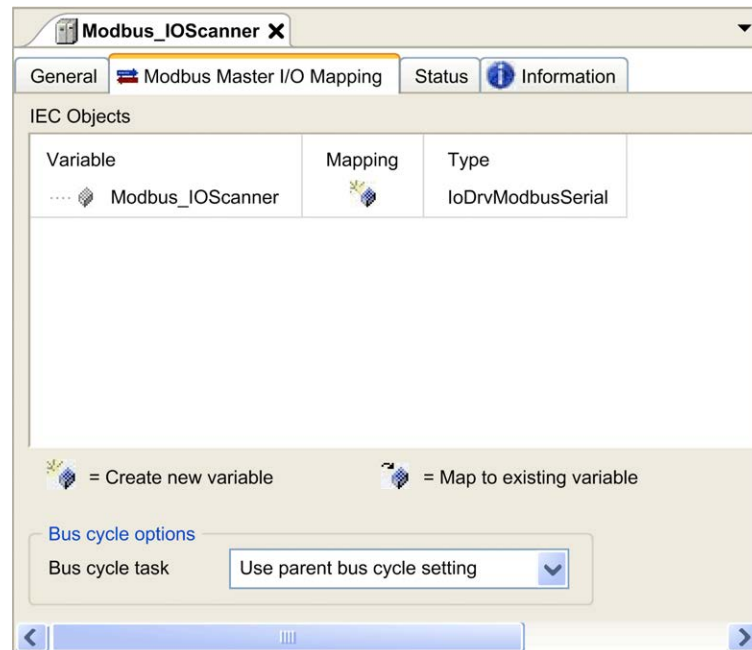
Element	Description
Transmission Mode	Specifies the transmission mode to use: • RTU: uses binary coding and CRC error-checking (8 data bits) • ASCII: messages are in ASCII format, LRC error-checking (7 data bits) Set this parameter identical for each Modbus device on the network.
Response Timeout (ms)	Timeout used in the exchanges.
Time between Frames (ms)	Delay to reduce data collision on the bus. Set this parameter identical for each Modbus device on the network.

NOTE: Do not use function blocks of the PLCCommunication library on a serial line with a Modbus IOScanner configured. This disrupts the Modbus IOScanner exchange.

Bus Cycle Task Selection

The Modbus IOScanner and the devices exchange data at each cycle of the chosen application task.

To select this task, select the **Modbus Master IO Mapping** tab. The configuration window is displayed as below:



The **Bus cycle task** parameter allows you to select the application task that manages the scanner:

- **Use parent bus cycle setting:** associate the scanner with the application task that manages the controller.
- **MAST:** associate the scanner with the MAST task.
- **Another existing task:** you can select an existing task and associate it to the scanner. For more information about the application tasks, refer to the EcoStruxure Machine Expert - Programming Guide (see EcoStruxure Machine Expert, Programming Guide).

The scan time of the task associated with the scanner must be less than 500 ms.

Adding a Device on the Modbus Serial IOScanner

Introduction

This section describes how to add a device on the Modbus IOScanner.

Adding a Device on the Modbus IOScanner

To add a device on the Modbus IOScanner, select the **Generic Modbus Slave** in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on the **Modbus_IOScanner** node of the **Devices tree**.

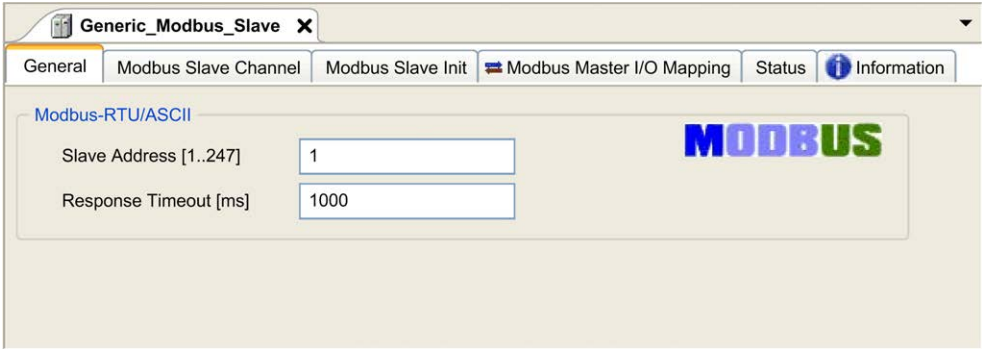
For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)

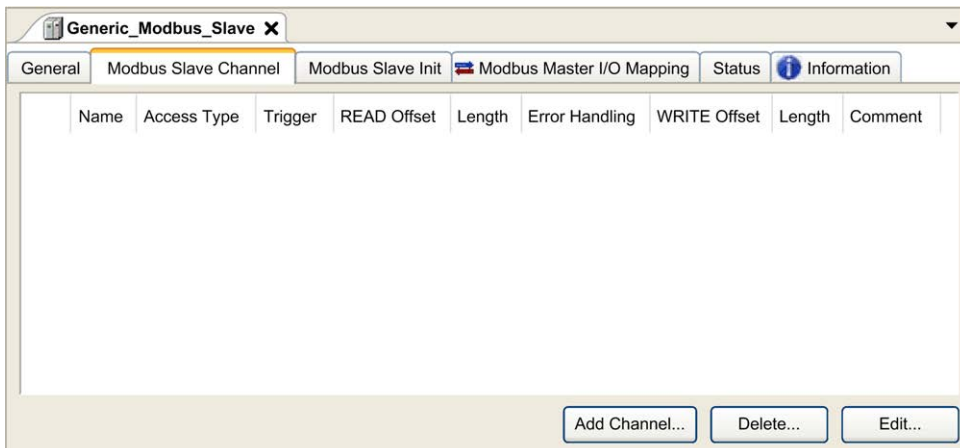
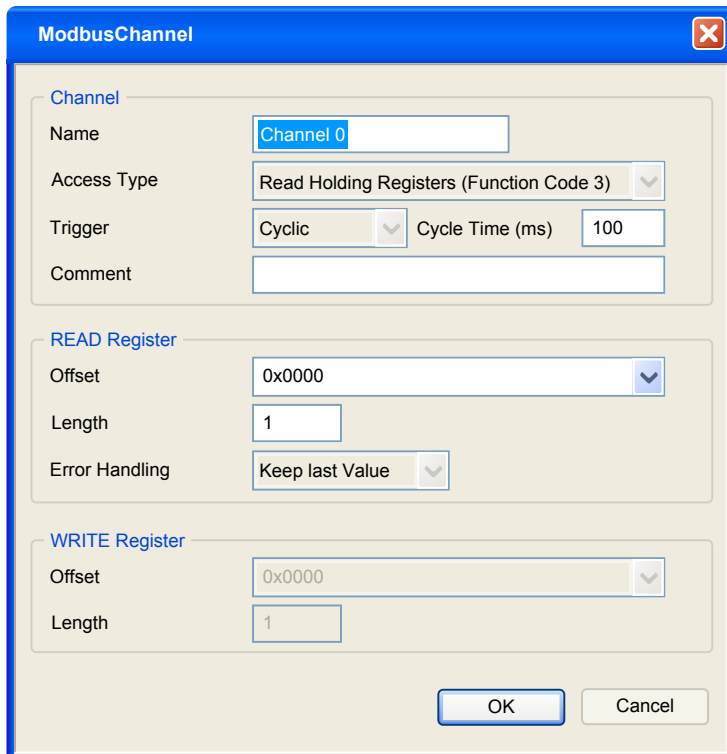
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)
- NOTE:** The variable for the exchange is automatically created in the %IWx and %QWx of the **Modbus Serial Master I/O Mapping** tab.

Configuring a Device Added on the Modbus IOScanner

To configure the device added on the Modbus IOScanner, proceed as follows:

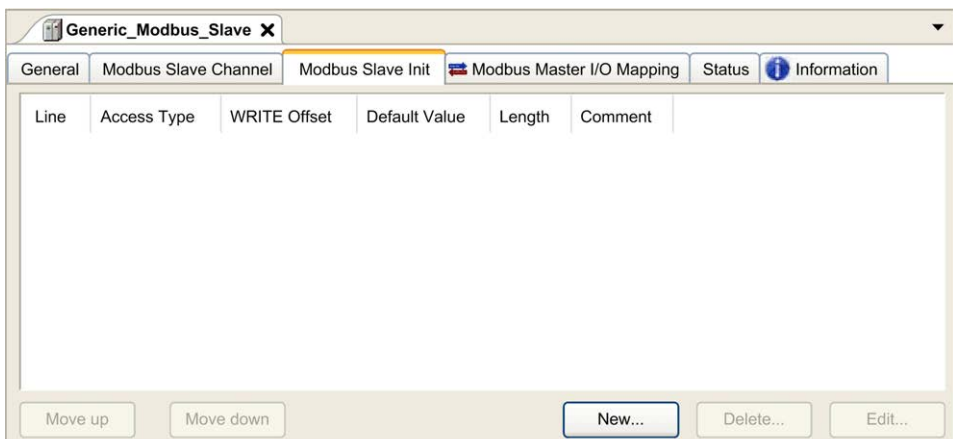
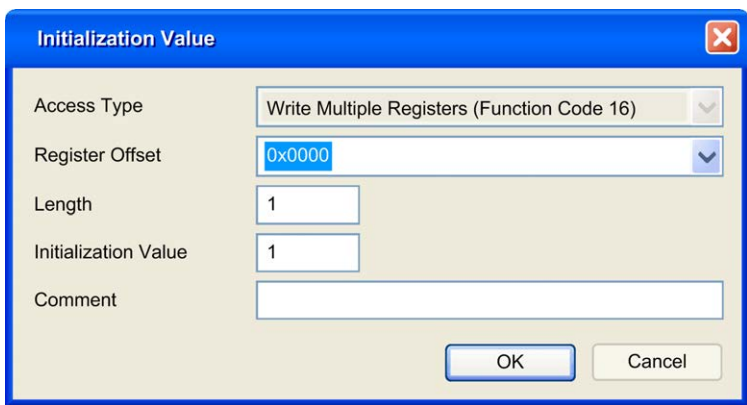
Step	Action
1	In the Devices tree, double-click Generic Modbus Slave. Result: The configuration window is displayed. 
2	Enter a Slave Address value for your device (choose a value from 1 to 247).
3	Choose a value for the Response Timeout (in ms).

To configure the **Modbus Channels**, proceed as follows:

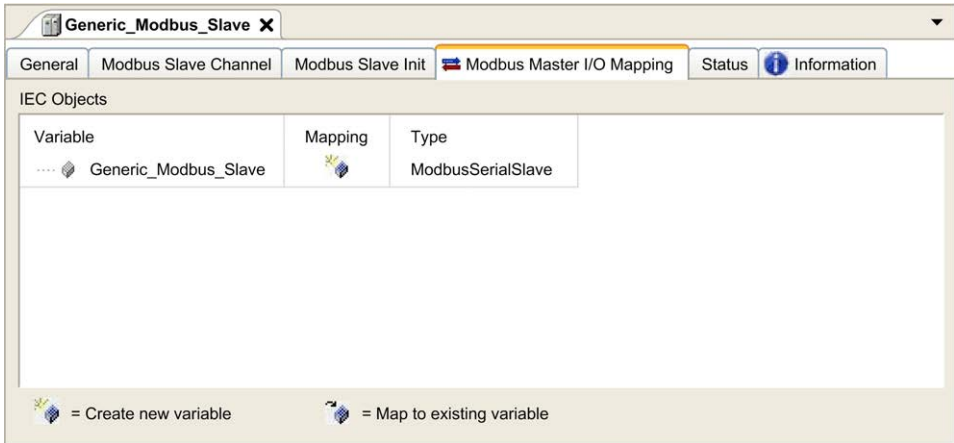
Step	Action
1	Click the Modbus Slave Channel tab: 
2	Click the Add Channel button: 

Step	Action																											
3	Configure an exchange: In the area Channel, you can add the following values: • Name: Enter a name for your channel. • Access Type: Choose the exchange type: Read or Write or Read/Write requests. See <i>Access Types</i>, page 196. • Trigger: Choose the trigger of the exchange. It can be CYCLIC with the period defined in Cycle Time (ms) field, started by a RISING EDGE on a boolean variable (this boolean variable is then created in the Modbus Master I/O Mapping tab), or by the Application. • Comment: Add a comment about this channel. In the area READ Register (if your channel is Read or Read/Write one), you can configure the %MW to be read on the Modbus slave. Those are mapped on %IW (see Modbus Master I/O Mapping tab): • Offset: Offset of the %MW to read. 0 means that the first object that is read is %MW0. • Length: Number of %MW to be read. For example, if 'Offset' = 2 and 'Length' = 3, the channel reads %MW2 , %MW3 and %MW4. • Error Handling: choose the behavior of the related %IW in case of loss of communication. In the area WRITE Register (if your channel is Write or Read/Write one), you can configure the %MW to be written to the Modbus slave. Those are mapped on %QW (see Modbus Master I/O Mapping tab): • Offset: Offset of the %MW to write. 0 means that the first object that is written is %MW0. • Length: Number of %MW to be written. For example, if 'Offset' = 2 and 'Length' = 3, the channel writes %MW2 , %MW3 and %MW4.																											
4	Click OK to validate the configuration of this channel. NOTE: You can also: • Click the Delete button to remove a channel. • Click the Edit button to change the parameters of a channel. Result: The configured channels are displayed: Library ManagerSR_MainTM3BC_ModbusTCP™TM3BC_ModbusSLGeneric_Modbus_Slave XModbus_IOScanner GeneralModbus Slave ChannelModbus Slave InitModbus Master I/O MappingModbus Master IEC ObjectsStatusInformation <table><tr><th>Name</th><th>Access Type</th><th>Trigger</th><th>READ Offset</th><th>Length</th><th>Error Handling</th><th>WRITE Offset</th><th>Length</th><th>Comment</th></tr><tr><td>0 Channel 0</td><td>Read Holding Registers (Function Code 03)</td><td>Cyclic, t#100ms</td><td>16#0000</td><td>1</td><td>Keep last value</td><td></td><td></td><td></td></tr><tr><td>1 Channel 1</td><td>Write Multiple Registers (Function Code 16)</td><td>Cyclic, t#100ms</td><td></td><td></td><td></td><td>16#0000</td><td>1</td><td></td></tr></table>	Name	Access Type	Trigger	READ Offset	Length	Error Handling	WRITE Offset	Length	Comment	0 Channel 0	Read Holding Registers (Function Code 03)	Cyclic, t#100ms	16#0000	1	Keep last value				1 Channel 1	Write Multiple Registers (Function Code 16)	Cyclic, t#100ms				16#0000	1	
Name	Access Type	Trigger	READ Offset	Length	Error Handling	WRITE Offset	Length	Comment																				
0 Channel 0	Read Holding Registers (Function Code 03)	Cyclic, t#100ms	16#0000	1	Keep last value																							
1 Channel 1	Write Multiple Registers (Function Code 16)	Cyclic, t#100ms				16#0000	1																					

To configure your **Modbus Initialization Value**, proceed as follows:

Step	Action
1	Click the Modbus Slave Init tab: 
2	Click New to create a new initialization value:  The Initialization Value window contains the following parameters: • Access Type: Enter the exchange type: Write requests , page 196. • Register Offset: Register number of register to be initialized. • Length: Number of %MW to be read. For example, if 'Offset' = 2 and 'Length' = 3, the channel reads %MW2 , %MW3 and %MW4. • Initialization Value: Value the registers are initialized with. • Comment: Add a comment about this channel.
3	Click OK to create a new Initialization Value. NOTE: You can also: • Click Move up or Move down to change the position of a value in the list. • Click Delete to remove a value in the list. • Click Edit to change the parameters of a value.

To configure your **Modbus Master I/O Mapping**, proceed as follows:

Step	Action
1	Click the Modbus Master I/O Mapping tab: 
2	Double-click in a cell of the Variable column to open a text field. Enter the name of a variable or click the browse button [...] and chose a variable with the Input Assistant.
3	For more information on I/O mapping, refer to EcoStruxure Machine Expert Programming Guide.

Access Types

This table describes the different access types available:

Function	Function Code	Availability
<i>Read Coils</i>	1	ModbusChannel
<i>Read Discrete Inputs</i>	2	ModbusChannel
<i>Read Holding Registers</i> (default setting for the channel configuration)	3	ModbusChannel
<i>Read Input Registers</i>	4	ModbusChannel
<i>Write Single Coil</i>	5	ModbusChannel Initialization Value
<i>Write Single Register</i>	6	ModbusChannel Initialization Value
<i>Write Multiple Coils</i>	15	ModbusChannel Initialization Value
<i>Write Multiple Registers</i> (default setting for the slave initialization)	16	ModbusChannel Initialization Value
<i>Read/Write Multiple Registers</i>	23	ModbusChannel

ControlChannel: Enables or Disables a Communication Channel

Function Description

This function allows you to enable or disable a communication channel.

A channel managed by this function is reinitialized to its default value after a reset (cold/warm).

After a stop or after a start, the channel remains disabled if it was disabled before.

On the contrary, after a reset, the channel is enabled even if it was disabled before.

In the case of the TM3BCSL Modbus Serial Line bus coupler, there are multiple, separate and independent communication channels.

⚠ WARNING

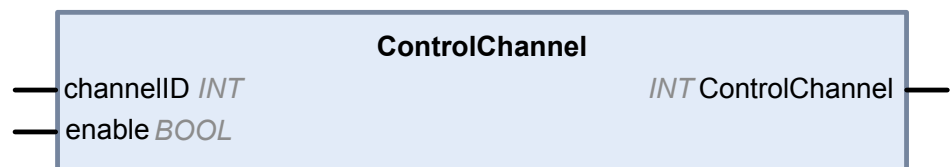
UNINTENDED EQUIPMENT OPERATION

Ensure that the Modbus serial line communication channels of the TM3BCSL bus coupler are set to the same state, either enabled or disabled.

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

NOTE: Use *ChannelID* value -1 to apply the *ControlChannel* on all channels configured on the TM3BCSL Modbus Serial Line bus coupler.

Graphical Representation



I/O Variable Description

This table describes the input variables:

Input	Type	Comment
<i>ControlChannel</i>	INT	Return 0 on success, a negative value on error.
<i>ChannelID</i>	INT	The channel number (visible in the first column of the configuration page). Or -1 to apply the command on all channels of this device.

This table describes the output variable:

Output	Type	Comment
<i>Enable</i>	BOOL	Enable or disable command.

Adding a Modem to a Manager

Introduction

A modem can be added to the following managers:

- ASCII Manager
- Modbus Manager

- Machine Expert Network Manager

NOTE: Use a modem which implements Hayes commands if you need a modem connection with Machine Expert Network Manager.

Adding a Modem to a Manager

To add a modem to your controller, select the modem you want in the **Hardware Catalog**, drag it to the **Devices tree**, and drop it on the manager node.

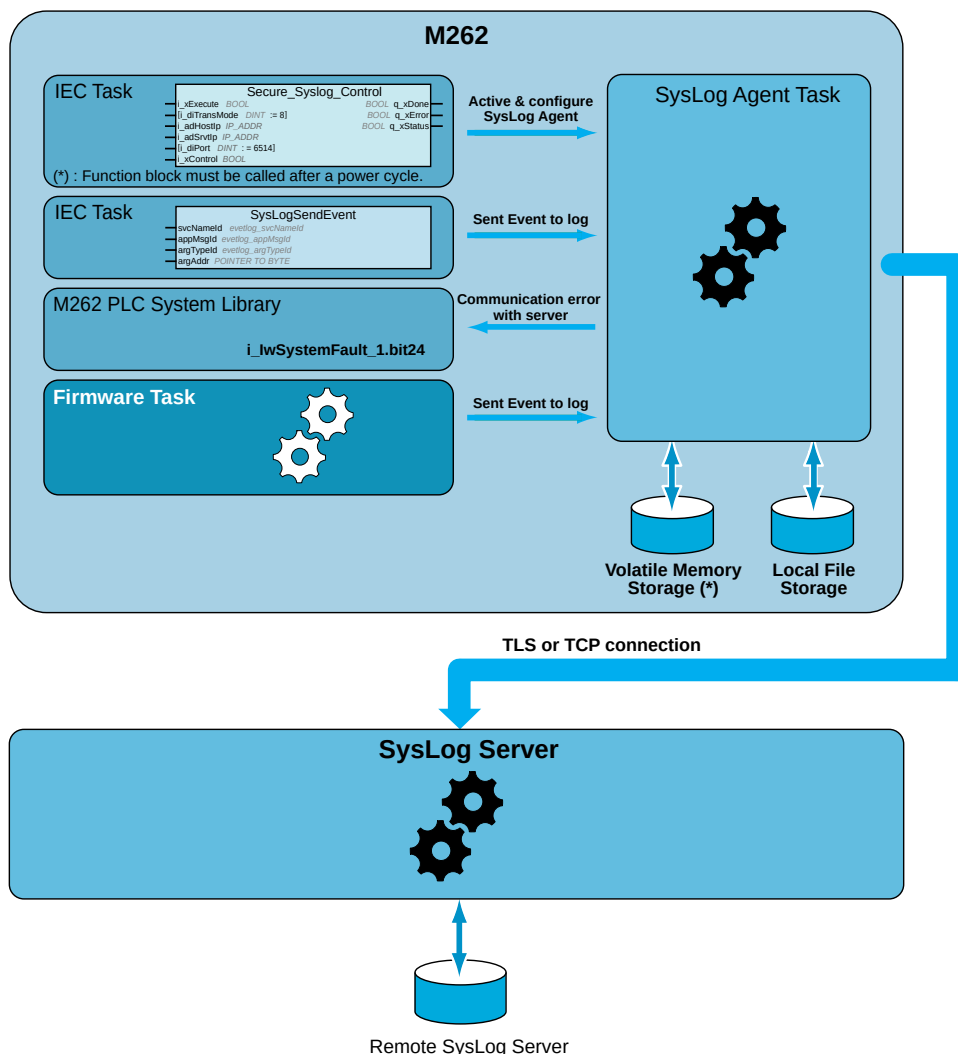
For more information on adding a device to your project, refer to:

- Using the Hardware Catalog (see EcoStruxure Machine Expert, Programming Guide)
- Using the Contextual Menu or Plus Button (see EcoStruxure Machine Expert, Programming Guide)

For further information, refer to EcoStruxure Machine Expert, Modem Functions, Modem Library Guide (see EcoStruxure Machine Expert, Modem Functions, Modem Library Guide).

SysLog Agent

System Overview



To activate a SysLog Agent, you have to instantiate a function block in your application. This function block starts and configures a SysLog Agent which operates even if you operate commands to download an application, stop, run or halt your controller.

NOTE: SysLog Agent must be started again after a power cycle.

You can send an Event to Log by using the `SysLogSendEvent` function. Refer to EcoStruxure Machine Expert – Syslog Library Guide.

SysLog Agent stores its configuration to communicate with the SysLog Server in the Volatile Memory Storage. SysLog Agent uses some directories as public key infrastructure (PKI) in Local File Storage to manage certificates (refer to EcoStruxure Machine Expert – Syslog Library Guide) of allowed server.

SysLog Agent stores historical information about Event to Log on SysLog Server. These files are useful to restore Events to Log during a period of disconnection. The Modicon M262 Logic/Motion Controller can store at least 2048 Events in these files.

The file access is restricted by the configuration of User Rights on your controller.

Diagnostic of SysLog Agent

A system bit (refer to Modicon M262 Logic/Motion Controller – System Library Guide) is set to 0 when an error is detected. This bit is identified as **PLC_GVL.PLC_R.i_lwSystemFault_1**.

TLS and Controller Compatibility

SysLog Agent is compatible with:

- TLS1.2 and TLS1.3
- TM262 firmware version 5.1.6.1 or later

OPC UA

Overview

This chapter describes how to configure the OPC UA server and client services of the M262 Logic/Motion Controller.

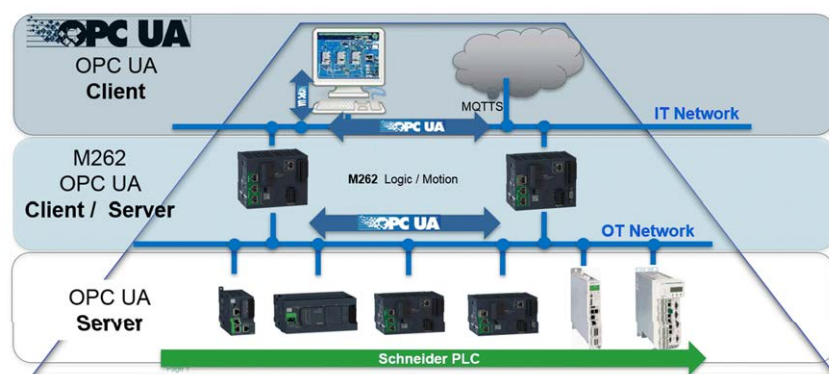
Overview

OPC UA Overview

Introduction

OPC Unified Architecture (OPC UA) is a vendor-independent communication protocol for industrial automation applications.

The M262 Logic/Motion Controller embeds both client and server services:



OPC UA Server Configuration

OPC UA Server Overview

Overview

The OPC Unified Architecture server (OPC UA server) allows the M262 Logic/Motion Controller to exchange data with OPC UA clients. Server and client communicate through sessions.

The items of data (also referred to as symbols) to be shared by the OPC UA server are manually selected from a list of the IEC variables used in the application.

The OPC UA server supports Read and Write access as well as the subscription model. When using the subscription model, the OPC UA server reads the values of symbols from devices at a fixed sampling rate, places the data in a queue, then sends them to clients as notifications at a regular publishing interval. The sampling interval can be shorter than the publishing interval, in which case notifications may be queued until the publishing interval elapses.

Symbols that have not changed value since the previous sample are not re-published. Instead, the OPC UA server sends regular KeepAlive messages to indicate to the client that the connection is still active.

User and Group Access Rights

Access to the OPC UA server is controlled by user rights. Refer to User Rights, page 69.

OPC UA Services

The following OPC Foundation Profiles and facets are implemented:

- OPC UA Micro Embedded Profile 2017
 - Core 2017 Server Facet
 - UA-TCP UA-SC UA-Binary
 - Embedded DataChange Subscription
- Security Category
 - SecurityPolicy – Basic256Sha256
 - SecurityPolicy – Basic256

The following functionalities are supported:

- Address Space Model
- Attribute Services
- Base Information
- Discovery Services
- Monitored item Services
- Protocol and Encoding (TCP UA Binary)
- Security
- Session Services
- Subscription Services
- View Services

OPC UA Server Configuration

Introduction

The OPC UA Server Configuration window allows you to configure the OPC UA server. OPC UA server is using encrypted communication by default with maximum security settings set by default.

Accessing the OPC UA Server Configuration Tab

To configure the OPC UA Server:

Step	Action
1	In the Devices tree , double-click MyController .
2	Select the OPC UA Server Configuration tab.

OPC UA Server Configuration Tab

The following figure shows the OPC UA Server Configuration window:

The screenshot displays the OPC UA Server Configuration window with the following settings:

- General settings:**
 - ☒ OPC UA Server enabled
- Security settings:**
 - ☒ Disable anonymous login
 - User credentials are managed in the Users and groups tab: [Users and Groups](#)
 - Security Policy: Basic256Sha256 (selected), None, Basic256(deprecated)
 - Message Security: SignAndEncrypt (selected), Sign
- Server configuration:**
 - Server port: 4840
 - Max subscriptions per session: 20
 - Max monitored items per subscription: 100
 - Max number of sessions: 2
 - Identifier type: String
 - Min publishing interval: 1000 ms
 - Min KeepAlive interval: 500 ms
- Diagnostic:**
 - ☐ Enable trace
 - Trace level: All
- Sampling rates (ms):**
 - Double-click to edit
 - 500
 - 1000
 - 2000
- Buttons:**
 - Reset to default

OPC UA Server Configuration Description

This table describes the OPC UA Server Configuration parameters:

General Settings

Parameter	Value	Default value	Description
OPC UA Server Enabled	Enabled/ Disabled	Disabled	This checkbox is used to enable or disable the OPC UA Server and Client on the controller.

Security Settings

Parameter	Value	Default value	Description
Disable anonymous login	Enabled/ Disabled	Enabled	Uncheck this checkbox to allow anonymous login on OPC UA server.
Security Policy	None Basic256 (deprecated) ⁽¹⁾ Basic256Sha256	Basic256- Sha256	This drop-down menu allows you to secure your exchanges by signing and encrypting the data you send and receive.
Message Security	None Sign SignAndEncrypt	SignAndEncrypt	The messages are related to the Security Policy selected.
(1) Security policies marked as deprecated are policies which no longer afford an acceptable level of security.			

Server Configuration

Parameter	Value	Default value	Description
Server port	1...65535	4840	The port number of the OPC UA server. OPC UA clients must append this port number to the TCP URL of the controller to connect to the OPC UA server.
Max. subscriptions per session	1...100	20	Specify the maximum number of subscriptions allowed within each session.
Min. publishing interval	200...5000	1000	The publishing interval defines how frequently the OPC UA server sends notification packages to clients. Specify the minimum time that must elapse between notifications, in ms.
Max. monitored items per subscription	1...1000	100	The maximum number of <i>monitored items</i> in each subscription that the server assembles into a notification package.
Min. KeepAlive interval	500...5000	500	The OPC UA server only sends notifications when the values of monitored items of data are modified. A <i>KeepAlive</i> notification is an empty notification sent by the server to inform the client that although no data has been modified, the subscription is still active. Specify the minimum interval between KeepAlive notifications, in ms.
Max. number of sessions	1...4	2	The maximum number of clients that can connect simultaneously to the OPC UA server.
Identifier type	String	String	Certain OPC UA clients require a specific format of unique symbol identifier (node ID).

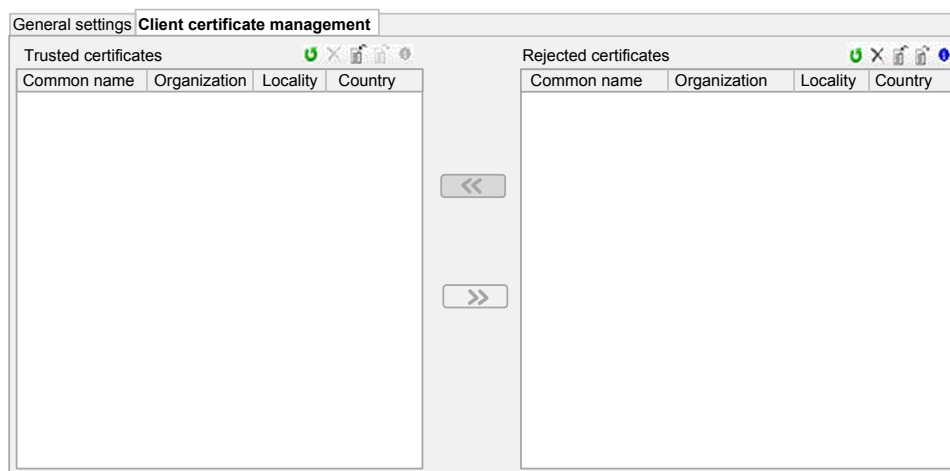
Diagnostic

Parameter	Value	Default value	Description
Enable trace	Enabled/disabled	Enabled	Select this checkbox to include OPC UA diagnostic messages in the controller log file (see EcoStruxure Machine Expert, Programming Guide). Traces are available from the Log tab or from the System Log File of the Web server. You can select the category of events to write to the log file: • None • Error • Warning • System • Info • Debug • Content • All (default)
Sampling rates (ms)	200...5000	500 1000 2000	The sampling rate indicates a time interval, in milliseconds (ms). When this interval has elapsed, the server sends the notification package to the client. The sampling rate can be shorter than the publishing interval, in which case notifications are queued until the publishing interval has elapsed. Sampling rates must be in the range 200...5000 (ms). Up to 3 different sampling rates can be configured. Double-click on a sampling rate to edit its value. To add a sampling rate to the list, right-click and choose Add a new rate. To remove a sample rate from the list, select the value and click .

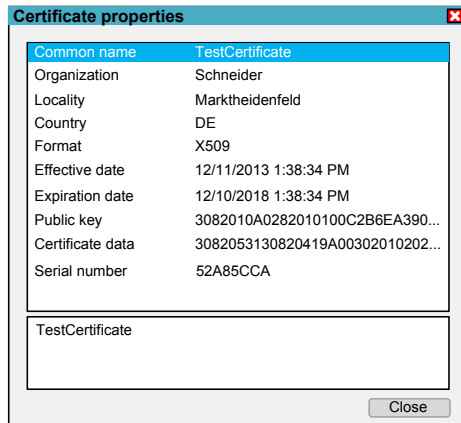
Click **Reset to default** to return the configuration parameters on this window to their default values.

Client Certificate Management Tab

This tab allows you to determine which OPC UA client certificates are trusted by the M262 Logic/Motion Controller OPC UA server.



Client Certificate Management Tab, Toolbar

Element	Description
	Both certificate lists are loaded or refreshed.
	Deletes the selected certificates.
	Opens a Windows dialog box (Open) to import a certificate that is uploaded to the selected certificate list (trusted certificates list or rejected certificates list).
	Opens a Windows dialog box (Save as) to export the selected certificates to a selectable path.
	Opens a dialog box containing additional information on the selected certificate. 

Trusted Certificates List and Rejected Certificates List

A certificate contains common information about the company that owns the certificate, how long a certificate is valid, and so on. The certificate management provides two list views:

- trusted certificates
- rejected certificates.

Element	Description
Trusted certificates	This list includes the client certificates the server trusts.
Rejected certificated	This list includes the client certificates the server does not trust.
 	Use the << and >> buttons to move a rejected certificate to the Trusted certificates list or the opposite way. During the moving procedure, a progress bar appears and displays the remaining files.

NOTE: OPC UA client and server share the same default PKI folder structure, including the trusted and untrusted (rejected) folders, this means trusting or untrusting (rejecting) a certificate has the same effect for both client and server.

NOTE: The OPC UA self-signed certificate has a limitation when the network interface through which OPC UA communicates is using dynamic IP addresses (DHCP). If you configured DHCP in such interface, make sure your OPC UA peer accepts the M262 Logic/Motion Controller OPC UA self-signed certificate without validation.

OPC UA Certificates Management Actions

This table describes each action concerning OPC UA certificates management and how to achieve it.

Action / Task	EcoStruxure Machine Expert Security Screen ⁽¹⁾	EcoStruxure Machine Expert M262 Files Screen ⁽²⁾	EcoStruxure Machine Expert M262 OPC UA Server Screen ⁽³⁾	M262 Webpage Maintenance - Certificates	FTP Protocol ⁽²⁾
Access to M262 OPC UA PKI folders	YES	YES	YES	NO	YES
Import a certificate	YES	YES	YES	NO	YES
Export a Certificate	YES	YES	YES	NO	YES
Remove a certificate	YES	YES	YES	NO	YES
Trust / Untrust a certificate	NO	YES ⁽⁴⁾	YES	YES ⁽⁵⁾	YES ⁽⁴⁾
Check a certificate information	YES	NO	YES	NO	NO
PKI: Public Key Infrastructure. (1) Only for M262 own certificates folder. (2) Except M262 own certificates folder. (3) Only trusted and untrusted (rejected) certificate folders. (4) Requires to manually move the certificate from the trusted folder to the untrusted (rejected) folder (and vice-versa). (5) Requires Administrator access.					

OPC UA PKI Folder List and Usage

The table describes the Public Key Infrastructure (PKI) shared between the M262 Logic/Motion Controller OPC UA server and OPC UA Client. It provides the folder list and their usage.

M262 File System Folders	Description
/usr/pki	Root folder of the default PKI
/usr/pki/issuer/certs	Contains Certificate Authority (CA) certificates that are required to validate Certification Paths
/usr/pki/issuer/crl	Contains Certificate Revocation Lists (CRL) for CA certificates
/usr/pki/trusted/certs	Contains Trusted certificates
/usr/pki/trusted/crl	Contains Certificate Revocation Lists (CRL) for the Trusted certificates
/usr/pki/untrusted	Contains Untrusted certificates
/usr/pki/quarantine	not used for M262 OPC UA (legacy for other services)
NOTE: Some of the PKI folders are only available after downloading the application enabling OPC UA (server/client), since some folders are only created in the runtime initialization of OPC UA.	

OPC UA Server Symbols Configuration

Introduction

Symbols are the items of data shared with OPC UA clients. Symbols are selected from a list of all the IEC variables used in the application. The selected symbols are then sent to the controller as part of the application download.

Each symbol is assigned a unique identifier. Identifiers are in string format.

This table describes IEC variable Base Types versus OPC UA Data Types:

IEC variable Base Types	OPC UA Data Types
BOOL, BIT	Boolean
BYTE, USINT	Byte
INT	Int16
WORD, UINT	UInt16
DINT, TOD, TIME	Int32
DWORD, UDINT	UInt32
LINT, LWORD, LTIME	Int64
ULINT	UInt64
REAL	Float
LREAL	Double
WSTRING, STRING	Up to 255 characters - String
DATE, DT	Second precision - DateTime
SINT	SByte

Bit memory variables (%MX) cannot be selected. In addition to IEC base data types, the OPC UA server can also expose OPC UA variables from IEC symbols that are composed of the following complex types :

- Arrays and Multi-Dimensional Arrays. These are limited to 3 dimensions.
- Structured data types, and nested structured data types. As long as they are not composed of a UNION field.

Displaying the List of Variables

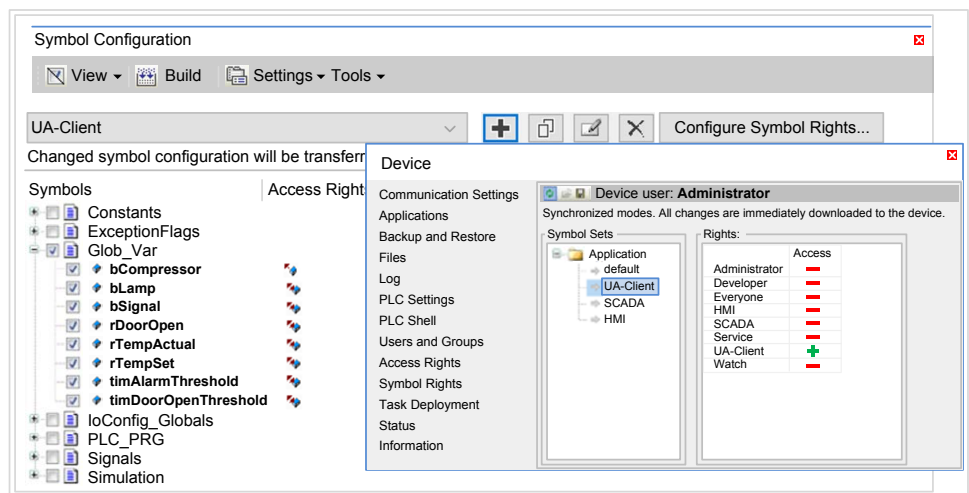
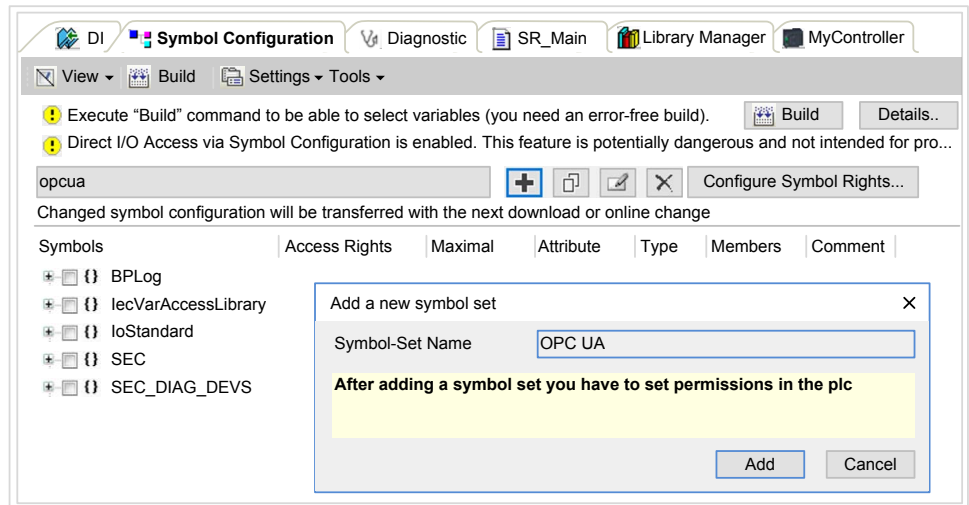
To display the list of variables:

Step	Action
1	On the Applications tree tab, right-click Application and choose Add object > Symbol Configuration . Result: The Add Symbol Configuration window is displayed. The controller starts the OPC UA server.
2	Click Add .

NOTE: The IEC objects %MX, %IX, %QX are not directly accessible. To access IEC objects you must first group their contents in located registers (refer to Relocation Table, page 30).

Selecting OPC UA Server Symbols

The **Symbol Configuration** window displays the variables available for selection as symbols:



Select **loConfig_Globals_Mapping** to select all the available variables. Otherwise, select individual symbols to share with OPC UA clients.

Each symbol has the following properties:

Name	Description
Symbols	The variable name followed by the address of the variable.
Type	The data type of the variable.
Access type	Click repeatedly to specify the access rights of the symbol: read-only (), write-only (), or read/write (). NOTE: Click in the Access type column of loConfig_Globals_Mapping to set the access rights of all the symbols at once.
Comment	An optional comment.

Click **Refresh** to update the list of available variables.

OPC UA Server Performance

Overview

As an example, the following provides capacity and performance information for the OPC UA server of the M262 Logic Controller. Design considerations are also provided to help you consider the optimal conditions for the performance of the OPC UA server. Of course, the performance realized by your application depend on many variables and conditions, and may differ from this example.

NOTE: These values are only valid if the OPC UA client function is not used.

System Configurations Used to Evaluate Performance

OPC UA server performance is determined by the system configuration, the number of symbols being published, and the percentage of symbols being refreshed.

The following table presents the number of elements small and medium configurations used for evaluating OPC UA server performance:

Elements	Small	Medium
TM3 expansion modules	0	4
CANopen slave devices	0	0
PTO functions	0	0
HSC functions	0	0
Profibus connections	0	0
Modbus TCP slave devices	0	0
Sercos devices	0	0
Incremental hardware encoders	0	0
Serial lines	0	1
EtherNet/IP adapters	0	1
EtherNet/IP Scanner devices	0	18
Generic TCP/UDP Managers	0	0

This table presents average read/write request times for the sample configurations and for different numbers of symbols:

Average Read/Write Request Times							
Configuration		Number of Symbols					
		50	100	250	400	500	1000
TM262L10MESE8T	Small	6 ms	11 ms	26 ms	41 ms	53 ms	132 ms
TM262L10MESE8T	Medium	16 ms	29 ms	71 ms	117 ms	149 ms	350 ms
TM262L20MESE8T	Small	3 ms	5 ms	12 ms	18 ms	23 ms	56 ms
TM262L20MESE8T	Medium	14 ms	23 ms	51 ms	80 ms	103 ms	123 ms

The following tables present the average time required to refresh a monitored set of symbols using a sampling rate of 200 ms and a publishing interval of 200 ms.

This table presents the average time required to refresh 100% of symbols for each of the sample configurations:

Average Time to Refresh 100% of Symbols				
Configuration		Number of Symbols		
		100	400	1000
TM262L10MESE8T	Small	204 ms	207 ms	218 ms
TM262L10MESE8T	Medium	197 ms	209 ms	680 ms
TM262L20MESE8T	Small	201 ms	203 ms	201 ms
TM262L20MESE8T	Medium	202 ms	205 ms	215 ms

This table presents the average time required to refresh 50% of symbols for each of the sample configurations:

Average Time to Refresh 50% of Symbols				
Configuration		Number of Symbols		
		100	400	1000
TM262L10MESE8T	Small	203 ms	204 ms	208 ms
TM262L10MESE8T	Medium	195 ms	201 ms	623 ms
TM262L20MESE8T	Small	201 ms	202 ms	204 ms
TM262L20MESE8T	Medium	202 ms	203 ms	207 ms

This table presents the average time required to refresh 1% of symbols for each of the sample configurations:

Average Time to Refresh 1% of Symbols				
Configuration		Number of Symbols		
		100	400	1000
TM262L10MESE8T	Small	201 ms	202 ms	202 ms
TM262L10MESE8T	Medium	194 ms	196 ms	285 ms
TM262L20MESE8T	Small	200 ms	201 ms	201 ms
TM262L20MESE8T	Medium	201 ms	202 ms	202 ms

Optimizing OPC UA Server Performance

The OPC UA server functionality is dependent on external communication networks, external device performance, and other external parameters. Data transmitted may be delayed or other possible communication errors may arise that impose practical limits on machine control. Do not use the OPC UA server functionality for safety-related data or other time-dependent purposes.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Do not allow safety-related data in OPC UA server data exchanges.
- Do not use OPC UA server data exchanges for any critical or time-dependent purposes.
- Do not use OPC UA server data exchanges to change equipment states without having done a risk analysis and implementing appropriate safety-related measures.

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

The above tables can be useful in determining whether OPC UA server performance is within acceptable limits. Be aware, however, that other external factors influence overall system performance, such as the volume of Ethernet traffic.

To optimize OPC UA server performance, consider the following:

- Minimize Ethernet traffic by setting the **Min. publishing interval** to the lowest value that yields an acceptable response time.
- The **task cycle time**, page 33 configured for the controller must be less than the configured **Min. publishing interval** value.
- Configuring a **Max. number of sessions** (the number of OPC UA clients that can simultaneously connect to the OPC UA server) value of greater than 1 decreases the performance of all sessions.
- The sampling rate determines the frequency at which data is exchanged. Tune the **Sampling rates (ms)** value to product the lowest response time that does not adversely affect the overall performance of the controller.

OPC UA Client Configuration

OPC UA Client Overview

Introduction

The OPC Unified Architecture client (OPC UA client) allows the TM262L20MESE8T, the TM262M25MESS8T and the TM262M35MESS8T to exchange data with remote OPC UA servers.

NOTE: At least one Ethernet interface (**Ethernet_1** or **Ethernet_2**) must be available on the controller and Ethernet traffic not blocked by the firewall, page 142.

The OPC UA client can connect with up to 5 servers concurrently. Each server can exchange 5000 items, with a maximum of 15000 items for all the servers.

Scope of the Security Model

The OPC UA client provides:

- Support of binary message encoding
- Support of message encryption and integrity :
 - **None**, **Basic256** and **Basic256Sha256** security policies.
 - **None**, **Sign** and **Sign and Encrypt** message security modes.
- Support of user authentication:
 - **None (Anonymous)** or user name and password.
- Support of user authorization:
 - To read and write value of a node.
 - To browse the information model based on the access rights of the information model, the user or the user's role.

User and Group Access Rights

Access to OPC UA connections and data is controlled by user rights. Refer to Users Rights, page 69.

OPC UA Services

The following OPC Foundation Profiles and facets are implemented:

- OPC UA Minimum Client Profile
 - SecurityPolicy – None
 - User Token – Anonymous Facet
 - UA-TCP UA-SC UA-Binary
 - User Token – User Name Password
- Security Category
 - SecurityPolicy – **Basic256Sha256**
 - SecurityPolicy – **Basic256**
- Data Access Facet
 - Attribute Read Client Facet
 - Attribute Write Client Facet
 - DataChange Subscriber Client Facet

The following functionalities are supported:

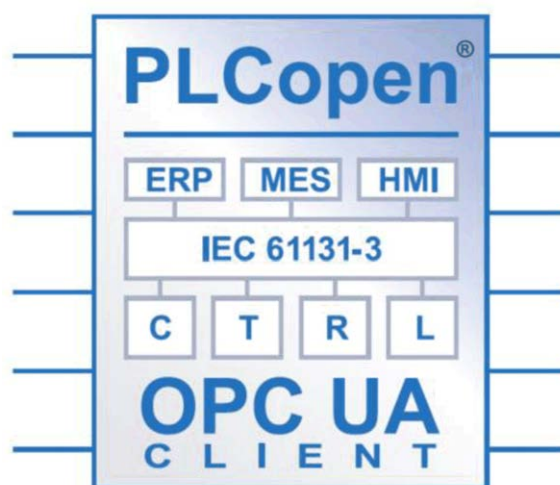
- Attribute Services (for Value attribute only)
- Discovery Services (Configure Endpoint only)
- Monitored Item Services
- Protocol and Encoding (TCP UA Binary)
- Security
- Session Services
- Subscription services
- View Services

Programming the OPC UA Client

Overview

OPC UA client functionality is delivered in the *OpcUaHandling* library.

This library contains IEC 61131-3 standard function blocks to include in your application:



The function blocks allow you to:

- Read/write multiple data items
- Perform diagnostics

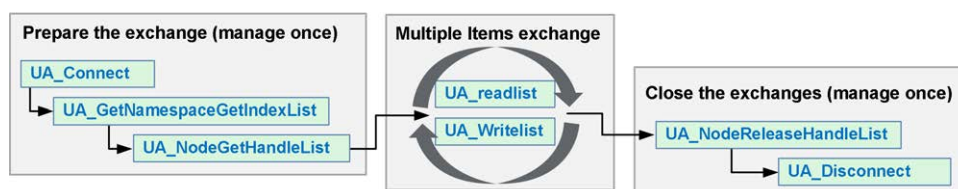
The following function blocks are supported:

- UA_Connect
- UA_ConnectionGetStatus
- UA_Disconnect
- UA_NamespaceGetIndexList
- UA_NodeGetHandleList
- UA_NodeGetInformation
- UA_NodeReleaseHandleList
- UA_ReadList
- UA_WriteList
- UA_Browse
- UA_SubscriptionCreate
- UA_SubscriptionDelete
- UA_SubscriptionProceed
- UA_MonitoredItemAddList
- UA_MonitoredItemOperateList
- UA_MonitoredItemRemoveList
- UA_TranslatePathList
- FB_TimeStamper

For details, refer to the OpcUaHandling Library Guide (see EcoStruxure Machine Expert, OpcUaHandling, Library Guide).

Example: Managing a Read/Write List

This figure shows the function blocks used to read and write items of data managed by a remote OPC UA server:



NOTE: Ensure that **OPC UA Server enabled** is activated to use OPC UA Client. See OPC UA Server Configuration Tab, page 203.

Post Configuration

Introduction

This chapter describes how to generate and configure the post configuration file of the Modicon M262 Logic/Motion Controller.

Post Configuration Presentation

Introduction

Post configuration is an option that allows you to modify some parameters of the application without changing the application. Post configuration parameters are defined in a file called **Machine.cfg**, which is stored in the controller.

By default, all parameters are set in the application. The parameters defined in the Post Configuration file are used instead of the corresponding parameters defined in the application.

Parameters

The Post Configuration file allows you to change network parameters.

Ethernet parameters:

- IP Address
- Subnet Mask
- Gateway Address
- IP Config Mode
- Device Name

Serial Line parameters, for each serial line in the application (embedded port or PCI module):

- Baud rate
- Parity
- Data bits
- Stop bit

FTP Server Encryption:

- FTP Server Encryption setting parameter

WebVisualisation:

- WebVisualisation connection type (encryption enforcing parameter)

CAE Enable:

- Cybersecurity Admin Expert parameter

ODVA Enable:

- ODVA parameter

NOTE: Parameter updates with a Post Configuration file that impacts parameters used by other devices via a communication port are not updated in the other devices.

For example, if the IP address used by an HMI is updated in the configuration with a Post Configuration file, the HMI uses the previous address. You must update the address used by the HMI independently.

Operating Mode

The Post Configuration file is read after:

- A Reset Warm command, page 51
- A Reset Cold command, page 52
- A reboot, page 55
- An application download, page 57

Refer to Controller States and Behaviors, page 41 for further details on controller states and transitions.

NOTE: The post configuration is ignored for applications where a scanner is configured.

Post Configuration File Management

Introduction

The file **Machine.cfg** is located in the directory `/usr/cfg`.

Each parameter is specified by a variable type, variable ID, and value. The format is:

```
id[moduleType].pos[param1Id].id[param2Id].param[param3Id].
paramField=value
```

Each parameter is defined on three lines in the Post Configuration file:

- The first line describes the internal 'path' for this parameter.
- The second line is a comment describing the parameter.
- The third line is the definition of the parameter (as described above) with its value.

Post Configuration File Generation

The Post Configuration file (**Machine.cfg**) is generated by EcoStruxure Machine Expert.

To generate the file, proceed as follows:

Step	Action
1	In the menu bar, choose Build > Generate... Result: An explorer window is displayed.
2	Select the destination folder of the Post Configuration file.
3	Click OK .

When you use EcoStruxure Machine Expert to create a Post Configuration file (**Generate**), it reads the value of each parameter assigned in your application program and then writes the values to the **Machine.cfg** Post Configuration file. After generating a Post Configuration file, review the file and remove any parameter assignments that you wish to remain under the control of your application. Keep only those parameter assignments that you wish changed by the Post Configuration function that are necessary to make your application portable and then modify those values appropriately.

Post Configuration File Transfer

After creating and modifying your Post Configuration file, transfer it to the `/usr/cfg` directory of the controller. The controller does not read the **Machine.cfg** file unless it is in this directory.

You can transfer the Post Configuration file by the following methods:

- SD card, page 235 (with the proper script)
- Download through the FTP server, page 120
- Download with EcoStruxure Machine Expert Controller Device Editor, page 60

Modifying a Post Configuration File

If the Post Configuration file is located in the PC, use a text editor to modify it.

NOTE: Do not change the text file encoding. The default encoding is ANSI.

To modify the Post Configuration file directly in the controller, use the **Setup** menu of the Web server, page 122.

To modify the Post Configuration file in the controller with EcoStruxure Machine Expert in online mode:

Step	Action
1	In the Devices tree , click the controller name.
2	Click Build > Post Configuration > Edit... Result: The Post Configuration file opens in a text editor.
3	Edit the file.
4	If you want to apply the modifications after saving them, select Reset device after sending .
5	Click Save as .
6	Click Close .

NOTE: If the parameters are invalid, they are ignored.

Deleting the Post Configuration File

You can delete the Post Configuration file by the following methods:

- SD card (with the delete script)
- Through the FTP server, page 120
- Online with EcoStruxure Machine Expert controller device editor, page 60, **Files** tab

For more information on **Files** tab of the Device Editor, refer to EcoStruxure Machine Expert Programming Guide.

NOTE: The parameters defined in the application are used instead of the corresponding parameters defined in the Post Configuration file after:

- A Reset Warm command, page 51
- A Reset Cold command, page 52
- A reboot, page 55
- An application download, page 57

Post Configuration Example

Post Configuration File Example

```
# TM262M25MESS8T / RNDIS USB address
# RNDIS USB address
.param[1104] = [192, 168, 200, 1]

# TM262M25MESS8T / RNDIS USB mask
# RNDIS USB mask
.param[1105] = [255, 255, 255, 0]

# TM262M25MESS8T / FTP Server Encryption
# 1=encryption enforced, 0=otherwise
.param[1106] = 1

# TM262M25MESS8T / WebVisu Connection Type
# 0=Only HTTP connections are supported, 1 = Only HTTPS
connections are supported, 2 = HTTP and HTTPS connections are
supported, 3 = HTTP connections are redirected to HTTPS
.param[1107] = 3

# TM262M25MESS8T / CAE Enable
# 1=Enabled, 0=Disabled
.param[1108] = 1

# TM262M25MESS8T / Advanced ODVA features Enable
# 1=Enabled, 0=Disabled
.param[1109] = 0

# TM262M25MESS8T / Ethernet_1 / IPAddress
# Ethernet IP address
id[45000].pos[5].id[111].param[0] = [192, 168, 1, 3]]

# TM262M25MESS8T / Ethernet_1 / SubnetMask
# Ethernet IP mask
id[45000].pos[5].id[111].param[1] = [255, 255, 255, 0]]

# TM262M25MESS8T / Ethernet_1 / GatewayAddress
# Ethernet IP gateway address
id[45000].pos[5].id[111].param[2] = [0, 0, 0, 0]]
```



```
# TM262M25MESS8T / Ethernet_1 / IPConfigMode
# IP configuration mode: 0:FIXED 1:BOOTP 2:DHCP
id[45000].pos[5].id[111].param[4] = 0

# TM262M25MESS8T / Ethernet_1 / DeviceName
# Name of the device on the Ethernet network
id[45000].pos[5].id[111].param[5] = 'my_Device'

# TM262M25MESS8T / Ethernet_2 / IPAddress
# Ethernet IP address
id[45000].pos[6].id[45111].param[0] = [192, 168, 102, 2]

# TM262M25MESS8T / Ethernet_2 / SubnetMask
# Ethernet IP mask
id[45000].pos[6].id[45111].param[1] = [255, 255, 255, 0]

# TM262M25MESS8T / Ethernet_2 / GatewayAddress
# Ethernet IP gateway address
id[45000].pos[6].id[45111].param[2] = [0, 0, 0, 0]

# TM262M25MESS8T / Ethernet_2 / IPConfigMode
# IP configuration mode: 0:FIXED 1:BOOTP 2:DHCP
id[45000].pos[6].id[45111].param[4] = 0

# TM262M25MESS8T / Ethernet_2 / DeviceName
# Name of the device on the Ethernet network
id[45000].pos[6].id[45111].param[5] = 'my_Device'

# TM262M25MESS8T / Serial_Line / Serial Line Configuration /
# Baudrate
# Serial Line Baud Rate in bit/s
id[45000].pos[7].id[40101].param[10000].Bauds = 19200

# TM262M25MESS8T / Serial_Line / Serial Line Configuration /
# Parity
# Serial Line Parity (0=None, 1=Odd, 2=Even)
id[45000].pos[7].id[40101].param[10000].Parity = 2

# TM262M25MESS8T / Serial_Line / Serial Line Configuration /
# DataBits
```

```
# Serial Line Data bits (7 or 8)
id[45000].pos[7].id[40101].param[10000].DataFormat = 8

# TM262M25MESS8T / Serial_Line / Serial Line Configuration /
StopBits

# Serial Line Stop bits (1 or 2)
id[45000].pos[7].id[40101].param[10000].StopBit = 1
```

Connecting a Modicon M262 Logic/Motion Controller to a PC

Introduction

This chapter shows how to connect a Modicon M262 Logic/Motion Controller to a PC.

Connecting the Controller to a PC

Overview

To transfer, run, and monitor the applications, you can use either a USB cable or an Ethernet connection to connect the controller to a computer with EcoStruxure Machine Expert installed.

NOTICE

INOPERABLE EQUIPMENT

Always connect the communication cable to the PC before connecting it to the controller.

Failure to follow these instructions can result in equipment damage.

USB Mini-B Port Connection

Cable Reference	Details
BMXXCAUSBH018	Grounded and shielded, this USB cable is suitable for long duration connections.
TCSXCNAMUM3P	This USB cable is suitable for short duration connections such as quick updates or retrieving data values.

NOTE: You can only connect 1 controller or any other device associated with EcoStruxure Machine Expert and its component to the PC at any one time.

The USB Mini-B Port is the programming port you can use to connect a PC with a USB host port using EcoStruxure Machine Expert software. Using a typical USB cable, this connection is suitable for quick updates of the program or short duration connections to perform maintenance and inspect data values. It is not suitable for long-term connections such as commissioning or monitoring without the use of specially adapted cables to help minimize electromagnetic interference.

⚠ WARNING

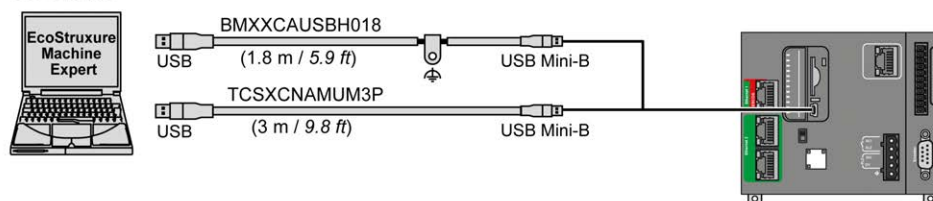
UNINTENDED EQUIPMENT OPERATION OR INOPERABLE EQUIPMENT

- You must use a shielded USB cable such as a BMX XCAUSBH0•• secured to the functional ground (FE) of the system for any long-term connection.
- Do not connect more than one controller or bus coupler at a time using USB connections.
- Do not use the USB port(s), if so equipped, unless the location is known to be non-hazardous.

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

The communication cable should be connected to the PC first to minimize the possibility of electrostatic discharge affecting the controller.

USB mini-B

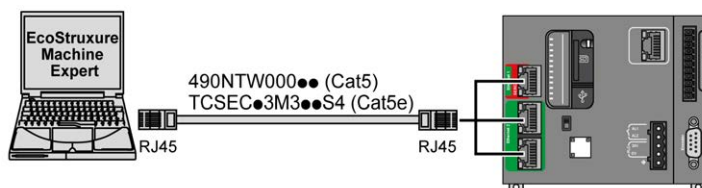


To connect the USB cable to your controller, follow the steps below:

Step	Action
1	1a. If making a long-term connection using the cable BMXXCAUSBH018, or other cable with a ground shield connection, be sure to securely connect the shield connector to the functional ground (FE) or protective ground (PE) of your system before connecting the cable to your controller and your PC. 1b. If making a short-term connection using the cable TCSXCNAMUM3P or other non-grounded USB cable, proceed to step 2.
2	Connect your USB cable to the computer.
3	Open the protective cover for the USB mini-B slot on the controller.
4	Connect the mini-B connector of your USB cable to the controller.

Ethernet Port Connection

You can also connect the controller to a PC using an Ethernet cable.



To connect the controller to the PC, do the following:

Step	Action
1	Connect the Ethernet cable to the PC.
2	Connect the Ethernet cable to any of the Ethernet ports on the controller.

Updating Firmware

Introduction

Updating the controller firmware is possible by using:

- An SD card with a compatible script file.
- The Controller Assistant.

Updating the TM3 and the TMS firmware is possible by using an SD card with a compatible script file.

Performing a firmware update deletes the application program in the device, including the configuration files, the user management, the user rights, the certificates and the Boot Application in non-volatile memory.

For more information about the firmware update and creating a new flash disk with firmware, refer to Project Settings - Firmware Update and Non-Volatile Memory Organization, page 26.

Updating the Controller Firmware by SD Card

Before Updating Firmware

The Modicon M262 Logic/Motion Controller accepts only SD cards formatted in FAT or FAT32.

The SD card must have a label. To add a label:

1. Insert the SD card in your PC.
2. Right-click on the drive in Windows Explorer.
3. Choose **Properties**.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have operational knowledge of your machine or process before connecting this device to your controller.
- Ensure that guards are in place so that any potential unintended equipment operation will not cause injury to personnel or damage to equipment.

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

When an SD card is inserted into the SD card slot of the controller, the firmware searches and executes the script contained in the SD card (/sys/cmd/Script.cmd).

Performing a firmware update deletes the application program in the device, including the configuration files, the user management, the user rights, the certificates and the Boot Application in non-volatile memory.

NOTICE

LOSS OF APPLICATION DATA

- Perform a backup of the application program to the hard disk of the PC before attempting a firmware update.
- Restore the application program to the device after a successful firmware update.

Failure to follow these instructions can result in equipment damage.

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.

The serial line ports of your controller are configured for the Machine Expert protocol by default when new or when you update the controller firmware. The Machine Expert protocol is incompatible with that of other protocols such as Modbus Serial Line. Connecting a new controller to, or updating the firmware of a controller connected to, an active Modbus configured serial line can cause the other devices on the serial line to stop communicating. Make sure that the controller is not connected to an active Modbus serial line network before first downloading a valid application having the concerned port or ports properly configured for the intended protocol.

NOTICE

INTERRUPTION OF SERIAL LINE COMMUNICATIONS

Be sure that your application has the serial line ports properly configured for Modbus before physically connecting the controller to an operational Modbus Serial Line network.

Failure to follow these instructions can result in equipment damage.

Updating Firmware

Follow these steps to update the firmware by an SD card:

Step	Action
1	Download the firmware updates for Modicon M262 Logic/Motion Controller on the Schneider Electric website (in .zip format).
2	Extract the .zip file to the root of the SD card. NOTE: The SD card folder \sys\cmd\ contains the download script file.
3	Remove power from the controller.
4	Insert the SD card into the controller.
5	Restore power to the controller. NOTE: The SD LED (green) is flashing during the operation.
6	Wait until the end of the download: • If the SD LED (green) is ON, the download ended successfully. • If the SD LED (yellow) is ON, an error is detected. A script.log file is created in the SD card \sys\cmd\ folder. Contact your Schneider Electric local support.
7	Remove the SD card from the controller. Result: If the download ended successfully, the controller restarts automatically with new firmware. The restart is longer than usual.

Updating the Controller Firmware by Controller Assistant

Before Updating Firmware

Performing a firmware update deletes the application program in the device, including the configuration files, the user management, the user rights, the certificates and the Boot Application in non-volatile memory.

NOTICE

LOSS OF APPLICATION DATA

- Perform a backup of the application program to the hard disk of the PC before attempting a firmware update.
- Restore the application program to the device after a successful firmware update.

Failure to follow these instructions can result in equipment damage.

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.

The serial line ports of your controller are configured for the Machine Expert protocol by default when new or when you update the controller firmware. The Machine Expert protocol is incompatible with that of other protocols such as Modbus Serial Line. Connecting a new controller to, or updating the firmware of a controller connected to, an active Modbus configured serial line can cause the other devices on the serial line to stop communicating. Make sure that the controller is not connected to an active Modbus serial line network before first downloading a valid application having the concerned port or ports properly configured for the intended protocol.

NOTICE

INTERRUPTION OF SERIAL LINE COMMUNICATIONS

Be sure that your application has the serial line ports properly configured for Modbus before physically connecting the controller to an operational Modbus Serial Line network.

Failure to follow these instructions can result in equipment damage.

Updating Firmware

You have two ways to update the firmware by the Controller Assistant:

- Using an SD card

- Writing on the controller

To execute a complete firmware update of a controller, using an SD card, replacing the Boot application and data, in offline mode, proceed as follows:

Step	Action
1	Insert an empty SD card into the PC.
2	Click Tools > External Tools > Open controller Assistant .
3	On the Home dialog box, click the Update firmware.... button. Result: The Update firmware (step 1 from 4) dialog box is displayed.
4	Select the Controller type and the Controller firmware version .
5	Click the Next button. Result: The Update firmware (step 2 from 4) dialog box is displayed.
6	If needed, change the Communication settings and click the Next button. Result: The Update firmware (step 3 from 4) dialog box is displayed.
7	Click the Write to... button. Result: The Update firmware (step 4 from 4) dialog box is displayed.
8	Select your SD card in Disk drive and click the Write button. At the end of the writing, the Home dialog box is displayed.

To execute a complete firmware update of a controller, replacing the Boot application and data, writing on the controller in online mode, proceed as follows:

Step	Action
1	Click Tools > External Tools > Open controller Assistant .
2	On the Home dialog box, click the Update firmware.... button. Result: The Update firmware (step 1 from 4) dialog box is displayed.
3	Select the Controller type and the Controller firmware version .
4	Click the Next button. Result: The Update firmware (step 2 from 4) dialog box is displayed.
5	If needed, change the Communication settings and click the Next button. Result: The Update firmware (step 3 from 4) dialog box is displayed.
6	Click the Write on controller... button. Result: The Update firmware (step 4 from 4) dialog box is displayed.
7	Select the Controller and click the Connect button. Result: The controller is in STOPPED state. At the end of the writing, the Home dialog box is displayed. A message is displayed to indicate that you must reboot the controller.

Updating TM3 Expansion Modules Firmware

Overview

The firmware update for the controller and the expansion modules are available on the [Schneider Electric website](#) (in .zip format).

Downloading Firmware to TM3 Expansion Modules

The firmware can be updated in:

- TM3X•HSC•
- TM3D• with firmware version (SV) ≥ 2.0 , except TM3DM16R and TM3DM32R
- TM3A• and TM3T• with firmware version (SV) ≥ 2.0

NOTE: The firmware version (SV) is found on the packaging and product labels.

Firmware updates are performed if, during a power on, at least one firmware file is present in the `/usr/TM3fwupdate/` directory of controller. You can download the file(s) to the controller using the SD card, an FTP file transfer or through EcoStruxure Machine Expert.

The controller updates the firmware of the TM3 expansion modules on the I/O bus, including those that are:

- Connected remotely, using a TM3 Transmitter/Receiver module.
- In configurations comprising a mix of TM3 and TM2 expansion modules.

The following table describes how to download firmware to one or more TM3 expansion modules using an SD card:

Step	Action
1	Insert an empty SD card into the PC.
2	Create the folder path <code>/sys/Cmd</code> and create a file called <i>Script.cmd</i> .
3	Edit the file and insert the following command for each firmware file you wish to transfer to the controller: <code>Download "usr/TM3fwupdate/<filename>"</code>
4	Create the folder path <code>/usr/TM3fwupdate/</code> in the SD card root directory and copy the firmware files to the <i>TM3fwupdate</i> folder.
5	Ensure that power is removed from controller.
6	Remove the SD card from the PC and insert it into the SD card slot of the controller.
7	Restore power to the controller. Wait until the end of the operation (until the SD LED is green ON). Result: The controller begins transferring the firmware file(s) from the SD card to the <code>/usr/TM3fwupdate</code> in the controller. During this operation, the SD LED on the controller is flashing. A <i>SCRIPT.log</i> file is created on the SD card and contains the result of the file transfer. If an error is detected, the SD and ERR LEDs flash and the detected error is logged in <i>SCRIPT.log</i> file.
8	Remove power from the controller.
9	Remove SD card from the controller.
10	Restore power to the controller. Result: The controller transfers the firmware file(s) to the appropriate TM3 I/O module(s). NOTE: The TM3 update process adds approximately 15 seconds to the controller boot duration.
11	Verify in the message logger of the controller that the firmware is successfully updated: <i>Your TM3 Module X successfully updated. X corresponds to the position of the module on the bus.</i> NOTE: You can also obtain the logger information in the <i>PlcLog.txt</i> file in the <code>/usr/Syslog/</code> directory of the controller file system. NOTE: If the controller encounters an error during the update, the update terminates with that module.
12	If all targeted modules were successfully updated, delete the firmware file(s) from <code>/usr/TM3fwupdate/</code> folder on the controller. You can delete the files directly using EcoStruxure Machine Expert or by creating and executing a script containing the following command: <code>Delete "usr/TM3fwupdate/*"</code> NOTE: If a targeted module was not updated successfully, or there are no message logger messages for all the targeted modules, see the <i>Recovery Procedure</i> , page 228.

Recovery Procedure

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.

If, during the reattempted firmware update, the update prematurely terminates with an error, it means that the communication interruption or power outage had damaged the firmware of one of your modules in your configuration, and that module must be reinitialized.

NOTE: Once the firmware update process detects an error with the firmware in the destination module, the update process is terminated. After you have reinitialized the damaged module following the recovery procedure, any modules that followed the damaged module remain unchanged and will need to have their firmware updated.

The following table describes how to reinitialize the firmware on TM3 expansion modules:

Step	Action
1	Ensure that the correct firmware is present in the <code>/usr/TM3fwupdate/</code> directory of the controller.
2	Remove power from the controller.
3	Disassemble from the controller all TM3 expansion modules that are functioning normally, up to the first module to recover. Refer to the hardware guides of the modules for disassembly instructions.
4	Apply power to the controller. NOTE: The TM3 update process adds approximately 15 seconds to the controller boot duration.
5	Verify in the message logger of the controller that the firmware is successfully updated: <code>Your TM3 Module X successfully updated</code> . X corresponds to the position of the module on the bus.
6	Remove power from the controller.
7	Reassemble the TM3 expansion module configuration to the controller. Refer to the hardware guides of the modules for assembly instructions.
8	Restore power to the controller. Result: The controller transfers the firmware file(s) to the appropriate and yet to be updated TM3 I/O module(s). NOTE: The TM3 update process adds approximately 15 seconds to the controller boot duration.
9	Verify in the message logger of the controller that the firmware is successfully updated: <code>Your TM3 Module X successfully updated</code> . X corresponds to the position of the module on the bus. NOTE: You can also obtain the logger information in the <code>Sys.log</code> file in the <code>/usr/Log</code> directory of the controller file system.
10	Delete the firmware file(s) from <code>/usr/TM3fwupdate/</code> folder on the controller.

Updating TMSES4 Expansion Module Firmware

Overview

The firmware update for the M262 Logic/Motion Controller are available on the Schneider Electric website (in .zip format).

Updating TMSES4 Module Firmware

The firmware can be updated in TMSES4 module.

Firmware updates are performed using a script file on an SD card.

When the SD card is inserted in the SD card slot of the controller, the controller updates the firmware of the TMSES4 expansion modules on the I/O bus

Follow these steps to update the firmware by an SD card:

Step	Action
1	Insert an empty SD card into the PC.
2	Create the folder path /TMS/ in the SD card root directory and copy the two .bin files to the TMS folder. NOTE: The smaller file is an information file for checking (reference, version...) which points to the larger file, containing the firmware itself.
3	Remove power from the controller.
4	Remove the SD card from the PC and insert it into the SD card slot of the controller.
5	Restore power to the controller. Result: The controller begins transferring the firmware file from the SD card to the updatable expansion modules. During this operation, the MOD STS LED on the module is fast flashing green. The firmware update takes until two minutes for each expansion module being updated. Do not remove power from the controller, or remove the SD card, while the operation is in progress. Otherwise, the firmware update may be unsuccessful, and the modules may no longer function correctly.
6	Wait until the end of the download. If the MOD STS LED of the module is: - Green ON, the download ended successfully. - Fast flashing red, an error is detected.
7	Verify in the message logger of the controller that the firmware is successfully updated.
8	Remove SD card from the controller.
9	Remove power from the controller, then restore power to the controller. Result: The controller restarts automatically with new firmware if the download ended successfully.

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.

Managing Script Files

Introduction

The following describes how to write script files (default script file or dynamic script file) to be executed from an SD card or by an application using the ExecuteScript function block (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide).

NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is `/usr/Syslog/FWLog.txt`.

NOTE: When User Rights are activated on a controller and access rights of the group **ExternalMedia** on objects **ExternalCmd** are denied, scripts used to **Upload/Download/Delete** files are disabled via SD card scripts (use of the ExecuteScript function block is unaffected by User Rights). For more details about User Rights, refer to the EcoStruxure Machine Expert Programming Guide.

Creating a Script

Introduction

The EcoStruxure Machine Expert script language provides a powerful tool to automate sequences. You can start single commands or complex command sequences directly from the EcoStruxure Machine Expert program environment. For more information on the script, refer to the EcoStruxure Machine Expert Programming Guide.

Before Creating Scripts Using an SD card

The Modicon M262 Logic/Motion Controller accepts only SD cards formatted in FAT or FAT32.

The SD card must have a label. To add a label:

1. Insert the SD card in your PC.
2. Right-click on the drive in Windows Explorer.
3. Choose **Properties**.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have operational knowledge of your machine or process before connecting this device to your controller.
- Ensure that guards are in place so that any potential unintended equipment operation will not cause injury to personnel or damage to equipment.

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

When an SD card is inserted into the SD card slot of the controller, the firmware searches and executes the script contained in the SD card (`/sys/cmd/Script.cmd`).

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

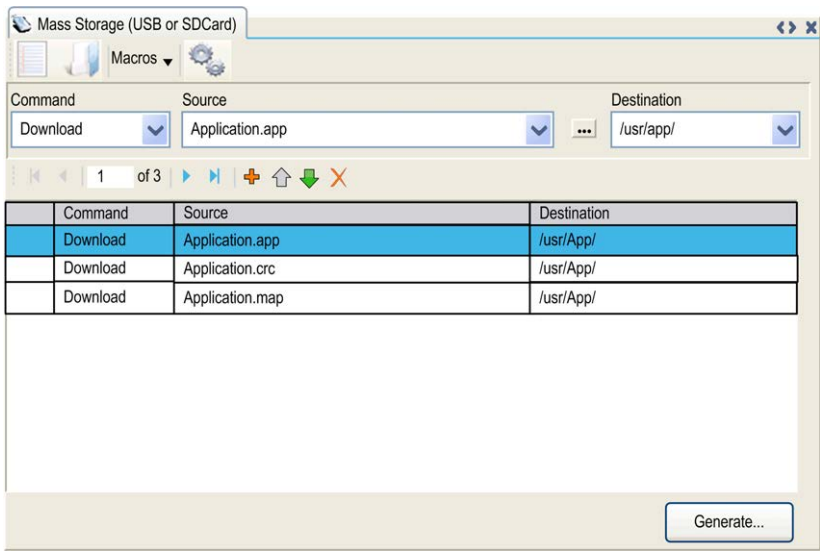
- Do not interrupt the transfer of the application program or a firmware change once the transfer has begun.
- Re-initiate the transfer if the transfer is interrupted for any reason.
- Do not attempt to place the device into service until the file transfer has completed successfully.

Failure to follow these instructions can result in equipment damage.

Creating a Script

The following lists the script syntax guidelines:

- If the line begins with a ";", the line is a comment.
- The maximum number of lines in a script file is 50.
- If the syntax is not respected in the script file, the script file is not executed. This means, for example, that the firewall configuration remains in the previous state.

Step	Action
1	Click Project > Mass Storage (USB or SDCard) in the main menu. Result: The Mass Storage (USB or SDCard) tab displays: 
2	Click  , then  .
3	Select a Command .
4	Depending on the selected command, select a Source and Destination .

Description of the Mass Storage (USB or SDCard) tab

This table describes the **Mass Storage (USB or SDCard)** tab:

Element	Description
New	Create a new script.
Open	Open a script.
Macros	Insert a Macro. A macro is a sequence of unitary commands. A macro helps to perform many common operations such as upload application, download application, and so on.
Generate	Generate the script and all necessary files on the SD card.
Command	Basic instructions.
Source	Source file path on the PC or the controller.
Destination	Destination directory on the PC or the controller.
Add New	Add a script command.
Move Up/Down	Change the script commands order.
Delete	Delete a script command.

This table describes the commands:

Command	Description	Source	Destination	Syntax
Download	Download a file from the SD card to the controller.	Select the file to download.	Select the controller destination directory.	'Download "/usr/Cfg/*"'
SetNode-Name	Sets the node name of the controller.	New node name.	Controller node name	'SetNodeName "Name_PLC"'
	Resets the node name of the controller.	Default node name.	Controller node name	'SetNodeName ""'
Upload	Upload files contained in a controller directory to the SD card.	Select the directory.	-	'Upload "/usr/*"'
Delete	Delete files contained in a controller directory. NOTE: Delete "*" does not delete system files.	Select the directory and enter a specific file name Important: by default, all directory files are selected.	-	'Delete "/usr/SysLog/*"'
Reboot	Restart the controller (only available at the end of the script).	-	-	'Reboot'

This table describes the macros:

Macros	Description	Directory/Files
Download App	Download the application from the SD card to the controller.	/usr/App/*.app
Upload App	Upload the application from the controller to the SD card.	/usr/App/*.crc
Download Sources	Download the project archive from the SD card to the controller.	/usr/App/*.prj
Upload Sources	Upload the project archive from the controller to the SD card.	
Download Multi-files	Download multiple files from the SD card to a controller directory.	Defined by user
Upload Log	Upload the log files from the controller to the SD card.	/usr/Log/*.log

Reset the User Rights to Default

You can manually create a script to remove the user rights, along with the application, from the controller. This script must contain this command:

Format "/usr"

Reboot

NOTE: This command also removes user application and data.

Step	Action
1	Remove power from the controller.
2	Insert the prepared SD card in the source controller.
3	Restore power to the source controller. Result: The copy starts automatically. During the copy, the PWR and I/O LEDs are ON and the SD LED flashes regularly.
4	Wait until the copy is completed. Result: The SD LED is ON and the controller reboots with default user rights. If an error was detected, the ERR LED is ON and the controller is in STOPPED state.

Generating Scripts and Files

Generating Existing Scripts and Files

Step	Action
1	Click Project > Mass Storage (USB or SDCard).... Result: The Mass Storage (USB or SDCard) tab displays.
2	Click Macros and select an action from the drop-down list.
3	Select the files to generate.
4	Click Generate....
5	Select the destination folder.

Generating New Scripts and Files

Step	Action
1	Click Project > Mass Storage (USB or SDCard)... Result: The Mass Storage (USB or SDCard) tab displays.
2	Create a script, page 231.
3	Select the files to generate.
4	Click Generate....
5	Select the destination folder.

Transferring Scripts and Files

Before Transferring Scripts and Files

You can transfer scripts and files from and to the controller using an SD card.

The Modicon M262 Logic/Motion Controller accepts only SD cards formatted in FAT or FAT32.

The SD card must have a label. To add a label:

1. Insert the SD card in your PC.
2. Right-click on the drive in Windows Explorer.
3. Choose **Properties**.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have operational knowledge of your machine or process before connecting this device to your controller.
- Ensure that guards are in place so that any potential unintended equipment operation will not cause injury to personnel or damage to equipment.

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

When an SD card is inserted into the SD card slot of the controller, the firmware searches and executes the script contained in the SD card (/sys/cmd/Script.cmd).

NOTE: The controller operation is not modified during file transfer.

Performing a firmware update deletes the application program in the device, including the configuration files, the user management, the user rights, the certificates and the Boot Application in non-volatile memory.

NOTICE

LOSS OF APPLICATION DATA

- Perform a backup of the application program to the hard disk of the PC before attempting a firmware update.
- Restore the application program to the device after a successful firmware update.

Failure to follow these instructions can result in equipment damage.

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.

Transferring

Step	Action
1	Create the script with the Mass Storage (USB or SDCard) editor. If necessary, refer to <i>Creating a Script</i> , page 231.
2	Click Generate... and select the SD card root directory. Result: The script and files are transferred on the SD card.
3	Insert the SD card into the controller. Result: The transfer procedure starts and the SD LED is flashing during this procedure.
4	Wait until the end of the download: • If the SD LED (green) is ON, the download ended successfully. • If the SD LED (green) is OFF, and the ERR and I/O LEDs (red) flash regularly, an error is detected.
5	Remove the SD card from the controller. NOTE: Changes are applied after next restart.

When the controller has executed the script, the result is logged on the SD card (file `/sys/cmd/script.log`).

Cloning a Controller

Introduction

The clone function allows you to upload the application from one controller and to download it only to a same controller reference.

This function clones every parameter of the controller (for example applications, firmware, data file, post configuration, remanent variables). Refer to [Memory Mapping](#), page 25.

Cloning the controller is possible by:

- Using an SD Card with a compatible script file
- Using the **FB_ControlClone**
- Using the **Controller Assistant**

When using an SD card, you can also copy the controller firmware and user access rights to the target controller.

NOTE: User access rights can only be copied with an SD Card if the **Include User Rights** button has previously been clicked on the **Maintenance > User Management > Clone Management** subpage of the [Web server](#), page 135.

Before Cloning a Controller

Safety Instructions

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.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Consult the controller state and behavior diagram in this document to understand the state that will be assumed by the controller after you cycle power.

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

Access Rights

By default, clone is allowed without using the function block **FB_ControlClone**. If you want to restrict access to the clone feature, you can remove the access rights of the `FrmUpdate` object on **ExternalMedia** group, page 71. As a result, cloning will be not allowed without using **FB_ControlClone**. For more details about this function block, refer to the Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide). For more details about Access Rights, refer to the EcoStruxure Machine Expert Programming Guide.

If you wish to control access to the cloned application in the target controller, you must use the **Include users rights** button (on the **Clone Management** subpage of the **Web server**, page 135) of the source controller before doing the clone operation. For more details about Access Rights, refer to the EcoStruxure Machine Expert Programming Guide.

SD Card Rules

The Modicon M262 Logic/Motion Controller accepts only SD cards formatted in FAT or FAT32.

When an SD card is inserted into the SD card slot of the controller, the firmware searches and executes the script contained in the SD card (`/sys/cmd/Script.cmd`).

The SD card must have a label. To add a label:

- 1. Insert the SD card in your PC.
- 2. Right-click on the drive in Windows Explorer.
- 3. Choose **Properties**.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have operational knowledge of your machine or process before connecting this device to your controller.
- Ensure that guards are in place so that any potential unintended equipment operation will not cause injury to personnel or damage to equipment.

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

Cloning a Controller

Cloning Procedure

Cloning the controller first removes the existing application from the target controller memory, if user access-rights are being copied to and enabled in the target controller. Refer to **Clone Management**, page 135Web server, page 135.

Step	Action
1	Erase an SD card and set the card label as follows: CLONExxx NOTE: The label must begin with ' CLONE ' (not case sensitive), followed by any normal character (a...z, A...Z, 0...9).
2	Select if you want to clone the Users Rights . Refer to the Clone Management subpage, page 135 of the Web server.

Step	Action
3	Remove power from the source controller.
4	Insert the prepared SD card in the source controller.
5	Restore power to the source controller. Result: The copy starts automatically. During the copy, the PWR and I/O LEDs are ON and the SD LED flashes regularly.
6	Wait until the copy is completed. Result: The SD LED is ON and the controller starts in normal application mode. If an error was detected, the ERR LED is ON and the controller is in STOPPED state.
7	Remove the SD card from the source controller.
8	Remove power from the target controller.
9	Insert the SD card into the target controller.
10	Restore power to the target controller. Result: The paste starts automatically and the SD LED is flashing during this procedure.
11	Wait until the end of the paste: • If the SD LED (green) is ON, the cloning ended successfully. • If the SD LED (green) is OFF, and the ERR and I/O LEDs (red) flash regularly, an error is detected. • If the SD LED (orange) is ON, the cloning is done with an error.
12	Remove the SD card to restart the target controller.

NOTE: When they are copied, access rights are only operational after a controller reboot.

Cloned and non cloned directories

For security reasons, not all directories of /usr files are cloned.

This table indicates the cloned and non cloned directories of /usr files:

Directory	Status
App	Cloned
Cfg	Cloned
Dta	Cloned
Fdr	Cloned
Log	Cloned
Other /usr directories	Cloned
pki	Not cloned
pki/own	Not cloned
Rcp	Cloned
Syslog	Not cloned
URF	Depends on the Clone management setting, page 135
Visu	Cloned
Web	Cloned

Compatibility

Software and Firmware Compatibilities

EcoStruxure Machine Expert Compatibility and Migration

Software and Firmware compatibilities are described in the EcoStruxure Machine Expert Compatibility and Migration User Guide.

Diagnostic

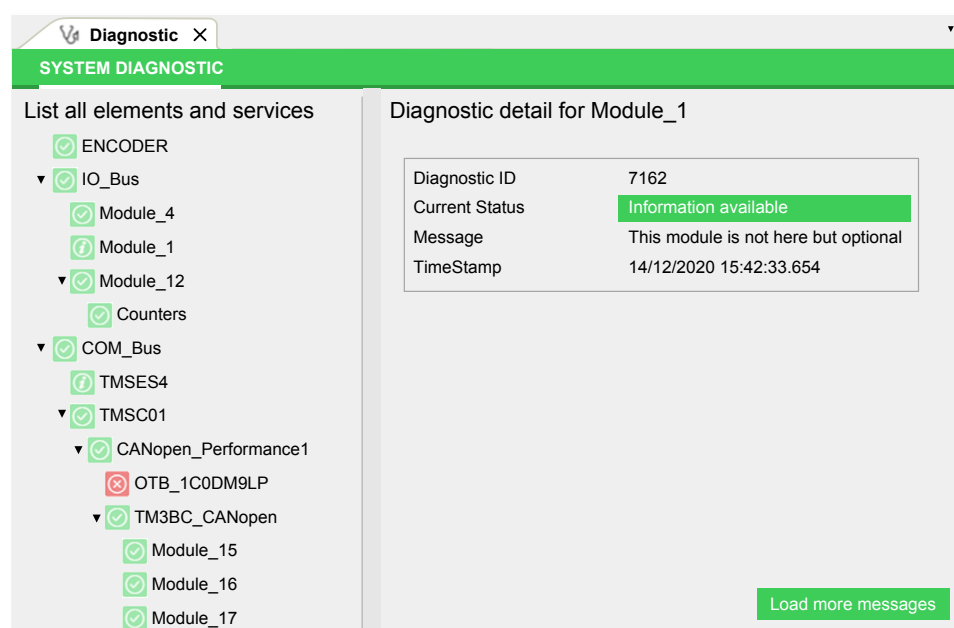
System Diagnostic

Presentation

The **Diagnostic** function displays diagnostic details as messages for the configured elements and services.

System Diagnostic View

To open the diagnostic view, double click **Diagnostic** in the **Devices tree**:



Diagnostic Messages

Diagnostic Messages Objects

Title	Description
Application and SD Card Diagnostic Messages	See M262 Application and SD Card Diagnostic Messages, page 242
Communication Diagnostic Messages	See M262 Communication Diagnostic Messages, page 243
OPC UA Functions Diagnostic Messages	See M262 OPC UA Functions Diagnostic Messages, page 246
M262 Hardware TM3 Expansions Diagnostic Messages	See M262 Hardware TM3 Expansions Diagnostic Messages, page 250
M262 Hardware TMS Expansions Diagnostic Messages	See M262 Hardware TMS Expansions Diagnostic Messages, page 252
M262 Hardware Expansions Diagnostic Messages	See M262 Hardware Expansions Diagnostic Messages, page 253
M262 Synchronized Motion Diagnostic Messages	See M262 Synchronized Motion Diagnostic Messages, page 254
M262 Motion Control Diagnostic Messages	See M262 Motion Control Diagnostic Messages, page 257
M262 Hardware IO Link Diagnostic Messages	See M262 Hardware IO Link Diagnostic Messages, page 259

Generic Diagnostic Messages

Diagnostic ID	Description	Criticality
1	Generic OK message	OK
2	Generic advisory message	Advisory
3	Generic error message	Error
4	Generic test message	Info

M262 Application and SD Card Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7235	Controller component is running.	-	-	OK
7236	System watchdogs exceeds its threshold condition. The threshold conditions for the three system watchdogs are defined as follows: If all of the tasks require more than 85% of the processor resources for more than 3 seconds, a system error is detected. The controller enters the HALT state. If the total execution time of the tasks with priorities between 0 and 24 reaches 100% of processor resources for more than 1 second, an application error is detected. The controller responds with an automatic reboot into the EMPTY state. If the lowest priority task of the system is not executed during an interval of 10 seconds, a system error is detected. The controller responds with an automatic reboot into the EMPTY state.	-	-	Advisory
7237	There is no application loaded in the controller.	-	-	Info
7238	The controller has a valid application that is stopped.	-	-	Info
7239	The controller is executing a valid application.	-	-	Info
7240	Application is in error, refer to EcoStruxure Machine Expert Programming Guide to get more details about the error.	-	-	Advisory
7241	-	-	-	OK
7242	Application has to be set in STOPPED state, verify the PLC_R_STOP_CAUSE data type value to get more details.	-	-	OK
7243	Boot project does not exist in non-volatile memory.	-	-	Info
7244	Boot project in non-volatile memory is the same as the project loaded in memory.	-	-	OK
7245	Boot project in non-volatile memory is different from the project loaded in memory.	-	-	Advisory
7246	Boot project is being created.	-	-	Info
7247	Error detected in the SD card. More details on the error that was detected are written to the file FwLog.txt.	-	-	Advisory
7248	No SD card detected in the slot or the slot is not connected.	-	-	Info
7249	SD card is in read-only mode.	-	-	Info
7250	SD card is in read/write mode.	-	-	Info

M262 Communication Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7106	The number of Ethernet interfaces to which the DHCP feature is applied is greater than authorized.	EcoStruxure Machine Expert project might be corrupted.	Re-create the project and re-compile.	Error
7107	The Ethernet interface cannot be found for this address.	EcoStruxure Machine Expert project might be corrupted.	Re-create the project and re-compile.	Error
7120	One IP address is set to two network interfaces in the same sub-network.	Incorrect IP address setting.	Verify the IP address setting for each network interface.	Error
7121	The bus coupler TM3 or TM5 is in valid communication state.	-	-	OK
7122	Bus coupler is in communication error.	Incorrect IP address configured or no connection between the controller and the bus coupler.	Verify the connection between controller and bus coupler. Verify the IP address of the bus coupler.	Error
7123	Bus coupler is incorrectly configured.	Configuration of the bus coupler in the EcoStruxure Machine Expert is incorrect.	Verify the configuration of the bus coupler, recompile and re-download the application.	Error
7124	The application was stopped.	-	-	Info
7126	Applied to generic Modbus devices, when the Modbus communication is stopped.	Modbus devices are in stop, no communication.	Run the application.	Info
7127	Applied to generic Modbus devices. The Modbus device is in operational state.	-	-	Info
7133	The module is OK.	-	-	OK
7134	The application is stopped.	-	-	Info
7135	Communication error is detected during the runtime process for slice modules. This module status is managed and sent to the controller by the bus coupler via Ethernet/IP or Modbus protocol.	Several possible causes related to the bus coupler.	Verify the bus coupler status.	Error
7136	The slice module configured in the EcoStruxure Machine Expert project is missing physically in the network.	No slice module connected to the bus couple.	Verify the slice module.	Error
7137	Indeterminable status reported by the I/O module.	-	-	Advisory
7138	Ethernet/IP or Modbus I/O Scanner detects communication errors with the bus coupler.	Cable disconnected or disturbances in the network.	Verify cable connection. Verify network configuration: IP address, netmask and gateway address.	Error
7139	Modbus communication is stopped.	-	Application stopped by user.	Advisory
7140	The bus coupler TM3 or TM5 is configured in the EcoStruxure Machine Expert project but missing physically.	No bus coupler is connected in the network.	Verify the EcoStruxure Machine Expert project or cable connection.	Advisory
7141	Bus coupler is in an error state during the runtime. All sub modules are automatically switched to this state. The application is in RUNNING state.	The bus coupler reports an error.	Verify the bus coupler status.	Advisory
7142	Undefined status reported by the bus coupler.	-	-	Advisory
7143	Ethernet/IP device error detected due to incorrect configuration.	Device is incorrectly configured.	Verify the device configuration.	Error
7144	Module is incorrectly configured. This runtime status is managed and sent to the controller by the bus coupler via Ethernet/IP or Modbus protocol.	Configuration of the slice in the EcoStruxure Machine Expert is incorrect.	Verify whether the module in the project is the same as the one present physically.	Error
7701	No memory available for memory allocation.	Memory allocation issue or EcoStruxure Machine Expert project corrupted.	Reboot the controller or rebuild the project.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7100	The Ethernet network interface is running without errors.	-	-	OK
7101	The Ethernet network interface is not running. ETH 1 and ETH 2 correspond to the Ethernet objects as seen in EcoStruxure Machine Expert. From ETH 3-7 are part of the TMS module which can be added to the controller.	Wiring issue or incorrect IP address setting.	Verify the wiring and verify the network configuration in EcoStruxure Machine Expert.	Error
7102	The Ethernet network interface IP address duplicated in the network. ETH 1 and ETH 2 correspond to the Ethernet objects as seen in EcoStruxure Machine Expert. From ETH 3-7 are part of the TMS module which can be added to the controller.	IP address duplicated.	Verify whether the IP address is unique in the network.	Error
7103	The Ethernet device is waiting for an IP address from server.	-	-	Info
7104	The Ethernet device is waiting to be configured according to the IP address received from server.	-	-	Info
7105	IP address conflict is detected between two Ethernet interfaces.	Incorrect IP address setting.	Verify IP address configuration for Ethernet interfaces in the EcoStruxure Machine Expert project.	Error
7110	When the Ethernet Interface e.g. ETH 1 is in an error state all the sub-modules are set to an error state. No Ethernet communication is possible.	The Ethernet interface is in an error state.	Verify the state of the Ethernet interface	Error
7111	The interface is not allowed to be configured with the specified IP address if the Ethernet scanner or Sercos is set on this interface.	Sercos or Ethernet/IP is configured for this interface.	Verify the setting of the Ethernet interface.	Error
7112	Controller is trying to configure a new gateway different from others inside the same sub-network.	More than one gateway setting detected in the sub-network.	Verify the gateway setting.	Error
7113	No such interface is found.	No such interface is in the EcoStruxure Machine Expert project.	Verify the interface setting.	Error
7114	IP address conflict is detected. If the conflict source is NOT from net manager or the task SetIpTask, interface uses the default setting.	Trying to use identical IP address for multiple interfaces.	Verify the IP address setting for each interface.	Error
7115	IP address conflict is detected. If the conflict source is from net manager or the task setIpTask, the setting is stopped and the interface setting keeps previous one.	Trying to use identical IP address for multiple interfaces.	Verify the IP address setting for each interface.	Error
7116	Memory allocation error due to insufficient memory.	EcoStruxure Machine Expert project corrupted	Reboot controller or rebuild the EcoStruxure Machine Expert project.	Error
7117	Network saturation is detected on Ethernet interface.	The saturation could be caused by a network misconfiguration or by an external agent.	Verify your network and security settings.	Error
7118	Network saturation ended.	-	-	Info
7119	The USB Ethernet interface is not working.	USB driver issue.	Verify PC controller-USB driver status on your Operating System. Verify the USB network mask setting.	Error
6100	The USB Ethernet interface setting is incorrect, use the default mask setting (255.255.255.0) instead.	Ethernet network issue: set network mask was unsuccessful.	Verify the USB Ethernet mask setting.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
6120	The CAN bus communication cannot be established due to an incorrect initialization .	• No TMSCO1/CAN cable connected • Incorrect baudrate • Physical CAN network configuration issue, such as incorrect terminal resistors, incorrect node ID	Connect the TMSCO1 bus, verify the baudrate, the physical connection and the node id.	Error
7800	Modem configuration error or device incommunicative.	The modem might be absent or is not correctly configured.	Verify the wiring or verify the configuration inside the EcoStruxure Machine Expert project.	Error

M262 OPC UA Functions Diagnostic Messages

Diagnostic ID	Description	Causes	Possible solutions	Criticality
7905	The system could not allocate memory for this specific element of the Symbol Configuration.	Not enough runtime memory.	Reduce the amount of symbols in the symbol configuration. Try to clean the memory with a reboot.	Error
7903	Name of the symbol in symbol configuration has more than 255 characters.	Symbol name string too long.	Reduce the name of the symbol exposed in the symbol configuration.	Error
7906	The maximum number of symbols to be allocated has been reached. Further symbols will be ignored.	Too many symbols.	Reduce the amount of symbols in the symbol configuration.	Info
7260	Finished creating the OPC UA Server AddressSpace.	-	-	OK
7262	Could not get OPC UA configuration from the application.	Server parameters missing or corrupted.	Make sure OPC UA server configuration is correct. Clear the application from the controller, recompile the application and download the application into the controller again.	Error
7263	Configuration of the OPC UA server in the application is OK.	-	-	OK
7269	Could not allocate memory for a value of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the array.	Error
7270	Could not allocate memory for a subscription sample value of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the sample queue of your OPC UA subscription.	Error
7271	Could not allocate memory for a value of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the array.	Error
7272	Could not allocate memory for a value of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the string.	Error
7273	Could not get the symbol size from the Symbol Configuration.	Internal error during calculation of the size of the symbol.	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7274	Could not allocate memory for a value of an OPC UA node.	Not enough runtime memory.	Try to reduce the number of symbols in the symbol configuration.	Error
7275	Could not instantiate an OPC UA node of this datatype, the datatype is not supported by the OPC UA Server.	Unsupported Symbol DataType.	Change the DataType of the Symbol.	Error
7276	Could not get the symbol corresponding to this OPC UA node.	Interface error.	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7277	Could not get the symbol corresponding to this OPC UA node.	Interface error.	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error

Diagnostic ID	Description	Causes	Possible solutions	Criticality
7278	Not enough memory to create symbol list.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the amount of symbols in the symbol configuration.	Error
7279	Could not allocate memory for an array of values of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the array.	Error
7280	Could not get the type description of an array.	Interface error.	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7281	The type of symbol is not supported by the OPCU Server.	Unsupported Symbol DataType.	Change the DataType of the Symbol.	Error
7282	The size of the symbol with DataType Wstring is over the limit.	Size of Symbol too big.	Reduce the size of the WSTRING symbol to 126 words or less.	Error
7283	Not enough memory to create symbol list.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the amount of symbols in the symbol configuration.	Error
7284	Could not create variable in address space.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the amount of symbols in the symbol configuration.	Error
7285	The type of array symbol is not supported by the OPC UA Server.	Unsupported Symbol DataType.	Change the DataType of the Array Symbol.	Error
7286	Could not allocate memory for a value of an OPC UA node.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the size of the array.	Error
7287	Could not get the value of an OPC UA node.	Interface error.	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7288	Could not allocate a new symbol.	Not enough runtime memory.	Try to clean the controller memory. Alternatively try to reduce the amount of symbols in the symbol configuration.	Error
7289	Could not create the OPC UA server address space.	-	Make sure symbol configuration is correct. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7290	Could not get the OPC UA server configuration from the application.	Interface error.	Make sure OPC UA server configuration is correct. Clear the application from the controller, recompile the application and download the application into the controller again.	Error
7291	Could not allocate memory for the configured OPC UA server endpoints.	Not enough runtime memory.	Try to change the server endpoint in the configuration. Try to adjust the security policy and/or message security.	Error

Diagnostic ID	Description	Causes	Possible solutions	Criticality
7292	Could not initialize the OPC UA stack with the given configuration.	Configuration error.	Make sure OPC UA server configuration is correct. Clear the application from the controller, recompile the application and download the application into the controller again.	Error
7293	Could not create the datatypes table of the OPC UA server.	Configuration error.	Make sure symbol configuration is correct and that exposed datatypes are supported. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7294	Could not add a datatype to the table of datatypes of the OPC UA server.	Configuration error. Not enough runtime memory.	Make sure symbol configuration is correct and that exposed datatypes are supported. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7296	The OPC UA server could not create the folder for the untrusted certificates.	FileSystem error. Not enough filesystem memory.	Clear some space on the controller physical memory.	Advisory
7297	The OPC UA server could not create the folder for the trusted certificates.	FileSystem error. Not enough filesystem memory.	Clear some space on the controller physical memory.	Advisory
7298	The OPC UA server could not create the folder for the revoked certificates list (CRL).	FileSystem error. Not enough filesystem memory.	Clear some space on the controller physical memory.	Advisory
7299	The OPC UA server could not create the folder for the Issuers certificates (other certificates in the certification path).	FileSystem error. Not enough filesystem memory.	Clear some space on the controller physical memory.	Advisory
7900	The OPC UA server could not create the folder for the Issuers certificate revoke list (CRL for other certificates in the certification path).	FileSystem error. Not enough filesystem memory.	Clear some space on the controller physical memory.	Advisory
7901	Could not add the user defined datatype to the table of datatypes of the OPC UA server.	Configuration error.	Make sure symbol configuration is correct and that exposed datatypes are supported. Clear the application from the controller, recompile the application and symbol configuration, then download the application into the controller again.	Error
7902	Finished adding user defined datatypes to the OPC UA server.	-	-	Info
7907	The user has connected to the OPC UA server.	-	-	Info
7908	The session of the user has expired and has timed out. The user has been disconnected.	-	-	Info
7909	You have disconnected from the OPC UA server manually.	-	-	Info
7910	A connection attempt has been made with an incorrect user name or password.	OPC UA Session Authentication.	Verify your authentication credentials in your OPC UA client, and reconnect.	Error
7911	The type of token used for connection authentication is invalid.	OPC UA Session Authentication.	Verify your authentication token in your OPC UA client, make sure it is supported by the server, and reconnect.	Error

Diagnostic ID	Description	Causes	Possible solutions	Criticality
7912	The server has reached a maximum of clients connected simultaneously.	OPC UA Server configuration.	Try to disconnect an unused client, and reconnect with the current one. Try to increase the maximum amount of client connections on the OPC UA server configuration.	Error
7913	The pointed client certificate has expired and is no longer valid for new OPC UA sessions.	-	Try to regenerate the client certificate with a new validity date.	Info
7914	The pointed client certificate is invalid.	-	Ensure that the client certificate respects the OPC UA defined extensions (such as the Alternative Subject OPC UA fields).	Info
7915	The pointed client certificate has been added into the untrusted folder.	-	This usually happens at the client first connection attempt. To accept a connection from this client either trust the certificate or move the certificate to the trusted folder.	Info
7916	The OPC UA server's certificate has been created.	-	-	Info
7917	The pointed client certificate has been trusted (added into the trusted folder).	-	You can now connect to the OPC UA server using this client.	Info
7918	The pointed client certificate has been added into the untrusted folder.	-	This usually happens at the client first connection attempt. To accept a connection from this client either trust the certificate or move the certificate to the trusted folder.	Info
7919	The client certificate did not pass the minimum required verifications.	OPC UA Client Certificate.	Make sure your client certificate is correct. Try to regenerate the client certificate.	Error
7920	The client has disconnected due to a low level transport protocol disconnection.	OPC UA Connection.	Try to reinitialize the client and reconnect. Try to restart the server and reconnect.	Error
7921	A successful connection to the OPC UA server was made from the indicated IP address.	-	-	Info
7922	At least three unsuccessful connections to the OPC UA server were made from the indicated IP address.	OPC UA Session Authentication.	Verify your authentication credentials in your OPC UA client, and reconnect (if it is a known client).	Error
7923	The user has disconnected from the OPC UA server manually (from the given IP address).	-	-	Info

M262 Hardware TM3 Expansions Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7160	Configuration of I/O expansion bus TM3 done without errors.	-	-	OK
7161	Error configuring the TM3 I/O expansion bus.	The EcoStruxure Machine Expert project is misconfigured.	Verify the EcoStruxure Machine Expert project if the configured TM3 slices match with the existing ones.	Error
7162	The optional module is not mandatory. If it is not present, there is no impact on other modules.	The module is missing. As it is an optional module this might be expected.	Verify if the module is missing and if this is intended.	Info
7163	Error detected during the configuration of the module.	The module is missing. The module reference does not correspond to the configured one in the project. The module has the correct reference but the firmware version might be different.	Verify whether the module is connected and that the reference and firmware version are correct.	Error
7164	The reinitialization of the TM3 I/O expansion bus was successful.	-	-	OK
7165	The reinitialization of the TM3 I/O expansion bus was unsuccessful.	The EcoStruxure Machine Expert project is misconfigured.	Verify the EcoStruxure Machine Expert project if the configured TM3 slices match with the existing ones	Error
7166	Indeterminable error during scanning of the internal bus. Identification of the modules was unsuccessful.	A module might be in error.	Verify whether all modules are correctly connected and powered.	Error
7167	Indeterminable error during scanning of the internal bus. Identification of the modules was unsuccessful.	A module might be in error.	Verify whether all modules are correctly connected and powered.	Error
7168	No module found on the bus.	A module might be in error.	Verify whether all modules are correctly connected and powered.	Error
7169	Preparing the update of the module.	-	-	Info
7170	The given firmware file is invalid.	Firmware file is invalid	Verify firmware file and/or download again from Schneider Electric website.	Error
7171	The given firmware file cannot be processed by the controller.	Firmware format is not supported by the controller.	Update the controller firmware version.	Error
7172	Internal bus error during firmware update.	Timeout on the internal bus.	Retry the process.	Error
7173	Internal system error during firmware update.	-	Retry the process.	Error
7174	Module will be updated with the new firmware.	-	-	Info
7175	Module will not be updated. Firmware of the module is already up to date.	-	-	Info
7176	The TM3 firmware for the designated module was updated.	-	-	OK
7177	Internal bus error during firmware update.	A module might be in error.	Retry the process.	Error
7178	Internal system error during firmware update.	A module might be in error.	Retry the process.	Error
7179	Firmware file cannot be processed by the controller.	Firmware format is not recognized by the controller.	Verify that the file is a firmware file. Update the controller firmware version, if necessary.	Error
7180	Module firmware cannot be updated.	Some old I/O modules do not support the firmware update.	Replace the module with a module hardware version that supports firmware updates.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7181	Internal error during firmware update.	Indeterminable system error during the process of the firmware update.	Retry the process.	Error
7183	The firmware update process was completed without errors.	-	-	OK
7184	I/O bus is in an error state during the runtime. All sub modules are automatically switched to this state. The application is in RUNNING state.	A module might be in error.	-	Advisory
7185	More modules than expected found in the I/O bus.	More modules connected than configured.	Remove extra modules from the bus.	Error

M262 Hardware TMS Expansions Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
6315	Health option requested to the module but no response.	The smart communication module has an internal error or one of its interface is overloaded.	Verify that the network is configured properly on any interface of the given TMS and make sure that it is connected correctly	Error
7210	The configured module is not matching the module found at the current position.	A module might be in error. The configured module is incompatible with the firmware version of the module.	Update the firmware of the module.	Error
7211	The module inserted at this specific position uses an unsupported firmware version.	The firmware version of the module is not supported by the controller firmware version.	Update the controller firmware version. Update the module firmware version.	Error
7212	The configured module is not supported by the controller firmware version.	The configured version of the module is not supported by the controller firmware version.	Update the controller firmware.	Error
7213	The configured module is not matching the module found at the current position.	A module might be in error. The configured module is incompatible with the firmware version of the module.	Update the firmware of the module.	Error
7221	An internal MAC address has been found twice in the configuration.	There is an error with the MAC address of the physical module.	The module in error must be replaced.	Error
7222	An internal MAC address has been found twice in the configuration.	There is an error with the MAC address of the physical module.	The module in error must be replaced.	Error
7223	More than 7 modules found on the bus or in the configuration. This is not supported by the system.	Misconfiguration in the application. There are more than 7 modules inserted on the controller.	Remove the extra module (s) from the configuration or remove them.	Error
7224	No power supply to the module detected.	The wiring of the module might be incorrect. The module might be in error.	Verify the wiring and power supply or change the module.	Error
7225	A module has been configured at this specific place but no module found.	A module is missing.	Change your configuration or insert the missing module on the bus.	Error
7226	A module has been configured at this specific place but no module found.	A module is missing.	Change your configuration or insert the missing module on the bus.	Error
7228	A module has been found on the bus but it is not configured.	An extra module has been found on the bus.	Change your configuration to add it or disconnect the module from the bus.	Error
7229	Internal communication error on the TMS bus.	A module might be in error.	Reboot the controller.	Error
7230	A module has been removed or added on the bus.	Hot swap of modules not supported by the TMS bus	Reboot the controller.	Error
7231	Internal communication error on the TMS while the bus was already stopped.	A module might be in error. Hot swap of modules not supported by the TMS bus.	Reboot the controller.	Error
7232	No module found on the bus.	A module might be in error.	Reboot the controller. Replace the module.	Error
7233	The firmware update is unsuccessful for the given module.	Possible error with the firmware file.	Verify whether the file is the correct one.	Error
7234	Module firmware is not transferred.	Could be that the communication with the TMS was interrupted or trying to transfer an incorrect firmware file.	Verify whether the TMS is correctly connected and that the file is the correct one.	Error
6310	TMS Ethernet discovery error detected.	-	-	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
6311	TMS module configuration error.	-	-	Error
6312	Error assigning the IP address to the TMS.	-	-	Error
6313	VLAN configuration error.	-	-	Error
6314	The Ethernet interface of the TMS is inoperative.	-	-	Error

M262 Hardware Expansions Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7510	Initialization error detected for I/O devices.	The configuration in the project might be incorrect.	Verify the EcoStruxure Machine Expert project.	Error
7511	Configuration error detected for I/O devices.	The configuration in the project might be incorrect.	Verify the EcoStruxure Machine Expert project.	Error
7512	Shortcut error detected for I/O devices.	The cabling of the I/O might have errors.	Verify the electrical wiring.	Error
7513	No power supply detected for I/O devices.	The power supply might be disconnected or the cabling might have errors.	Verify the electrical wiring.	Error
7610	No power supply detected for the encoder.	The power supply might be disconnected or the cabling might be incorrect.	Verify the electrical wiring.	Error
7611	No communication with the encoder.	Communication with the encoder is unstable or inoperative.	Verify the wiring.	Error

M262 Synchronized Motion Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7300	Sercos Phase NRT successfully activated.	-	-	Info
7301	Sercos Phase 0 successfully activated.	-	-	Info
7302	Sercos Phase 1 successfully activated.	-	-	Info
7303	Sercos Phase 2 successfully activated.	-	-	Info
7304	Sercos Phase 3 successfully activated.	-	-	Info
7305	Sercos Phase 4 successfully activated.	-	-	Info
7306	Sercos Simulated Phase 2 successfully activated.	-	-	Info
7307	Sercos Simulated Phase 3 successfully activated.	-	-	Info
7308	Sercos Simulated Phase 4 successfully activated.	-	-	Info
7309	Shows the number of physically connected Sercos devices independently of the device type (I/O or drive).	-	-	Info
7310	Transition to Sercos Phase 0 unsuccessful.	No Sercos devices connected e.g. cable damaged or absent.	-	Error
7311	Transition to Sercos Phase 1 unsuccessful.	-	-	Error
7312	Transition to Sercos Phase 2 unsuccessful.	-	-	Error
7313	Transition to Sercos Phase 3 unsuccessful.	-	-	Error
7314	Transition to Sercos Phase 4 unsuccessful.	-	-	Error
7315	AxisRef was not stopped properly before stopping the controller application.	One axis was running while the controller application has been stopped. An automatic Errorstop has been triggered on this axis.	Ensure that all axes have been stopped properly (e.g.: using MC_Stop) before stopping the controller application.	Error
7316	The Sercos master is controlling if the drives are sending a correct connection control in each Sercos cycle. If the master detects an incorrect connection control (usually the NewData-Bit is not toggled correctly). The master creates a Log Message about this. One incorrect connection control is allowed. If a second one follows in the next Sercos cycle the axis connected to the drive will be put to ErrorStop .	Connection issue with the Sercos slave.	Verify the wiring for this specific slave.	Advisory
7317	Removing existing Network Address Translation (NAT) rule unsuccessful.	vxWorks rejected NAT rule removal.	Reboot the controller.	Error
7318	Setting NAT rule TCP unsuccessful.	xWorks rejected NAT rule creation for TCP.	Reboot the controller.	Error
7319	Setting NAT rule UDP unsuccessful.	xWorks rejected NAT rule creation for UDP.	Reboot the controller.	Error
7320	The Sercos master netmask, set under Ethernet 1, must be set to 255.255.255.0.	"Sercos master" netmask, set under Ethernet 1, is not set to 255.255.255.0.	The "Sercos master" netmask, set under Ethernet 1, must be set to 255.255.255.0.	Advisory
7321	Two logical devices tried to connect to one physical slave: conflict with Sercos address.	Multiple devices have been configured to the same Sercos address in your application.	Ensure that each device has a unique Sercos address configured in your application.	Error
7322	Two logical devices tried to connect to one physical slave: conflict with topological address.	Multiple devices have been configured to the same Topology address in your application.	Ensure that each device has a unique Topology address configured in your application.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7323	Two logical devices tried to connect to one physical slave: conflict between one topological address and a Sercos address.	Multiple devices have been configured to the same Sercos and Topology address in your application.	Ensure that each device has a unique Sercos and Topology address configured in your application.	Error
7324	Sercos slave at a given topological address reports error, Sercos phase change to Phase 2 with duplicate Sercos address.	Multiple devices have the same Sercos address configured in their communication settings.	Ensure that each device has a unique Sercos address configured in their communication settings.	Error
7325	In case that the Sercos address of a device is internally set to 0, or if duplicated Sercos address have been detected. And automatic reassignment of the Sercos address is done by the master.	Settings in the device are blocking the master from reassigning its Sercos address (e.g. a hardware switch is defining the Sercos address).	Reassign manually the device to a free Sercos address.	Info
7326	In case that the Sercos address of a device is internally set to 0, or if duplicated Sercos address have been detected. And automatic reassignment of the Sercos address is done by the master.	Settings in the device are blocking the master from reassigning its Sercos address (e.g. a hardware switch is defining the Sercos address).	Reassign manually the device to a free Sercos address.	Error
7327	IP address already used by Sercos master and configured for another device.	-	Change IP address of Sercos master or Sercos device.	Advisory
7328	IP address already used by another device.	-	Reconfigure IP address of Sercos device.	Advisory
7329	Phase up to phase X is not possible as a CoSeMa error Y is detected.	-	Contact the support.	Error
7330	Phase up to phase X is not possible as a CoSeMa error Y is detected.	-	Contact the support.	Error
7331	Phase up to phase X is not possible as a CoSeMa error Y is detected.	-	Contact the support.	Error
7332	In case that the Sercos address of a device is internally set to 0, or if duplicated Sercos address have been detected. And automatic reassignment of the Sercos address is done by the master.	-	-	Info
7333	In case that the Sercos address of a device is internally set to 0, or if duplicated Sercos address have been detected. And automatic reassignment of the Sercos address is done by the master. Leading to a new phase up initiated by the master.	Duplicated Sercos address identified on the network.	Change the Sercos address assignment in the project Sercos configuration.	Info
7334	No communication with Sercos slave at the given Topological Address.	-	Verify whether the Sercos slave is connected to the master and still operating correctly.	Error
7335	No data received from Sercos slave at the given Topological Address.	The connection control IDN from the slave stopped toggling.	Verify whether the Sercos slave is connected to the master and still operating correctly.	Error
7336	Class 1 error from Sercos slave at Topological Address.	An error is detected on the Sercos slave.	Trigger procedure command S-0-099.	Error
7337	Class 2 error from Sercos slave at Topological Address.	An advisory is detected on the Sercos slave.	-	Advisory
7338	Class 1 error from Sercos slave at Topological Address.	An error is detected on the Sercos slave.	Read IDN S-0-0390.0.0.	Error
7339	Class 2 error from Sercos slave at Topological Address.	An advisory is detected on the Sercos slave.	Read IDN S-0-0390.0.0.	Advisory
7340	Displays the number of devices configured in the application, and how many have been scanned on Sercos by the master.	-	-	Info
7341	Device limit exceeded for CycleTime.	Too many devices have been configured in your application for the cycle time configured.	Increase the cycle time configured or reduce the number of devices configured.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7342	Sercos cable disconnected from master.	The Sercos cable has been disconnected from the master.	Ensure that the Sercos cable is connected to the master.	Error
7343	Stack error raised by Sercos master for multiple consecutive cycles.	Sercos slave disconnected from fieldbus or unresponsive.	Ensure that all Sercos slaves are wired correctly and operational.	Error
7344	Stack error raised by Sercos master for multiple consecutive cycles.	Sercos slave did not respond.	Ensure that all Sercos slaves are wired correctly and operational.	Error
7345	RTMP time exceeds the Sercos cycle.	The load of the Real Time Motion.	Reduce the load in your application.	Error
7346	The message 'RTMP time exceeds the Sercos cycle ...' is not displayed further until next Sercos phase up to avoid too many redundant messages.	Process (Motion+Sercos task) have exceeded the maximum allowed load.	Reduce the load in your application.	Error
7347	Displays the Sercos Cycle Time configured in your application in ns.	-	-	Info
7348	There is a logical device configured with Sercos address X, that cannot get mapped to a physical device on Sercos line.	Connection issue with the Sercos slave or an incorrect Sercos address is configured.	Ensure that the configured device is connected to the master, and operational.	Error
7349	There is a logical device configured with topological address X, that cannot get mapped to a physical device on Sercos line.	Connection issue with the Sercos slave or an incorrect topological address is configured.	Ensure that the configured device is connected to the master, and operational.	Error
7350	Phase up to phase X is not possible as a CoSeMa error Y is detected.	No connection to Sercos devices.	Ensure that all Sercos slaves are wired correctly and operational.	Error
7351	Phase up to phase X is not possible as a CoSeMa error Y is detected.	-	-	Error
7352	Phase up to phase X is not possible as a CoSeMa error Y is detected.	Incorrect timing configuration or incorrect process data configuration or incorrect IP address configuration or incorrect device assignment.	Verify your device configuration and application device mapping in the EcoStruxure Machine Expert project.	Error
7353	Phase up to phase X is not possible as a CoSeMa error Y is detected.	Maximum limits of real axis reached or duplicated Sercos address.	Reduce the number of physical axes and verify in your project the uniqueness of the Sercos address.	Error
7354	Phase up to phase X is not possible as a CoSeMa error Y is detected.	-	-	Error
7355	Writing data on the given IDN is unsuccessful.	One IDN configuration is unsuccessful.	Re-initiate a new Phase up and ensure that the device is operational.	Error
7356	LXM32S CoplaCommunication module software revision is inferior to that required for correct operation.	-	Update the firmware of the LXM32S Copla module.	Error
7357	LXM32S Software Revision is inferior to that required for correct operation.	-	Update the firmware of the LXM32S.	Error
7358	An undeterminable external exception occurred and terminated either the Lxm32s-Homing or the SercosStateMachine Task.	Indeterminable firmware-generated response.	Reboot the controller.	Error
7359	An undeterminable external exception occurred and terminated the motion task.	Indeterminable firmware-generated response.	Reboot the controller.	Error
7360	An undeterminable internal exception occurred and terminated either the Lxm32s-Homing or the SercosStateMachine Task.	Indeterminable firmware-generated response.	Reboot the controller.	Error
7361	An undeterminable internal exception occurred and terminated the motion task.	Indeterminable firmware-generated response.	Reboot the controller.	Error

M262 Motion Control Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7400	The axis is blocked by a different function block which cannot be interrupted.	MC_Stop.Execute = TRUE and another motion function block is executed or MC_Home is busy and another motion function block is executed.	Set axis to Standstill.	Error
7401	The power stage must be enabled before the function block can be executed.	MC_Power.Enable = FALSE when executing a motion function block.	Call MCPower with Enable = TRUE.	Error
7402	The function block cannot be repeated as long as the output Busy = TRUE.	A function block is executed again while a previous execution is ongoing.	Ensure that the function block is not busy.	Error
7403	The specified parameter address is not supported by the device.	The parameter address assigned to the MC_ReadParameter or MC_WriteParameter is not supported by the device.	Verify whether the parameter address is correct. Verify whether the parameter you want to access is supported by the device.	Error
7404	The number entered for the signal input is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7405	The number specified for the signal output is outside of the permissible value range.	Dedicated output is outside the valid value range.	Verify that the value of the output is inside the valid value range.	Error
7406	The command is not executed, the device is not ready.	The library is configuring the drive and a function block is executed.	Call the function block MC_ReadAxisInfo and verify that the output ReadyForPowerOn is TRUE.	Error
7407	Communication error detected. The connection to the device has been interrupted.	Incorrect fieldbus settings (address,...) or damaged/incorrect cable.	Verify fieldbus settings (device configuration). Verify wiring (hardware).	Error
7408	The command is not executed within the permissible time delay.	The execution time of the function block exceeds the specified timeout.	Increase the value for the dedicated timeout property.	Error
7409	Value out of range. The value is outside the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7410	Buffer full. Internal error detected.	The buffer of the internal FIFO for acyclic data exchange reaches the limit.	Reduce parallel execution of read and write function blocks. Contact your Schneider Electric service representative.	Error
7411	Parameter not supported by device.	The value of the input ParameterNumber assigned to the MC_ReadParameter or MC_WriteParameter is not supported by the device.	Verify whether the value of the input ParameterNumber is correct.	Error
7412	Touch probe number invalid. The specified value for the number of the Touchprobe input is invalid.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7413	The specified edge of the Touchprobe input is invalid.	Input TriggerEdge is outside the valid value range or the selected trigger edge is not supported by the drive.	Verify that the value of the input TriggerEdge is inside the valid value range. Verify whether the selected trigger edge is supported by the drive.	Error
7414	Touch probe inactive. An attempt has been made to cancel an inactive Touchprobe.	MC_AbortTrigger is executed for a touch probe which is not active.	Execute MC_AbortTrigger only for active touch probes.	Error
7415	Touch probe active. An attempt has been made to execute an active Touchprobe.	MC_TouchProbe is executed for an already active touch probe.	Execute MC_TouchProbe only for inactive touch probes.	Error

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7416	The detected error cannot be reset with MC_Reset.	Execution of the function block MC_Reset does not reset the drive error (e.g. STO drive error).	Verify drive state. Restart the device after having remedied the cause of the detected error.	Error
7417	Acceleration out of range. The value for the acceleration is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7418	Deceleration out of range. The value for the deceleration is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7419	Position out of range. The value for the target position is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7420	Velocity out of range. The value for the target velocity is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7421	Torque out of range. The value for the target torque is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7422	Numerator out of range. The value for the numerator is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7423	Denominator out of range. The value for the denominator is outside of the permissible value range.	Dedicated input is outside the valid value range.	Verify that the value of the input is inside the valid value range.	Error
7424	Halt active. The Halt function is active and the command is not executed.	The external halt function of the drive is active.	Verify that the external halt function is not active.	Error
7425	Function block Control_ATV active. The function block cannot be executed as long as the function block Control_ATV is enabled.	A motion function block is executed while the ATV drive is commanded by the Control_ATV function block.	Verify that the Control_ATV function block is not commanding the ATV.	Error
7426	Not ready for applying power. The power stage cannot be enabled in the operating state of the drive.	The drive is not able to apply power (e.g. no main supply).	Verify the drive status.	Error
7427	Incorrect drive type. The function block does not support the linked Axis_Ref type.	The executed function block does not support the drive (e.g. MoveVelocity_LXM32 is executed with ATV axis).	Verify that the executed function block is supported by the drive.	Error
7428	Setpoint source invalid. Invalid value at the input SetpointSource of the function block TorqueControl_LXM32 or MoveVelocity_LXM32.	The value of the input SetpointSource is outside the valid value range. (Only for the function blocks MoveVelocity_LXM32 and MoveVelocity_SD328A).	Verify that the value of the input SetpointSource is supported by the drive.	Error
7429	The selected homing method is not supported.	The input HomingMode is not supported by the drive.	Verify that the value of the input HomingMode is supported by the drive.	Error
7430	The digital output is set to an incorrect signal output function. Set the signal output function to Freely Available.	Execution of the function block MC_WriteDigitalOutput for ILX drive and the output is not configured as Freely Available.	Verify that the function of the drive output is Freely Available.	Error
7431	The operating mode is not supported.	The drive does not support the operation mode requested by the executed function block or ATV does not support operation mode profile position or homing.	Verify that the executed function block is supported by the drive.	Error

M262 Hardware IO Link Diagnostic Messages

Diagnostic ID	Description	Causes	Possible Solutions	Criticality
7960	Module is in INACTIVE state.	Configuration choice.	Modify your configuration and download again.	Advisory
7961	Module is in SIO_OUT mode.	Configuration choice.	-	OK
7962	Module is in SIO_IN mode.	Configuration choice.	-	OK
7963	Module is in PREOPERATIONAL state.	Module is in PREOPERATIONAL state.	-	OK
7964	Module is in OPERATIONAL state.	User configuration and startup of the IO-link device.	-	OK
7965	Parameter server data is OK.	User configuration and startup of the IO-link device with the parameter server enabled.	-	OK
7966	Parameter server: Upload ongoing.	Consequence of a user request.	-	OK
7967	Parameter server: Download ongoing.	Consequence of a user request.	-	OK
7968	Parameter server: Undeterminable error.	- Parameter server not supported - Error accessing an object that is managed by the Parameter - Internal Error	Verify the compliance of the IO-link device with the Parameter server needs.	Advisory
7969	Parameter server is locked.	Consequence of a user request.	-	Info
7970	Parameter server is empty.	Parameter server manipulation before filling it with data.	Ensure to download data beforehand.	Info
7971	Parameter server: New serial number recognized.	New device of the same kind of the previous one connected.	Verify that the IO-link device corresponds to the IODD file imported.	Info
7972	Invalid process data.	Invalid process data definition.	Verify the cable and replace if the necessary.	Advisory
7973	No communication.	Incorrect wiring or/and issue in the IO-link device.	Verify that the IO-link device corresponds to the IODD file imported	Error
7974	Device/Vendor ID mismatch.	Incorrect device definition.	Replace with a new IO-link device	Error
7975	Startup error detected.	Issue in the IO-link device during the startup phase.	Verify the hardware configuration relative to the software configuration and the wiring	Error
7976	The IO-link communication module has an issue.	Configuration or wiring error.		Advisory

Machine Assistant

Introduction

The Industrial Plug and Work technology supports Machine Assistant. It facilitates the machine configuration through Ethernet network.

Accessing the Web Server Through Industrial Plug and Work

Launching the Web Server

How to Launch the Web Server

This table describes how to launch the Web server:

Step	Action
1	Connect the controller to the PC using an RJ45 cable and open your computer network explorer. Result: The controller appears in your computer network explorer.
2	Double-click the controller to access the Web server authentication page.
3	Log in to access the home page of the Web server site., page 122

Using the Machine Assistant

Launching the Machine Assistant

Overview

Machine Assistant is displayed similarly in EcoStruxure Machine Expert and on the controller Web Server. Using this tab, you can monitor the controller and its connected devices.

Launching the Machine Assistant in the Web Server

Launch the Web server, page 260 and log in to access the home page of the Web server site, page 122. Click the **Machine Assistant** tab. The **Machine Assistant** window is displayed.

Launching the Machine Assistant in EcoStruxure Machine Expert

Step	Action
1	Create a project with a M262 Logic/Motion Controller.
2	Double-click the Machine Assistant node in the Devices tree . Result: The Machine Assistant window is displayed.

Managing the Network Scan

Overview

The network scan allows you to detect your controller and all the slave devices connected.

NOTE: EtherNet/IP devices are detected if they are located in the same subnetwork as does the controller.

Scanning the Network in the Web Server

Click the **scan** button.

Result: The scan is launched and run continuously. All the devices connected to the network are detected.

The scan is stopped when you click **Stop Scan** or close the **Machine Assistant**.

NOTE: The buttons are displayed in the control menu after the run scan has detected devices. Depending on your device, different buttons are displayed.

Scanning the Network in EcoStruxure Machine Expert

Connect to the controller and click **Launch scan**.

Result: The scan is launched and run continuously. All the devices connected to the network are detected.

The scan is automatically stopped when you close the **Machine Assistant**.

Scan status

You must add devices to the project.

This table describes the status of the scan:

Color of the device display	Status
Red	The device exists in the project but is not detected.
Blue	The device is detected but not configured.
Orange	The device is partially detected. The configuration must be updated.

Updating the Device Configuration

Click **Add/update selected device in project** in EcoStruxure Machine Expert to add or update a device. If a device connected to the controller is not detected, verify that the devices are in the same subnetwork.

Locating a Device

This function allows you to identify your target device. The **Locate** button is displayed in the Web server when a scan is launched and has started to detect devices. When the scan has detected a device, click the **Locate** button to make the LED of the target device flash.

NOTE: the locate service must be supported by your devices. Refer to the devices documentation.

Removing the Network Scan Result

Click the **Clear** button to remove the scan result.

Managing the Devices Network Settings

Setting the IP Address Configuration

You can modify the IPv4 address and subnet mask of your slave device using the command **Set IP Address**:

Step	Action
1	Click the desired device.
2	Click the locate button to make the LED of the target device flash.
3	Click the Set IP Address command. Result: The set IP menu is displayed.
4	Modify the data in the desired fields.
5	Check the save box.
6	Click the Send command button before closing.

Setting DHCP

You can use DHCP and modify the DHCP name of your slave device using the command **Set DHCP**:

Step	Action
1	Click the desired device.
2	Click the locate button to make the LED of the target device flash.
3	Click the Set DHCP command. Result: The set DHCP menu is displayed.
4	Modify the DHCP network name in the required fields.
5	Check the save box.
6	Click the Send command button before closing.

NOTE: the network name modification is applied at next power ON.

Setting BOOTP

You can use BOOTP using the command **Set BOOTP**:

Step	Action
1	Click the desired device.
2	Click the locate button to make the LED of the target device flash.
3	Click the Set BOOTP command. Result: The set BOOTP menu is displayed.
4	Check the save box.
5	Click the Send command button before closing.

Create Link/Delete Link

You can create a network link to a device using the command **Create link**. A link to the device appears and allows users to connect to the device via the Web server. You can delete the link by clicking **Delete http link**. These commands are available by using Machine Assistant in the Web server.

NOTE: You must select the **secure** option to create an operational secured link (HTTPS).

Backing Up/Restoring Configuration

Introduction

You can save and restore the application and firmware of a scanned device.

NOTE: The **Backup** button and the **Restore** button are displayed if a scan has been performed.

Backing Up Configuration

This table describes how to back up the configuration:

Step	Action
1	Insert an SD card in the master controller, page 238.
2	Click the Locate button menu to make the LED of the target device flash.
3	Click the Backup button under the commands menu. Result: The backup menu is displayed.
4	Log in (FTP username and password).
5	Click the Send command button. Result: The saved files are stored in the SD Card.

Restoring Configuration

The **Restore** button is displayed if a backup has been performed.

This table describes how to restore the configuration:

Step	Action
1	Insert the SD card which contains your saved configurations in the source controller, page 238.
2	Click the Restore button under the commands menu. Result: The restore menu is displayed.
3	Log in (FTP username and password).
4	Select the configuration to restore.
5	Click the Send command button. Result: A message is displayed asking you to reboot the device.
6	Reboot the device and restart the controller.

Exporting/Importing .semtd Files

Introduction

Machine Assistant allows you to export your project when using EcoStruxure Machine Expert or to export the scan results when using the Web server. You can import the scan results from the Web server in an empty project in EcoStruxure Machine Expert. You can also import a project from EcoStruxure Machine Expert to the Web server. You can compare the configured devices to the scanned devices.

Exporting .semtd Files

This table describes how to export an .semtd file from the Web server:

Step	Action
1	Click the scan button to scan the connected devices.
2	Click the Export scan results button
3	Save the .semtd file in your PC. Result: Your project and the detected devices during the scan are exported.

This table describes how to export an .semtd file from EcoStruxure Machine Expert:

Step	Action
1	Open your project in offline mode.
2	Click the scan button to scan your project.
3	Click the Export configuration as semtd file button.
4	Save the .semtd file on your PC. Result: Your project is exported.

Importing .semtd Files

The **load .semtd file** button allows you to upload a project in EcoStruxure Machine Expert or scanned devices in the Web server.

Appendices

What's in This Part

How to Change the IP Address of the Controller.....	266
Functions to Get/Set Serial Line Configuration in User Program.....	268
Controller Performance.....	272
M262 Logic/Motion Controller Event Messages	274

Overview

This appendix lists the documents necessary for technical understanding of the Modicon M262 Logic/Motion Controller Programming Guide.

How to Change the IP Address of the Controller

What's in This Chapter

changeIPAddress: Change the IP address of the controller 266

changeIPAddress: Change the IP address of the controller

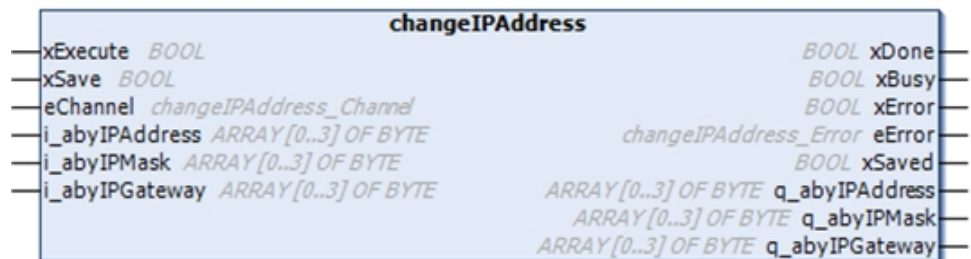
Function Block Description

The *changeIPAddress* function block provides the capability to change dynamically a controller IP address, its subnet mask and its gateway address. The function block can also save the IP address so that it is used in subsequent reboots of the controller.

NOTE: Changing the IP addresses is only possible if the IP mode is configured to **fixed IP address**. For more details, refer to IP Address Configuration, page 114.

NOTE: For more information on the function block, use the **Documentation** tab of EcoStruxure Machine Expert Library Manager Editor. For the use of this editor, refer EcoStruxure Machine Expert Functions and Libraries User Guide (see EcoStruxure Machine Expert, Functions and Libraries User Guide).

Graphical Representation



Parameter Description

Input	Type	Comment
<i>xExecute</i>	BOOL	- Rising edge: action starts. - Falling edge: resets outputs. If a falling edge occurs before the function block has completed its action, the outputs operate in the usual manner and are only reset if either the action is completed or in the event that an error is detected. In this case, the corresponding output values (<i>xDone</i>, <i>xError</i>, <i>iError</i>) are present at the outputs for exactly one cycle.
<i>xSave</i>	BOOL	TRUE: save configuration for subsequent reboots of the controller.
<i>eChannel</i>	changeIPAddress_Channel	The input <i>eChannel</i> is the Ethernet port to be configured. Depending on the number of the ports available on the controller, it is one of 5 values, page 267 in <i>changeIPAddress_Channel</i> (0 or 1).
<i>i_abyIPAddress</i>	ARRAY[0..3] OF BYTE	The new IP Address to be configured. Format: 0.0.0.0. NOTE: If this input is set to 0.0.0.0 then the controller default IP addresses, page 116 is configured.
<i>i_abyIPMask</i>	ARRAY[0..3] OF BYTE	The new subnet mask. Format: 0.0.0.0
<i>i_abyIPGateway</i>	ARRAY[0..3] OF BYTE	The new gateway IP address. Format: 0.0.0.0

Output	Type	Comment
<i>xDone</i>	BOOL	TRUE: if IP Addresses have been successfully configured or if default IP Addresses have been successfully configured because input <i>i_abyIPAddress</i> is set to 0.0.0.0.
<i>xBusy</i>	BOOL	Function block active.
<i>xError</i>	BOOL	- TRUE: error detected, function block aborts action. - FALSE: no error has been detected.
<i>eError</i>	changeIPAddress_Error	Error code of the detected error, page 267.
<i>xSaved</i>	BOOL	Configuration saved for the subsequent reboots of the controller.
<i>q_abyIPAddress</i>	ARRAY[0..3] OF BYTE	Current controller IP address. Format: 0.0.0.0.
<i>q_abyIPMask</i>	ARRAY[0..3] OF BYTE	Current subnet mask. Format: 0.0.0.0.
<i>q_abyIPGateway</i>	ARRAY[0..3] OF BYTE	Current gateway IP address. Format: 0.0.0.0.

changeIPAddress_Channel: Ethernet port to be configured

The *changeIPAddress_Channel* enumeration data type contains the following values:

Enumerator	Value	Description
<i>CHANNEL_ETHERNET_NETWORK</i>	0	M241, M251MESC, M258, LMC058, LMC078: Ethernet port M251MESE: Ethernet_2 port
<i>CHANNEL_DEVICE_NETWORK</i>	1	M241: TM4ES4 Ethernet port M251MESE: Ethernet_1 port
<i>CHANNEL_M262_ETH1</i>	2	Ethernet_1 port
<i>CHANNEL_M262_ETH2</i>	3	Ethernet_2 port
<i>CHANNEL_M262_TMS1</i>	4	1st TMS module

changeIPAddress_Error: Error Codes

The *changeIPAddress_Error* enumeration data type contains the following values:

Enumerator	Value	Description
<i>ERR_NO_ERROR</i>	00 hex	No error detected.
<i>ERR_UNKNOWN</i>	01 hex	Internal error detected.
<i>ERR_INVALID_MODE</i>	02 hex	IP address is not configured as a fixed IP address.
<i>ERR_INVALID_IP</i>	03 hex	Invalid IP address.
<i>ERR_DUPLICATE_IP</i>	04 hex	The new IP address is already used in the network.
<i>ERR_WRONG_CHANNEL</i>	05 hex	Incorrect Ethernet communication port.
<i>ERR_IP_BEING_SET</i>	06 hex	IP address is already being changed.
<i>ERR_SAVING</i>	07 hex	IP addresses not saved due to a detected error or no non-volatile memory present.
<i>ERR_DHCP_SERVER</i>	08 hex	A DHCP server is configured on this Ethernet communication port.

Functions to Get/Set Serial Line Configuration in User Program

What's in This Chapter

GetSerialConf: Get the Serial Line Configuration	268
SetSerialConf: Change the Serial Line Configuration	269
LinkNumber: Communication Port Number	270
SERIAL_CONF: Structure of the Serial Line Configuration Data Type	270

Overview

This section describes the functions to get/set the serial line configuration in your program.

To use these functions, add the **M2xx Communication** library.

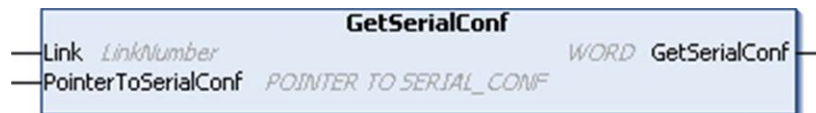
For further information on adding a library, refer to the EcoStruxure Machine Expert Programming Guide.

GetSerialConf: Get the Serial Line Configuration

Function Description

GetSerialConf returns the configuration parameters for a specific serial line communication port.

Graphical Representation



Parameter Description

Input	Type	Comment
<i>Link</i>	<i>LinkNumber</i> , page 270	<i>Link</i> is the communication port number.
<i>PointerToSerialConf</i>	<i>PointerToSerialConf</i> , page 270	<i>PointerToSerialConf</i> is the address of the configuration structure (variable of <i>SERIAL_CONF</i> type) in which the configuration parameters are stored. The <i>ADR</i> standard function must be used to define the associated pointer. (See the example below.)
Output	Type	Comment
<i>GetSerialConf</i>	WORD	This function returns: 0: The configuration parameters are returned 255: The configuration parameters are not returned because: - the function was not successful - the function is in progress

Example

Refer to the *SetSerialConf*, page 270 example.

SetSerialConf: Change the Serial Line Configuration

Function Description

SetSerialConf is used to change the serial line configuration.

Graphical Representation



NOTE: Changing the configuration of the Serial Line(s) port(s) during programming execution can interrupt ongoing communications with other connected devices.

⚠ WARNING

LOSS OF CONTROL DUE TO CONFIGURATION CHANGE

Validate and test all the parameters of the *SetSerialConf* function before putting your program into service.

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

Parameter Description

Input	Type	Comment
<i>Link</i>	<i>LinkNumber</i> , page 270	<i>LinkNumber</i> is the communication port number.
<i>PointerToSerialConf</i>	<i>PointerToSerialConf</i> , page 270	<i>PointerToSerialConf</i> is the address of the configuration structure (variable of <i>SERIAL_CONF</i> type) in which the new configuration parameters are stored. The <i>ADR</i> standard function must be used to define the associated pointer. (See the example below.) If 0, set the application default configuration to the serial line.

Output	Type	Comment
<i>SetSerialConf</i>	WORD	This function returns: 0: The new configuration is set 255: The new configuration is refused because: - the function is in progress - the input parameters are not valid

Example

```

VAR
  MySerialConf: SERIAL_CONF
  result: WORD;
END_VAR
(*Get current configuration of serial line 1*)
GetSerialConf(1, ADR(MySerialConf));
(*Change to modbus RTU slave address 9*)
MySerialConf.Protocol := 0; (*Modbus RTU/Machine
Expert protocol (in this case CodesysCompliant selects the
protocol)*)
MySerialConf.CodesysCompliant := 0; (*Modbus RTU*)
MySerialConf.address := 9; (*Set modbus address to
9*)
(*Reconfigure the serial line 1*)
result := SetSerialConf(1, ADR(MySerialConf));

```

LinkNumber: Communication Port Number

Enumerated Type Description

The *LinkNumber* enumerated data type is a list of the available communication ports. It contains these values:

Enumerator	Value (hex)	Description
<i>USBConsole</i>	00	USB port not available for communication exchanges
<i>COM1</i>	01	Serial COM 1 (embedded serial link)
<i>COM2</i>	02	Serial COM 2
<i>EthEmbed and TM4ES4</i>	03	Embedded Ethernet link and TM4ES4 expansion modules
<i>CANEmbed</i>	04	Embedded CANopen link
<i>COM3</i>	05	Serial COM 3

If one serial PCI module is installed, then the serial PCI module link is COM2, regardless of the physical PCI slots used.

If two serial PCI modules are installed, then the serial PCI module inserted on the left side PCI slots is COM2 and the serial PCI module inserted on the right side PCI slots is COM3.

SERIAL_CONF: Structure of the Serial Line Configuration Data Type

Structure Description

The *SERIAL_CONF* structure contains configuration information about the serial line port. It contains these variables:

Variable	Type	Description
<i>Bauds</i>	DWORD	Baud rate
<i>InterframeDelay</i>	WORD	Minimum time (in ms) between 2 frames in Modbus (RTU, ASCII)

Variable	Type	Description
<i>FrameReceivedTimeout</i>	WORD	In the ASCII protocol, <i>FrameReceivedTimeout</i> allows the system to conclude the end of a frame at reception after a silence of the specified number of ms. If 0 this parameter is not used.
<i>FrameLengthReceived</i>	WORD	In the ASCII protocol, <i>FrameLengthReceived</i> allows the system to conclude the end of a frame at reception, when the controller received the specified number of characters. If 0, this parameter is not used.
<i>Protocol</i>	BYTE	0: Modbus RTU or Machine Expert (see <i>CodesysCompliant</i>)
		1: Modbus ASCII
		2: ASCII
<i>Address</i>	BYTE	Modbus address 0 to 255 (0 for Master)
<i>Parity</i>	BYTE	0: none
		1: odd
		2: even
<i>Rs485</i>	BYTE	0: RS232
		1: RS485
<i>ModPol</i> (polarization resistor)	BYTE	0: no
		1: yes
<i>DataFormat</i>	BYTE	7 bits or 8 bits
<i>StopBit</i>	BYTE	1: 1 stop bit
		2: 2 stop bits
<i>CharFrameStart</i>	BYTE	In the ASCII protocol, 0 means there is no start character in the frame. Otherwise, the corresponding ASCII character is used to detect the beginning of a frame in receiving mode. In sending mode, this character is added at the beginning of the user frame.
<i>CharFrameEnd1</i>	BYTE	In the ASCII protocol, 0 means there is no second end character in the frame. Otherwise, the corresponding ASCII character is used to detect the end of a frame in receiving mode. In sending mode, this character is added at the end of the user frame.
<i>CharFrameEnd2</i>	BYTE	In the ASCII protocol, 0 means there is no second end character in the frame. Otherwise, the corresponding ASCII character is used (along with <i>CharFrameEnd1</i>) to detect the end of a frame in receiving mode. In sending mode, this character is added at the end of the user frame.
<i>CodesysCompliant</i>	BYTE	0: Modbus RTU
		1: Machine Expert (when <i>Protocol</i> = 0)
<i>CodesysNetType</i>	BYTE	not used

Controller Performance

What's in This Chapter

Processing Performance..... 272

This chapter provides information about the Modicon M262 Logic/Motion Controller processing performance.

Processing Performance

Introduction

This chapter provides information about the Modicon M262 Logic/Motion Controller processing performance.

Logic Processing

This table presents logic processing performance for various logical instructions:

IL Instruction Type	Duration for 1000 Instructions (µs)	
	TM262L01MESE8T TM262L10MESE8T TM262M05MESS8T TM262M15MESS8T	TM262L20MESE8T TM262M25MESS8T TM262M35MESS8T
Addition/subtraction/multiplication of INT	5	3
Addition/subtraction/multiplication of DINT	5	3
Addition/subtraction of REAL	11	6
Multiplication of REAL	14	7
Division of REAL	39	20
Operation on BOOLEAN, for example, Status:= Status and value	12	6
LD INT + ST INT	6	3
LD DINT + ST DINT	6	3
LD REAL + ST REAL	6	3

Retain and Retain-Persistent Performance

The retain and retain-persistent variables are saved in a dedicated memory, see NVRAM Memory Organization, page 30. Each read/write access to these variables, impacts the cycle time.

This table presents the performance impact of retain and retain-persistent variables on cycle time during POU execution:

IL Instruction Type	Duration for 1000 variables (µs)	
	TM262L01MESE8T	TM262L20MESE8T
	TM262L10MESE8T	TM262M25MESS8T
	TM262M05MESS8T	TM262M35MESS8T
	TM262M15MESS8T	
READ 1000 INT	434	377
WRITE 1000 INT	418	359
READ 1000 BYTE	434	377
WRITE 1000 BYTE	419	359
READ1000 DINT	662	685
WRITE 1000 DINT	699	539

Communication and System Processing Time

The communication processing time varies, depending on the number of requests sent and received.

Response Time on Event for Embedded Input

The response time presented in the following table represents the time between a signal rising edge on an input triggering an external task and the edge of an output set by this task:

Minimum	Typical	Maximum
60 µs	80 µs	100 µs

M262 Logic/Motion Controller Event Messages

What's in This Chapter

SysLog Messages from M262 Logic/Motion Controller 274

SysLog Messages from M262 Logic/Motion Controller

Message ID List

This table presents the list of SysLog event messages generated by the M262 Logic/Motion Controller:

ID Code	Message ID
0x001	CONNECTION_SUCCESS, page 274
0x003	CONNECTION_FAILURE, page 275
0x006	DISCONNECTION, page 275
0x20A	CONFIGURATION_CHANGE, page 275
0x403	OPERATING_MODE_CHANGE, page 276
0x406	TAMPERING, page 276
0x501	USERACCOUNT_CHANGE, page 276

CONNECTION_SUCCESS (0x001)

Characteristics	Description
Event Title	Successful connection
Event Description	Successful connections from a user (human or machine) to a machine. It can be through: - Secured protocol - Unsecured protocol if allowed by your security policy - Local interface Local port and local interface are product dependent.
Event Result	Connection OK
Protocols or Service	HTTP FTP Machine Expert Communication OPC UA
Example	<86>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 "Protocol name" CONNECTION_SUCCESS [meta sequenceId="x"] [authn@3833 itf="localPort" peer="@IpPeer:peerPort" user = "userName"]
Severity	Informational

CONNECTION_FAILURE (0x003)

Characteristics	Description
Event Title	Unsuccessful connection
Event Description	Unsuccessful connections from a user (human or machine) to a machine. It can be through: - Secured protocol - Unsecured protocol if allowed by your security policy - Local interface Standardized reasons are specified in Event Result.
Event Result	Invalid password Indeterminable user Maximum number of connections reached
Protocols or Service	HTTP FTP Machine Expert Communication OPC UA
Example	<85>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 "Protocol name" CONNECTION_FAILURE [meta sequenceId=""x"] [authn@3833 itf="localPort" peer=""peerIpAddr:peerPort" user="userName"] Max connection reached
Severity	Notice

DISCONNECTION (0x006)

Characteristics	Description
Event Title	Disconnection
Event Description	A human or a component disconnected manually or after a time-out due to inactivity. Standardized reasons are specified in Event Result (MSG).
Event Result	Manual logout
Protocols or Service	HTTP Machine Expert Communication OPC UA
Example	<86>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 "Protocol name" DISCONNECTION [meta sequenceId=""x"] [authn@3833 itf="localPort" peer="peerFQDN:peerPort" user="userName"] Manual logout
Severity	Informational

CONFIGURATION_CHANGE (0x20A)

Characteristics	Description
Event Title	Configuration change
Event Description	A new (not cyber-security related) configuration has been successfully uploaded, verified and changed. Standardized objects are Applications, Web Pages.
Event Result	-
Protocols or Service	Configuration
Example	<86>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 Configuration CONFIGURATION_CHANGE [meta sequenceId=""x"] [config@3833 object="Object" value="version"]
Severity	Informational

OPERATING_MODE_CHANGE (0x403)

Characteristics	Description
Event Title	Operating mode change
Event Description	Operating mode change (Run, Stop, Init, ...) requested by logged human user. Standardized modes are specified in Event Result (MSG).
Event Result	Init Run Stop
Protocols or Service	System
Example	<85>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 System OPERATING_MODE_CHANGE [meta sequenceId="x"] - Init
Severity	Notice

TAMPERING (0x406)

Characteristics	Description
Event Title	Detection of an attack on the system security
Event Description	Detection of hardware tampering (SL3) or detection of flash tampering during secure boot if SysLog supported by bootloader (SL2) or detection of software intrusion (SL4). Standardized intrusions are specified in Event Result (MSG).
Event Result	Network Storm
Protocols or Service	System
Example	<81>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 System TAMPERING [meta sequenceId="x"] - Physical tamper detection
Severity	Alert

USERACCOUNT_CHANGE (0x501)

Characteristics	Description
Event Title	User account creation, modification or deletion
Event Description	Creation of new ID/password or modification of ID/password or Role Based Access Control (RBAC) levels of authorization. Standardized actions are specified in Event Result (MSG).
Event Result	User account creation User account modification User account deletion Password update
Protocols or Service	Credential
Example	<86>1 2018-03-28T23:20:50.52Z "localIpAddr" M262 Credential USERACCOUNT_CHANGE [meta sequenceId="x"] [cred@3833 name="UserName"] User account creation
Severity	Informational

A

analog input:

Converts received voltage or current levels into numerical values. You can store and process these values within the logic controller.

analog output:

Converts numerical values within the logic controller and sends out proportional voltage or current levels.

application source:

The collection of human-readable controller instructions, configuration data, HMI instructions, symbols, and other program documentation. The application source file is saved on the PC and you can download the application source file to most logic controllers. The application source file is used to build the executable program that runs in the logic controller.

application:

A program including configuration data, symbols, and documentation.

ARP:

(*address resolution protocol*) An IP network layer protocol for Ethernet that maps an IP address to a MAC (hardware) address.

ASIC:

(*application specific integrated circuit*) A silicon processor (chip) custom designed especially for an application.

AT:

(*acknowledge telegram*) On Sercos bus, data are sent by the slaves to the master through AT telegrams (feedback values).

B

BCD:

(*binary coded decimal*) The format that represents decimal numbers between 0 and 9 with a set of 4 bits (a nybble/nibble, also titled as half byte). In this format, the 4 bits used to encode decimal numbers have an unused range of combinations.

For example, the number 2,450 is encoded as 0010 0100 0101 0000.

BOOL:

(*boolean*) A basic data type in computing. A `BOOL` variable can have one of these values: 0 (`FALSE`), 1 (`TRUE`). A bit that is extracted from a word is of type `BOOL`; for example, `%MW10.4` is a fifth bit of memory word number 10.

Boot application:

(*boot application*) The binary file that contains the application. Usually, it is stored in the controller and allows the controller to boot on the application that the user has generated.

BOOTP:

(bootstrap protocol) A UDP network protocol that can be used by a network client to automatically obtain an IP address (and possibly other data) from a server. The client identifies itself to the server using the client MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its pre-configured IP address. BOOTP was originally used as a method that enabled diskless hosts to be remotely booted over a network. The BOOTP process assigns an infinite lease of an IP address. The BOOTP service utilizes UDP ports 67 and 68.

byte:

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C**CA:**

(Certificate Authority) An entity that issues digital certificates to certify the ownership of a public key by the named subject of the certificate.

CAE:

(Cybersecurity Admin Expert) Schneider Electric software used by Security Administrator to manage substation security.

CFC:

(continuous function chart) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

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.

continuous function chart language:

A graphical programming language (an extension of the IEC61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to inputs of other blocks to create complex expressions.

control network:

A network containing logic controllers, SCADA systems, PCs, HMI, switches, ...

Two kinds of topologies are supported:

- flat: all modules and devices in this network belong to same subnet.
- 2 levels: the network is split into an operation network and an inter-controller network.

These two networks can be physically independent, but are generally linked by a routing device.

controller:

Automates industrial processes (also known as programmable logic controller or programmable controller).

CRC:

(*cyclical redundancy check*) A method used to determine the validity of a communication transmission. The transmission contains a bit field that constitutes a checksum. The message is used to calculate the checksum by the transmitter according to the content of the message. Receiving nodes, then recalculate the field in the same manner. Any discrepancy in the value of the 2 CRC calculations indicates that the transmitted message and the received message are different.

CRL:

(*Certificate Revocation List*) A list of digital certificates that have been revoked by the issuing Certificate Authority (CA) before their scheduled expiration date.

D**data log:**

The controller logs events relative to the user application in a *data log*.

device network:

A network that contains devices connected to a specific communication port of a logic controller. This controller is seen as a master from the devices point of view.

DHCP:

(*dynamic host configuration protocol*) An advanced extension of BOOTP. DHCP is more advanced, but both DHCP and BOOTP are common. (DHCP can handle BOOTP client requests.)

DINT:

(*double integer type*) Encoded in 32-bit format.

DNS:

(*domain name system*) The naming system for computers and devices connected to a LAN or the Internet.

DWORD:

(*double word*) Encoded in 32-bit format.

E**EDS:**

(*electronic data sheet*) A file for fieldbus device description that contains, for example, the properties of a device such as parameters and settings.

equipment:

A part of a machine including sub-assemblies such as conveyors, turntables, and so on.

Ethernet:

A physical and data link layer technology for LANs, also known as IEEE 802.3.

expansion bus:

An electronic communication bus between expansion I/O modules and a controller or bus coupler.

F

FBD:

(*function block diagram*) One of 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks, where each network contains a graphical structure of boxes and connection lines, which represents either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

FE:

(*functional Earth*) A common grounding connection to enhance or otherwise allow normal operation of electrically sensitive equipment (also referred to as functional ground in North America).

In contrast to a protective Earth (protective ground), a functional earth connection serves a purpose other than shock protection, and may normally carry current. Examples of devices that use functional earth connections include surge suppressors and electromagnetic interference filters, certain antennas, and measurement instruments.

firmware:

Represents the BIOS, data parameters, and programming instructions that constitute the operating system on a controller. The firmware is stored in non-volatile memory within the controller.

freewheeling:

When a logic controller is in freewheeling scan mode, a new task scan starts as soon as the previous scan has been completed. Contrast with *periodic scan mode*.

FreqGen:

(*frequency generator*) A function that generates a square wave signal with programmable frequency.

FTP:

(*file transfer protocol*) A standard network protocol built on a client-server architecture to exchange and manipulate files over TCP/IP based networks regardless of their size.

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.

GVL:

(*global variable list*) Manages global variables within an EcoStruxure Machine Expert project.

H

HE10:

Rectangular connector for electrical signals with frequencies below 3 MHz, complying with IEC 60807-2.

HSC:

(*high-speed counter*) A function that counts pulses on the controller or on expansion module inputs.

I

I/O:

(*input/output*)

ICMP:

(*Internet control message protocol*) Reports errors detected and provides information related to datagram processing.

IEC 61131-3:

Part 3 of a 3-part IEC standard for industrial automation equipment. IEC 61131-3 is concerned with controller programming languages and defines 2 graphical and 2 textual programming language standards. The graphical programming languages are ladder diagram and function block diagram. The textual programming languages include structured text and instruction list.

IEC:

(*international electrotechnical commission*) A non-profit and non-governmental international standards organization that prepares and publishes international standards for electrical, electronic, and related technologies.

IL:

(*instruction list*) A program written in the 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 (refer to IEC 61131-3).

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).

INT:

(*integer*) A whole number encoded in 16 bits.

IP:

(*Internet protocol*) Part of the TCP/IP protocol family that tracks the Internet addresses of devices, routes outgoing messages, and recognizes incoming messages.

K

KeepAlive:

Messages sent by the OPC UA server to keep a subscription active. This is necessary when none of the monitored items of data have been updated since the previous publication.

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).

LD:

(*ladder diagram*) 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 (refer to IEC 61131-3).

LED:

(light emitting diode) An indicator that illuminates under a low-level electrical charge.

LINT:

(long integer) A whole number encoded in a 64-bit format (4 times `INT` or 2 times `DINT`).

LRC:

(longitudinal redundancy checking) An error-detection method for determining the correctness of transmitted and stored data.

LREAL:

(long real) A floating-point number encoded in a 64-bit format.

LWORD:

(long word) A data type encoded in a 64-bit format.

M**MAC address:**

(media access control address) A unique 48-bit number associated with a specific piece of hardware. The MAC address is programmed into each network card or device when it is manufactured.

MAST:

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

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

NOTE:**MDT:**

(master data telegram) On Sercos bus, an MDT telegram is sent by the master once during each transmission cycle to transmit data (command values) to the servo drives (slaves).

MIB:

(management information base) An object database that is monitored by a network management system like SNMP. SNMP monitors devices are defined by their MIBs. Schneider Electric has obtained a private MIB, `groupeschneider (3833)`.

monitored items:

In OPC UA, the items of data (samples) made available by the OPC UA server that clients subscribe to.

MSB:

(most significant bit/byte) The part of a number, address, or field that is written as the left-most single value in conventional hexadecimal or binary notation.

ms:

(millisecond)

MST:

(master synchronization telegram) On Sercos bus, an MST telegram is broadcast by the master at the beginning of each transmission cycle to synchronize the timing of the cycle.

N

network:

A system of interconnected devices that share a common data path and protocol for communications.

node:

An addressable device on a communication network.

notifications:

In OPC UA, messages sent by the OPC UA server to inform clients that new items of data are available.

NTP:

(*Network Time Protocol*) is a protocol for synchronizing clocks, within a few milliseconds of Coordinated Universal Time (UTC), of asynchronous computer systems connected over nondeterministic data networks.

NVM:

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

O

OPC UA:

(*OPC Unified Architecture*) OPC UA is an interoperability standard for the secured and reliable exchange of data in the industrial automation space. It is a platform independent communication protocol using the server/client model. The connection between client and server is commonly based on the reliable transport layer protocol (TCP, Transmission Control Protocol).

For more information about the OPC especially OPC UA refer to the official webpage of the OPC Foundation at <https://opcfoundation.org>.

OS:

(*operating system*) A collection of software that manages computer hardware resources and provides common services for computer programs.

P

PCI:

(*peripheral component interconnect*) An industry-standard bus for attaching peripherals.

PE:

(*Protective Earth*) A common grounding connection to help avoid the hazard of electric shock by keeping any exposed conductive surface of a device at earth potential. To avoid possible voltage drop, no current is allowed to flow in this conductor (also referred to as *protective ground* in North America or as an equipment grounding conductor in the US national electrical code).

PKI:

(*Public Key Infrastructure*) A system for creating, storing, and distributing digital certificates which are used to verify that a particular public key belongs to a certain entity. The PKI creates digital certificates which map public keys to entities, securely stores these certificates in a central repository and revokes them if necessary.

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 overloading the configuration parameters of the application.

POU:

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

program:

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

protocol:

A convention or standard definition that controls or enables the connection, communication, and data transfer between 2 computing system and devices.

PTO:

(pulse train outputs) A fast output that oscillates between off and on in a fixed 50-50 duty cycle, producing a square wave form. PTO is especially well suited for applications such as stepper motors, frequency converters, and servo motor control, among others.

publishing interval:

In OPC UA, the frequency at which the OPC-UA server sends notifications to clients informing them that data updates are available.

PWM:

(pulse width modulation) A fast output that oscillates between off and on in an adjustable duty cycle, producing a rectangular wave form (though you can adjust it to produce a square wave).

R**REAL:**

A data type that is defined as a floating-point number encoded in a 32-bit format.

RJ45:

A standard type of 8-pin connector for network cables defined for Ethernet.

RPDO:

(receive process data object) An unconfirmed broadcast message or sent from a producer device to a consumer device in a CAN-based network. The transmit PDO from the producer device has a specific identifier that corresponds to the receive PDO of the consumer devices.

RPI:

(requested packet interval) The time period between cyclic data exchanges requested by the scanner. EtherNet/IP devices publish data at the rate specified by the RPI assigned to them by the scanner, and they receive message requests from the scanner with a period equal to RPI.

RSTP:

(rapid spanning tree protocol) A high-speed network protocol that builds a loop-free logical topology for Ethernet networks.

RTC:

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

run:

A command that causes the controller to scan the application program, read the physical inputs, and write to the physical outputs according to solution of the logic of the program.

S**sampling rate:**

In OPC UA, the frequency at which the OPC UA server reads items of data from connected devices.

scan:

A function that includes:

- reading inputs and placing the values in memory
- executing the application program 1 instruction at a time and storing the results in memory
- using the results to update outputs

SCEP:

(Simple Certificate Enrollment Protocol) A certificate management protocol allowing IT administrators to issue certificates automatically by standardizing the exchange with the CA. The certificates can be enrolled on devices on a large-scale.

SDO:

(service data object) A message used by the field bus master to access (read/write) the object directories of network nodes in CAN-based networks. SDO types include service SDOs (SSDOs) and client SDOs (CSDOs).

Sercos:

(serial real-time communications system) A digital control bus that interconnects, motion controls, drives, I/Os, sensors, and actuators for numerically controlled machines and systems. It is a standardized and open controller-to-intelligent digital device interface, designed for high-speed serial communication of standardized closed-loop real-time data.

SFC:

(sequential function chart) A language that is composed of steps with associated actions, transitions with associated logic condition, and directed links between steps and transitions. (The SFC standard is defined in IEC 848. It is IEC 61131-3 compliant.)

SINT:

(signed integer) A 15-bit value plus sign.

SNMP:

(simple network management protocol) A protocol that can control a network remotely by polling the devices for their status and viewing information related to data transmission. You can also use it to manage software and databases remotely. The protocol also permits active management tasks, such as modifying and applying a new configuration.

STOP:

A command that causes the controller to stop running an application program.

string:

A variable that is a series of ASCII characters.

ST:

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

T**task:**

A group of sections and subroutines, executed cyclically or periodically for the MAST task or periodically for the FAST task.

A task possesses a level of priority and is linked to inputs and outputs of the controller. These I/O are refreshed in relation to the task.

A controller can have several tasks.

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.

terminal block:

(*terminal block*) The component that mounts in an electronic module and provides electrical connections between the controller and the field devices.

TLS:

(*Transport Layer Security*) is a secure protocol technique used to protect information over a computer network.

U**UDINT:**

(*unsigned double integer*) Encoded in 32 bits.

UDP:

(*user datagram protocol*) A connectionless mode protocol (defined by IETF RFC 768) in which messages are delivered in a datagram (data telegram) to a destination computer on an IP network. The UDP protocol is typically bundled with the Internet protocol. UDP/IP messages do not expect a response, and are therefore ideal for applications in which dropped packets do not require retransmission (such as streaming video and networks that demand real-time performance).

UINT:

(*unsigned integer*) Encoded in 16 bits.

V**variable:**

A memory unit that is addressed and modified by a program.

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.

WORD:

A type encoded in a 16-bit format.

Index

A

Adding an Encoder	
Incremental Encoder	84
SSI Encoder	84
ASCII Manager	188

C

changeIPAddress	266
changing the controller IP address	266
ControlChannel	196
Enables or disables a communication channel ...	196
controller configuration	
communication settings	61
Controller Configuration	
NTP	67
PLC Settings	62
Services	63
cyclic data exchanges, generating EDS file for	158

D

Data Types	
LinkNumber	270
DHCP server	156
Download application	57

E

EDS file, generating	158
Embedded Functions Configuration	
Embedded I/O Configuration	79
Enables or disables a communication channel	
ControlChannel	196
Ethernet	
changeIPAddress function block	266
FTP Server	120
Modbus TCP Client/Server	119
Modbus TCP slave device	175
Services	112
SNMP	121
Web server	122
EtherNet	
EtherNet/IP device	156
EtherNet/IP Adapter	157
External Event	35

F

Fast Device Replacement	156
features	
key features	13
firewall	
configuration	144
default script file	144
script commands	146
firmware	
downloading to TMS expansion modules	230
FTP Server	
Ethernet	120

G

GetSerialConf	268
getting the serial line configuration	268

H

Hardware Initialization Values	49

I

I/O bus configuration	107
I/O configuration general information	
general practices	102
Industrial Ethernet	
overview	152
Industrial Plug and Work	260
IP address	
changeIPAddress	266

K

KeepAlive (OPC UA)	201
KeepAlive interval (OPC UA)	203

L

libraries	23
LinkNumber	270
Data Types	270

M

M2•• communication	
GetSerialConf	268
LinkNumber	270
SERIAL_CONF	270
SetSerialConf	269
Machine Assistant	260
Memory Mapping	25
Modbus	
Protocols	119
Modbus Ioscaner	190
Modbus Manager	185
Modbus TCP Client/Server	
Ethernet	119
monitored items (OPC UA)	201

O

OPC UA server	
configuration	202
KeepAlive interval	203
overview	201
publishing interval	203
sampling interval	203
selecting symbols	209
symbols configuration	208
Output Behavior	49–50
Output Forcing	50
overview of the Sercos standard	179

P

Post Configuration	215
baud rate	215

CAE Enable	215
data bits	215
device name	215
Example	218
file management	216
FTP	215
gateway address	215
IP address	215
IP configuration mode	215
ODVA Enable	215
parity	215
presentation	215
stop bit	215
subnet mask	215
WebVisualisation	215
programming languages	
IL, LD, Grafcet	13
protocols	
SNMP	121
Protocols	112
IP	114
Modbus	119
publishing interval (OPC UA)	201, 203

R

Reboot	55
Remanent variables	58
Reset cold	52
Reset origin	52
Reset origin device	53
Reset warm	51
Run command	50

S

sampling interval (OPC UA)	201, 203
script commands	
firewall	146
serial line	
ASCII Manager	188
GetSerialConf	268
Modbus Manager	185
SERIAL_CONF	270
SetSerialConf	269
SERIAL_CONF	270
SetSerialConf	269
setting the serial line configuration	269
SNMP	
Ethernet	121
protocols	121
Software Initialization Values	49
State diagram	42
Stop command	51
symbols (OPC UA)	208

T

Task	
Cyclic task	34
Event task	35
External Event Task	35
Freewheeling task	34
Types	33
Watchdogs	37
TMS analog I/O modules	
downloading firmware to	230

U

updating the firmware of TM3 expansion	
modules	226
updating the firmware of TMSES4 expansion	
module	229

W

Web server	
Ethernet	122

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2022 Schneider Electric. All rights reserved.

EIO0000003651.08

Modicon M262

Logic/Motion Controller

System Functions and Variables

System Library Guide

EIO0000003667.04
06/2022



Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2022 – Schneider Electric. All rights reserved.

Table of Contents

About the Book.....	7
M262 PLCSystem	11
M262 System Variables.....	12
System Variables: Definition and Use	12
Understanding System Variables	12
Using System Variables	13
PLC_R and PLC_W Structures	14
PLC_R: Controller Read-Only System Variables	14
PLC_W: Controller Read/Write System Variables	18
ETH_R and ETH_W Structures	18
ETH_R: Ethernet Port Read-Only System Variables	19
ETH_W: Ethernet Port Read/Write System Variables	21
M262 System Functions.....	22
M262 Read Functions	22
GetImmediateFastInput: Read Input of an Embedded Expert I/	
O	22
GetRtc: Get Real Time Clock	23
IsFirstMastColdCycle: Indicate if this Cycle is the First MAST	
Cold Start Cycle	24
IsFirstMastCycle: Indicate if this Cycle is the First MAST	
Cycle.....	24
IsFirstMastWarmCycle: Indicate if this Cycle is the First MAST	
Warm Start Cycle	26
GetExternalEventValue: Get Current Value of an External	
Event	26
M262 Write Functions	27
PhysicalWriteFastOutputs: Write Fast Output of an Embedded	
Expert I/O	27
SetRTCDrift: Set Compensation Value to the RTC	28
M262 User Functions	29
FB_GetFreeDiskSpace: Gets the Free Memory Space	
Asynchronously	30
FB_GetLabel: Gets the Label of Memory Medium.....	31
FB_GetTotalDiskSpace: Gets the Size of Memory Medium.....	32
FB_CheckAllowedControllerMacAddr: Check If MAC Address	
Allowed by Controller.....	33
FB_ControlClone: Clone the Controller	34
DataFileCopy: Copy File Commands	35
ExecuteScript: Run Script Commands	37
M262 Library Data Types	39
PLC_RW System Variables Data Types.....	39
PLC_R_APPLICATION_ERROR: Detected Application Error	
Status Codes	39
PLC_R_BOOT_PROJECT_STATUS: Boot Project Status	
Codes	41
PLC_R_IO_STATUS: I/O Status Codes	41
PLC_R_SDCARD_STATUS: SD Card Slot Status Codes	42
PLC_R_STATUS: Controller Status Codes.....	42

<i>PLC_R_STOP_CAUSE</i> : From RUN State to Other State	
Transition Cause Codes	43
<i>PLC_R_TERMINAL_PORT_STATUS</i> : Programming Port	
Connection Status Codes	45
<i>PLC_R_TM3_BUS_STATE</i> : TM3 Bus Status Codes	45
<i>PLC_W_COMMAND</i> : Control Command Codes	45
DataFileCopy System Variables Data Types	46
<i>DataFileCopyError</i> : Detected Error Codes	46
<i>DataFileCopyLocation</i> : Location Codes	46
ExecScript System Variables Data Types	47
<i>ExecuteScriptError</i> : Detected Error Codes	47
ETH_RW System Variables Data Types	47
<i>ETH_R_FRAME_PROTOCOL</i> : Frame Transmission Protocol	
Codes	48
<i>ETH_R_IPFORWARDING</i> : IP Forwarding	48
<i>ETH_R_IP_MODE</i> : IP Address Source Codes	48
<i>ETH_R_ITF_STRUCT</i> : Ethernet Interface Parameters	49
<i>ETH_R_PORT_DUPLEX_STATUS</i> : Transmission Mode	
Codes	50
<i>ETH_R_PORT_IP_STATUS</i> : Ethernet TCP/IP Port Status	
Codes	50
<i>ETH_R_PORT_LINK_STATUS</i> : Communication Link Status	
Codes	51
<i>ETH_R_PORT_SPEED</i> : Communication Speed of the Ethernet	
Port Codes	51
<i>ETH_R_RUN_IDLE</i> : Ethernet/IP Run and Idle States Codes	51
System Function Data Types	52
<i>IMMEDIATE_ERR_TYPE</i> : <i>GetImmediateFastInput</i> Read Input of	
Embedded Expert I/O Codes	52
<i>RTCSETDRIFT_ERROR</i> : <i>SetRTCDrift</i> Function Detected Error	
Codes	53
SerialLine System	54
M262 Serial Line System Variables	55
<i>SERIAL_R</i> : Serial Line Diagnostic Variables	55
<i>SERIAL_W</i> : Serial Line Diagnostic Variables	56
TM3 System	57
TM3 System Variables	58
<i>TM3_MODULE_R[0...13]</i> : TM3 Modules Read-Only System	
Variables	58
TM3 System Functions	59
<i>storetm3bus_w</i> : Change the TM3 Management Mode	59
<i>TM3_GetModuleBusStatus</i> : Get TM3 Module Bus Status	60
<i>TM3_GetModuleInternalStatus</i> : Get TM3 Module Internal Status	60
<i>TM3_SendDc2Cmd</i> : Send a DC2 Command to the TM3 bus	62
TM3 System Data Types	64
<i>TM3_BUS_PARAM_ID</i> : TM3 Expansion Module Diagnostics	64
<i>TM3_BUS_W_IJOBUSERMOD</i> : TM3 bus error mode	64
<i>TM3_BUS_W_IJOBUSINIT</i> : Reset the Bus Communication	64
<i>TM3_BUS_W</i> : TM3 Bus System Variables	65
<i>TM3_ERR_CODE</i> : TM3 Expansion Module Detected Error	
Codes	65

<i>TM3_MODULE_R_ARRAY_TYPE</i> : TM3 Expansion Module Read Array Type.....	66
<i>TM3_MODULE_STATE</i> : TM3 Expansion Module State Codes	66
TMS System	67
TMS System Variables.....	68
<i>TMS_BUS_DIAG_R</i> : TMS Bus Diagnostic Error Codes	68
<i>TMS_MODULE_DIAG_R</i> : TMS Expansion Module Diagnostic Error Codes	68
TMS System Data Types.....	70
<i>TMS_IP_STATE</i> : TMS Expansion Module IP State	70
<i>TMS_MODULE_STATE</i> : TMS Expansion Module State Codes	70
<i>TMS_PIXCMD_STATE</i> : TMS Expansion Module PIXCMD State	71
Appendices	73
Function and Function Block Representation	74
Differences Between a Function and a Function Block	74
How to Use a Function or a Function Block in IL Language	75
How to Use a Function or a Function Block in ST Language	77
Glossary	81
Index	87

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

 DANGER DANGER indicates a hazardous situation which, if not avoided, will result in death or serious injury.
 WARNING WARNING indicates a hazardous situation which, if not avoided, could result in death or serious injury.
 CAUTION CAUTION indicates a hazardous situation which, if not avoided, could result in minor or moderate injury.
NOTICE NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

Document Scope

This document will acquaint you with the system functions and variables offered within the Modicon M262 Logic/Motion Controller. The M262 System library contains functions and variables to get information from and send commands to the controller system.

This document describes the data type functions and variables of the following M262 system libraries:

- M262 PLCSystem
- Serial Line System
- TM3 System
- TMS System

The following knowledge is required:

- Basic information on the functionality, structure, and configuration of the M262 Logic/Motion Controller.
- Programming in the FBD, LD, ST, IL, or CFC language.
- System variables (global variables).

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.3.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Related Documents

Title of Documentation	Reference Number
EcoStruxure Machine Expert - Programming Guide	EIO0000002854 (ENG)
	EIO0000002855 (FRE)
	EIO0000002856 (GER)
	EIO0000002858 (SPA)
	EIO0000002857 (ITA)
	EIO0000002859 (CHS)
Modicon M262 Logic/Motion Controller - Hardware Guide	EIO0000003659 (ENG)
	EIO0000003660 (FRE);
	EIO0000003661 (GER)
	EIO0000003662 (SPA)
	EIO0000003663 (ITA)
	EIO0000003664 (CHS)
	EIO0000003665 (POR)
Modicon M262 Logic/Motion Controller - Programming Guide	EIO0000003651 (ENG)
	EIO0000003652 (FRA)
	EIO0000003653 (GER)
	EIO0000003654 (SPA)
	EIO0000003655 (ITA)
	EIO0000003656 (CHS)
	EIO0000003657 (POR)
	EIO0000003658 (TUR)

Product Related Information

WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

M262 PLCSystem

What's in This Part

M262 System Variables	12
M262 System Functions	22
M262 Library Data Types	39

Introduction

This part describes the M262 PLCSystem Library.

M262 System Variables

What's in This Chapter

System Variables: Definition and Use.....	12
<i>PLC_R</i> and <i>PLC_W</i> Structures.....	14
<i>ETH_R</i> and <i>ETH_W</i> Structures	18

Overview

This chapter:

- gives an introduction to the system variables, page 12
- describes the system variables, page 14 included with the M262 PLCSystem library

System Variables: Definition and Use

Overview

This section defines system variables and how to implement them in the Modicon M262 Logic/Motion Controller.

Understanding System Variables

Introduction

This section describes how system variables are implemented. System variables:

- allow you to access general system information, perform system diagnostics, and command simple actions.
- are structured variables conforming to IEC 61131-3 definitions and naming conventions. You can access the system variables using the IEC symbolic name *PLC_GVL*. Some of the *PLC_GVL* variables are read-only (for example, *PLC_R*) and some are read/write (for example, *PLC_W*).
- are automatically declared as global variables. They have system-wide scope and can be accessed by any Program Organization Unit (POU) in any task.

Naming Convention

The system variables are identified by:

- a structure name that represents the category of system variable. For example, *PLC_R* represents a structure name of read-only variables used for the controller diagnostic.
- a set of component names that identifies the purpose of the variable. For example, *i_wVendorID* represents the controller vendor ID.

You can access the system variables by typing the structure name of the variables followed by the name of the component.

Here is an example of system variable implementation:

```
VAR
myCtr_Serial : DWORD;
myCtr_ID : DWORD;
myCtr_FramesRx : UDINT;
END_VAR
```

```
myCtr_Serial := PLC_GVL.PLC_R.i_dwSerialNumber;
myCtr_ID := PLC_GVL.PLC_R.i_wVendorID;
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK
```

NOTE: The fully-qualified name of the system variable in the example above is `PLC_GVL.PLC_R`. The `PLC_GVL` is implicit when declaring a variable using the **Input Assistant**, but it may also be entered with the prefix. Good programming practice often dictates the use of the fully-qualified variable name in declarations.

System Variables Location

Two kinds of system variables are defined for use when programming the controller:

- located variables
- unlocated variables

The located variables:

- are accessible through Modbus TCP, Modbus serial, and EtherNet/IP requests both in RUNNING and STOPPED states.
- are used in EcoStruxure Machine Expert programs according to the *structure_name.component_name* convention explained previously. %MW addresses from 0 to 59999 can be accessed directly. Addresses greater than this are considered out of range by EcoStruxure Machine Expert and can only be accessed through the *structure_name.component_name* convention.

The unlocated variables:

- are not physically located in the %MW area.
- are not accessible through any fieldbus or network requests unless you locate them in the relocation table, and only then these variables can be accessed in RUNNING and STOPPED states. The relocation table uses the following dynamic %MW areas:
 - %MW60200 to %MW61999 for read-only variables
 - %MW62200 to %MW63999 for read/write variables
- are used in EcoStruxure Machine Expert programs according to the *structure_name.component_name* convention explained previously.

Using System Variables

Introduction

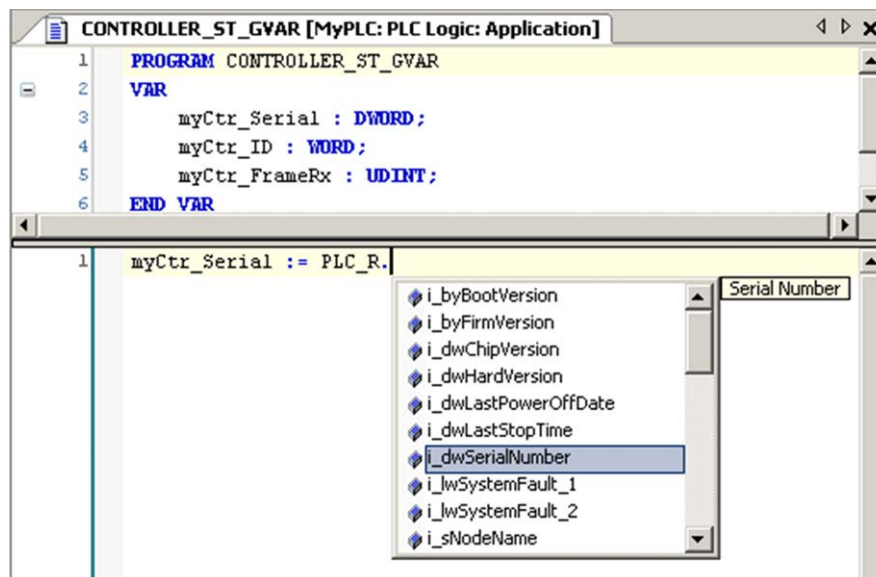
This section describes the steps required to program and to use system variables in EcoStruxure Machine Expert.

System variables are global in scope, and you can use them in all the Program Organization Units (POUs) of the application.

System variables do not need to be declared in the Global Variable List (GVL). They are automatically declared from the controller system library.

Using System Variables in a POU

EcoStruxure Machine Expert has an auto-completion feature. In a **POU**, start by entering the system variable structure name (*PLC_R PLC_W...*) followed by a dot. The system variables appear in the **Input Assistant**. You can select the desired variable or enter the full name manually.



NOTE: In the example above, after you type the structure name *PLC_R*, EcoStruxure Machine Expert offers a pop-up menu of possible component names/variables.

Example

The following example shows the use of some system variables:

```

VAR
myCtr_Serial : DWORD;
myCtr_ID : WORD;
myCtr_FramesRx : UDINT;
END_VAR
myCtr_Serial := PLC_R.i_dwSerialNumber;
myCtr_ID := PLC_R.i_wVendorID;
myCtr_FramesRx := SERIAL_R[0].i_udiFramesReceivedOK;

```

PLC_R and PLC_W Structures

Overview

This section lists and describes the different system variables *SEC.PLC_GVL*, *PLC_R* and *SEC.PLC_GVL.PLC_W* of the library *SE_PLCSys* using the Namespace *SEC*.

The structures of variables are defined in the *PLCSysBase* library.

PLC_R: Controller Read-Only System Variables

Library and Namespace

Library name: **SE_PLCSys**

Namespace: **SEC**

Variable Structure

This table describes the parameters of the *PLC_R* system variable (*PLC_R_STRUCT* type):

Modbus Address ⁽¹⁾	Variable Name	Type	Comment
60000	<i>i_wVendorID</i>	WORD	Controller Vendor ID. 101A hex = Schneider Electric
60001	<i>i_wProductID</i>	WORD	Controller Reference ID. NOTE: Vendor ID and Reference ID are the components of the Target ID of the controller displayed in the communication settings view (Target ID = 101A XXXX hex). Value per controller: • TM262L01MESE8T: 101A 0816 HEX • TM262L10MESE8T: 101A 0813 HEX • TM262L20MESE8T: 101A 0810 HEX • TM262M05MESS8T: 101A 0815 HEX • TM262M15MESS8T: 101A 0814 HEX • TM262M25MESS8T: 101A 0811 HEX • TM262M35MESS8T: 101A 0812 HEX
60002	<i>i_dwSerialNumber</i>	DWORD	Controller Serial Number
60004	<i>i_byFirmVersion</i>	ARRAY[0..3] OF BYTE	Controller Firmware Version [aa.bb.cc.dd]: • <i>i_byFirmVersion</i>[0] = aa • ... • <i>i_byFirmVersion</i>[3] = dd
60006	<i>i_byBootVersion</i>	ARRAY[0..3] OF BYTE	Controller Boot Version [aa.bb.cc.dd]: • <i>i_byBootVersion</i>[0] = aa • ... • <i>i_byBootVersion</i>[3] = dd
60008	<i>i_dwHardVersion</i>	DWORD	Controller Hardware Version.
60010	<i>i_dwChipVersion</i>	DWORD	Controller Coprocessor Version.
60012	<i>i_wStatus</i>	<i>PLC_R_STATUS</i> , page 42	State of the controller.
60013	<i>i_wBootProjectStatus</i>	<i>PLC_R_BOOT_PROJECT_STATUS</i> , page 41	Returns information about the boot application stored in non-volatile memory.
60014	<i>i_wLastStopCause</i>	<i>PLC_R_STOP_CAUSE</i> , page 43	Cause of the last transition from RUN to another state.
60015	<i>i_wLastApplicationError</i>	<i>PLC_R_APPLICATION_ERROR</i> , page 39	Cause of the last controller exception.

Modbus Address ⁽¹⁾	Variable Name	Type	Comment
60016	<i>i_lwSystemFault_1</i>	LWORD	Bit field FFFF FFFF FFFF FFFF hex indicates no error detected. A bit at low level (0) means that an error has been detected: • bit 0 = Expert I/O error detected • bit 1 = I/O bus error detected • bit 2 = Ethernet IF1 error detected • bit 3 = Ethernet IF2 error detected • bit 4 = Serial 1 in overcurrent error detected • bit 5 = Serial 2 in overcurrent error detected • bit 6 = CAN 1 error detected • bit 7 = Reserved • bit 8 = Reserved • bit 9 = Communication bus error detected • bit 10 = SD Card error detected • bit 11 = Firewall error detected • bit 12 = DHCP/PS/FDR server error detected • bit 13 = OPC UA server error detected • bit 14 = Communication bus error detected • bit 15 = Communication bus health error detected • bit 16 = Ethernet IF2 Ring root error detected • bit 17 = Encoder power supply error detected • bit 18 = Encoder communication error detected • bit 19 = TMS4 IF1 configuration error detected • bit 20 = TMS4 IF2 configuration error detected • bit 21 = TMS4 IF3 configuration error detected • bit 22 = Server address error detected • bit 23 = NTP error detected • bit 24 = Syslog error detected
60020	<i>i_lwSystemFault_2</i>	LWORD	Bit field FFFF hex indicates no error detected. If <i>i_wIOStatus1</i> = <i>PLC_R_IO_SHORTCUT_FAULT</i>, the meaning of <i>i_lwSystemFault_2</i> is: • bit 0 = 0: short-circuit detected in output group 0 (Q0...Q1) • bit 1 = 0: short-circuit detected in output group 1 (Q2...Q3) • bit 2 = 0: short-circuit detected in output group 2 (Q4...Q7) • bit 3 = 0: short-circuit detected in output group 3 (Q8...Q11) • bit 4 = 0: short-circuit detected in output group 4 (Q12...Q15)
60024	<i>i_wIOStatus1</i>	<i>PLC_R_IO_STATUS</i> , page 41	Embedded Expert I/O status.
60025	<i>i_wIOStatus2</i>	<i>PLC_R_IO_STATUS</i> , page 41	TM3 I/O status.
60026	<i>i_wClockBatterystatus</i>	WORD	Status of the battery of the RTC: 100 = Battery fully charged Not applicable for M262 Logic/Motion Controller.
60028	<i>i_dwAppliSignature1</i>	DWORD	First DWORD of 4 DWORD signature (16 bytes total). The application signature is generated by the software during build.
60030	<i>i_dwAppliSignature2</i>	DWORD	Second DWORD of 4 DWORD signature (16 bytes total). The application signature is generated by the software during build.

Modbus Address ⁽¹⁾	Variable Name	Type	Comment
60032	<i>i_dwAppliSignature3</i>	DWORD	Third DWORD of 4 DWORD signature (16 bytes total). The application signature is generated by the software during build.
60034	<i>i_dwAppliSignature4</i>	DWORD	Fourth DWORD of 4 DWORD signature (16 bytes total). The application signature is generated by the software during build.
n/a	<i>i_sVendorName</i>	STRING(31)	Name of the vendor: "Schneider Electric".
n/a	<i>i_sProductRef</i>	STRING(31)	Reference of the controller.
n/a	<i>i_sNodeName</i>	STRING(99)	Node name on EcoStruxure Machine Expert Network.
n/a	<i>i_dwLastStopTime</i>	DWORD	The time of the last detected STOP in seconds beginning with January 1, 1970 at 00:00 UTC.
n/a	<i>i_dwLastPowerOffDate</i>	DWORD	The date and time of the last detected power off in seconds beginning with January 1, 1970 at 00:00 UTC. NOTE: Convert this value into date and time by using the function <i>SysTimeRtcConvertUtcToDate</i> . For more information about Time and Date conversion, refer to the Systime Library Guide (see SoMachine, Getting & Setting Real Time Clock, SysTimeRtc and SysTimeCore Library Guide).
n/a	<i>i_uiEventsCounter</i>	UINT	Number of external events detected on inputs configured for external event detection since the last cold start. Reset by a Cold Start or by the <i>PLC_W.q_wResetCounterEvent</i> command.
n/a	<i>i_wTerminalPortStatus</i>	<i>PLC_R_TERMINAL_PORT_STATUS</i> , page 45	Status of the USB Programming Port (USB Mini-B).
n/a	<i>i_wSdCardStatus</i>	<i>PLC_R_SDCARD_STATUS</i> , page 42	Status of the SD card.
n/a	<i>i_wUsrFreeFileHdl</i>	WORD	Number of available File Handles. A File Handle is the resource allocated by the system when you open a file.
n/a	<i>i_udiUsrFsTotalBytes</i>	UDINT	User FileSystem total memory size (in bytes). It is the size of the non-volatile memory for the directory "/usr/".
n/a	<i>i_udiUsrFsFreeBytes</i>	UDINT	User FileSystem free memory size (in bytes).
n/a	<i>i_uiTM3BusState</i>	<i>PLC_R_TM3_BUS_STATE</i> , page 45	TM3 bus state. <i>i_uiTM3BusState</i> can have the following values: 1: TM3_CONF_ERROR Configuration mismatch between physical configuration and EcoStruxure Machine Expert configuration. 3: TM3_OK Physical configuration matches EcoStruxure Machine Expert configuration. 4: TM3_POWER_SUPPLY_ERROR TM3 bus is not powered (for example when the controller is powered by USB).
n/a	<i>i_ExpertIO_RunStop_Input</i>	BYTE	Run/Stop input location is: 16...FF hex if the expert I/O is not configured 0 for %IX0.0 1 for %IX0.1 2 for %IX0.2 ...and so on.

Modbus Address ⁽¹⁾	Variable Name	Type	Comment
n/a	<i>i_x10msClk</i>	BOOL	TimeBase bit of 10 ms. This variable is toggling On/Off with period = 10 ms. The value toggles when the controller is in STOPPED and in RUN state.
n/a	<i>i_x100msClk</i>	BOOL	TimeBase bit of 100 ms. This variable is toggling On/Off with period = 100 ms. The value toggles when the controller is in STOPPED and in RUN state.
n/a	<i>i_x1sClk</i>	BOOL	TimeBase bit of 1 s. This variable is toggling On/Off with period = 1 s. The value toggles when the controller is in STOPPED and in RUN state.
(1) Not accessible through the application as %MW. n/a means there is no pre-defined Modbus address mapping for this system variable.			

PLC_W: Controller Read/Write System Variables

Library and Namespace

Library name: **SE_PLCSys**tem

Namespace: **SEC**

Variable Structure

This table describes the parameters of the *PLC_W* system variable (*PLC_W_STRUCT* type):

%MW	Variable Name	Type	Comment
n/a	<i>q_wResetCounterEvent</i>	WORD	Transition from 0 to 1 resets the events counter (<i>PLC_R.i_uiEventsCounter</i>). To reset the counter again, it is necessary to write 0 to this variable before another transition from 0 to 1 can take place.
n/a	<i>q_uiOpenPLCControl</i>	UINT	When the value of the variable passes from 0 to 6699, the command previously written in the following <i>PLC_W.q_wPLCControl</i> is executed.
n/a	<i>q_wPLCControl</i>	<i>PLC_W_COMMAND</i> , page 45	Controller RUN / STOP command executed when the system variable <i>PLC_W.q_uiOpenPLCControl</i> value passes from 0 to 6699.
n/a means that there is no pre-defined %MW mapping for this system variable			

ETH_R and ETH_W Structures

Overview

This section lists and describes the different system variables *SEC.PLC_GVL.ETH_R* and *SEC.PLC_GVL.ETH_W* of the library *SE_PLCSys*tem using the Namespace *SEC*.

The structures of variables are defined in the *PLCSys*temBase library.

ETH_R: Ethernet Port Read-Only System Variables

Library and Namespace

Library name: **SE_PLCSYSTEM**

Namespace: **SEC**

Variable Structure

This table describes the parameters of the *ETH_R* system variable (*ETH_R_STRUCT* type). One structure exists for each Ethernet port:

%MW	Var Name	Type	Comment
60050	<i>i_byIPAddress</i>	ARRAY[0..3] OF BYTE	IP address of the Ethernet_1 or Ethernet_2 interface [aaa.bbb.ccc.ddd]: <i>i_byIPAddress</i>[0] = aaa ... <i>i_byIPAddress</i>[3] = ddd
60052	<i>i_bySubNetMask</i>	ARRAY[0..3] OF BYTE	Subnet Mask of the Ethernet_1 or Ethernet_2 interface [aaa.bbb.ccc.ddd]: <i>i_bySub-netMask</i>[0] = aaa ... <i>i_bySub-netMask</i>[3] = ddd
60054	<i>i_byGateway</i>	ARRAY[0..3] OF BYTE	Gateway address of the Ethernet_1 or Ethernet_2 interface [aaa.bbb.ccc.ddd]: <i>i_byGateway</i>[0] = aaa ... <i>i_byGateway</i>[3] = ddd
60056	<i>i_byMACAddress</i>	ARRAY[0..5] OF BYTE	MAC address of the Ethernet_1 or Ethernet_2 interface [aa.bb.cc.dd.ee.ff]: <i>i_byMACAddress</i>[0] = aa ... <i>i_byMACAddress</i>[5] = ff
60059	<i>i_sDeviceName</i>	STRING(15)	Name used to get IP address from server.
n/a	<i>i_uciPForwarding</i>	<i>ETH_R_IPFORWARDING</i> , page 48	IP Forwarding.
n/a	<i>i_wIpMode</i>	<i>ETH_R_IP_MODE</i> , page 48	Method used to obtain an IP address.
n/a	<i>i_byFDRServerIPAddress</i>	ARRAY[0..3] OF BYTE	The IP address [aaa.bbb.ccc.ddd] of the DHCP or BootP server: <i>i_byFDRServerIPAddress</i>[0] = aaa ... <i>i_byFDRServerIPAddress</i>[3] = ddd Equals 0.0.0.0 if Stored IP or Default IP used.
n/a	<i>i_udiOpenTcpConnections</i>	UDINT	Number of open TCP connections.
n/a	<i>i_udiFramesTransmittedOK</i>	UDINT	Number of frames successfully transmitted. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiFramedReceivedOK</i>	UDINT	Number of frames successfully received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiTransmitBufferErrors</i>	UDINT	Numbers of frames transmitted with detected errors. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiReceiveBufferErrors</i>	UDINT	Numbers of frames received with detected errors. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .

%MW	Var Name	Type	Comment
n/a	<i>i_wFrameSendingProtocol</i>	ETH_R_FRAME_PROTOCOL, page 48	Ethernet protocol configured for frames sending (IEEE 802.3 or Ethernet II).
n/a	<i>i_wPortALinkStatus</i>	ETH_R_PORT_LINK_STATUS, page 51	Link of the Ethernet Port (0 = No Link, 1 = Link connected to another Ethernet device).
n/a	<i>i_wPortASpeed</i>	ETH_R_PORT_SPEED, page 51	Ethernet Port network speed (10Mb/s, 100Mb/s, or 1Gb/s).
n/a	<i>i_wPortADuplexStatus</i>	ETH_R_PORT_DUPLEX_STATUS, page 50	Ethernet Port duplex status (0 = Half or 1 = Full duplex).
n/a	<i>i_udiPortACollisions</i>	UDINT	Number of frames involved in one or more collisions and subsequently transmitted successfully. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_wPortAlpStatus</i>	ETH_R_PORT_IP_STATUS, page 50	Ethernet TCP/IP port stack status.
n/a	<i>i_ethInterface</i>	ARRAY[1..6] OF ETH_R_ITF_STRUCT, page 49	Ethernet interface common parameters structure.
Modbus TCP/IP Specific			
n/a	<i>i_udiModbusMessageTransmitted</i>	UDINT	Number of Modbus messages transmitted. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiModbusMessageReceived</i>	UDINT	Number of Modbus messages received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiModbusErrorMessage</i>	UDINT	Modbus detected error messages transmitted and received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_byMasterIpTimeouts</i>	BYTE	Ethernet Modbus TCP Master timeout events counter. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_byMasterIpLost</i>	BYTE	Ethernet Modbus TCP Master link status: 0 = link OK, 1 = link lost.
EtherNet/IP Specific			
n/a	<i>i_udiETHIP_IOMessagingTransmitted</i>	UDINT	EtherNet/IP Class 1 frames transmitted. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiETHIP_IOMessagingReceived</i>	UDINT	EtherNet/IP Class 1 frames received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiUCMM_Request</i>	UDINT	EtherNet/IP Unconnected Messages received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiUCMM_Error</i>	UDINT	EtherNet/IP invalid Unconnected Messages received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiClass3_Request</i>	UDINT	EtherNet/IP Class 3 requests received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiClass3_Error</i>	UDINT	EtherNet/IP invalid class 3 requests received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .

%MW	Var Name	Type	Comment
n/a	<i>i_uiAssemblyInstanceInput</i>	UINT	Input Assembly Instance number. See the appropriate Programming Guide of your controller for more information.
n/a	<i>i_uiAssemblyInstanceInputSize</i>	UINT	Input Assembly Instance size. See the appropriate Programming Guide of your controller for more information.
n/a	<i>i_uiAssemblyInstanceOutput</i>	UINT	Output Assembly Instance number. See the appropriate Programming Guide of your controller for more information.
n/a	<i>i_uiAssemblyInstanceOutputSize</i>	UINT	Output Assembly Instance size. See the appropriate Programming Guide of your controller for more information.
n/a	<i>i_uiETHIP_ConnectionTimeouts</i>	UINT	Number of connection timeouts. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_ucEipRunIdle</i>	<i>ETH_R_RUN_IDLE</i> , page 51	Run (value = 1)/Idle(value = 0) flag for EtherNet/IP class 1 connection.
n/a means that there is no predefined %MW mapping for this system variable.			

ETH_W: Ethernet Port Read/Write System Variables

Library and Namespace

Library name: **SE_PLCSYSTEM**

Namespace: **SEC**

Variable Structure

This table describes the parameters of the *ETH_W* system variable (*ETH_W_STRUCT* type):

%MW	Variable Name	Type	Comment
n/a	<i>q_wResetCounter</i>	WORD	Transition from 0 to 1 resets all <i>ETH_R</i> counters. To reset again, it is necessary to write 0 to this variable before another transition from 0 to 1 can take place.
n/a means that there is no predefined %MW mapping for this system variable.			

M262 System Functions

What’s in This Chapter

M262 Read Functions.....22

M262 Write Functions.....27

M262 User Functions.....29

Overview

This chapter describes the system functions included in the M262 PLCSystem library.

M262 Read Functions

Overview

This section describes the read functions included in the M262 PLCSystem library.

GetImmediateFastInput: Read Input of an Embedded Expert I/O

Function Description

This function returns the value of the input, which may be different from the logical value of that input. The value is read directly from the hardware at function call time. Only I0 to I3 can be accessed through this function.

Library and Namespace

Library name: **SE_PLCSystem**
Namespace: **SEC**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variables:

Input	Type	Comment
<i>Block</i>	INT	Not used.
<i>Input</i>	INT	Input Index to read from 0...3.

The following table describes the output variable:

Output	Type	Comment
<i>GetImmediateFastInput</i>	BOOL	Value of the input < <i>Input</i> > – FALSE/TRUE.

The following table describes the input/output variables:

Input/Output	Type	Comment
<i>Error</i>	BOOL	FALSE= operation is successful. TRUE= operation error, the function returns an invalid value.
<i>ErrID</i>	<i>IMMEDIATE_ERR_TYPE</i> , page 52	Operation error code when <i>Error</i> is TRUE.

GetRtc: Get Real Time Clock

Function Description

This function returns RTC time in seconds in UNIX format (time expired in seconds since January 1, 1970 at 00:00 UTC).

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the I/O variable:

Output	Type	Comment
<i>GetRtc</i>	DINT	RTC in seconds in UNIX format.

Example

The following example describes how to get the RTC value:

```
VAR
MyRTC : DINT := 0;
END_VAR
MyRTC := GetRtc();
```

IsFirstMastColdCycle: Indicate if this Cycle is the First MAST Cold Start Cycle

Function Description

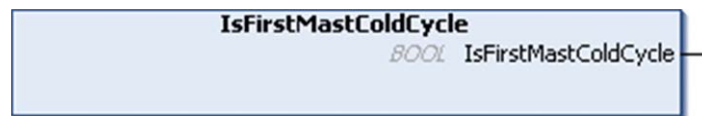
This function returns TRUE during the first MAST cycle after a cold start (first cycle after download or reset cold).

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The table describes the output variable:

Output	Type	Comment
<i>IsFirstMastColdCycle</i>	BOOL	TRUE during the first MAST task cycle after a cold start.

Example

Refer to the function *IsFirstMastCycle*, page 24.

IsFirstMastCycle: Indicate if this Cycle is the First MAST Cycle

Function Description

This function returns TRUE during the first MAST cycle after a start.

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

Output	Type	Comment
<i>IsFirstMastCycle</i>	BOOL	TRUE during the first MAST task cycle after a start.

Example

This example describes the three functions *IsFirstMastCycle*, *IsFirstMastColdCycle* and *IsFirstMastWarmCycle* used together.

Use this example in MAST task. Otherwise, it may run several times or possibly never (an additional task might be called several times or not called during 1 MAST task cycle):

```

VAR
MyIsFirstMastCycle : BOOL;
MyIsFirstMastWarmCycle : BOOL;
MyIsFirstMastColdCycle : BOOL;
END_VAR
MyIsFirstMastWarmCycle := IsFirstMastWarmCycle();
MyIsFirstMastColdCycle := IsFirstMastColdCycle();
MyIsFirstMastCycle := IsFirstMastCycle();
IF (MyIsFirstMastWarmCycle) THEN
(*This is the first Mast Cycle after a Warm Start: all
variables are set to their initialization values except the
Retain variables*)
(*=> initialize the needed variables so that your
application runs as expected in this case*)
END_IF;
IF (MyIsFirstMastColdCycle) THEN
(*This is the first Mast Cycle after a Cold Start: all
variables are set to their initialization values including
the Retain Variables*)
(*=> initialize the needed variables so that your
application runs as expected in this case*)
END_IF;
IF (MyIsFirstMastCycle) THEN
(*This is the first Mast Cycle after a Start, i.e. after a
Warm or Cold Start as well as STOP/RUN commands*)
(*=> initialize the needed variables so that your
application runs as expected in this case*)
END_IF;

```

IsFirstMastWarmCycle: Indicate if this Cycle is the First MAST Warm Start Cycle

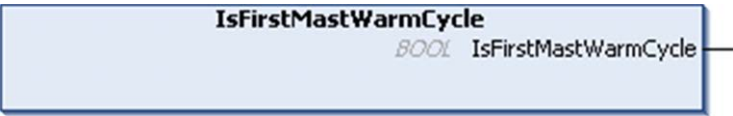
Function Description

This function returns TRUE during the first MAST cycle after a warm start.

Library and Namespace

Library name: **PLCSystemBase**
Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the output variable:

Output	Type	Comment
<i>IsFirstMastWarmCycle</i>	BOOL	TRUE during the first MAST task cycle after a warm start.

Example

Refer to the function *IsFirstMastCycle*, page 24.

GetExternalEventValue: Get Current Value of an External Event

Function Description

Use this function to get the value associated with an external event task.

NOTE: The function must be called from within an external event task.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variables Description

This table describes the input variables:

Inputs	Type	Comment
<i>pValue</i>	POINTER TO DINT	Address of the variable where the value is copied if the function returns <i>EXTEVT_VAL_OK</i> .

This table describes the output variables:

Outputs	Type	Comment
<i>GetExternalEventValue</i>	EXTEVT_VAL_RES	Returns one of the following values: <i>EXTEVT_VAL_OK</i>: Valid value <i>EXTEVT_VAL_FAILED</i>: Unable to obtain value <i>EXTEVT_VAL_NOT_IN_EXT_EVT_TASK</i>: Function was not called from within an external event task <i>EXTEVT_VAL_NOT_AVAILABLE</i>: No value available for this external task

M262 Write Functions

Overview

This section describes the write functions included in the M262 PLCSystem library.

PhysicalWriteFastOutputs: Write Fast Output of an Embedded Expert I/O

Function Description

This function writes a state to the Q0 to Q3 outputs at function call time.

Library and Namespace

Library name: **SE_PLCSys**tem

Namespace: **SEC**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variables:

Input	Type	Comment
<i>Q0Value</i>	BOOL	Requested value for the output 0.
<i>Q1Value</i>	BOOL	Requested value for the output 1.
<i>Q2Value</i>	BOOL	Requested value for the output 2.
<i>Q3Value</i>	BOOL	Requested value for the output 3.

The following table describes the output variable:

Output	Type	Comment
<i>PhysicalWriteFastOutputs</i>	WORD	Output value of the function.

NOTE: Only the first 4 bits of the output value are significant and used as a bit field to indicate if the output is written.

If the bit corresponding to the output is 1, the output is written successfully.

If the bit corresponding to the output is 0, the output is not written because it is already used by an expert function.

If the bit corresponding to the output is 1111 bin, all of the 4 outputs are written correctly.

If the bit corresponding to the output is 1110 bin, Q0 is not written.

NOTE: Values are applied at the beginning and end of a processing cycle. The function is applying a value within the cycle.

SetRTCDrift: Set Compensation Value to the RTC

Function Description

This function accelerates or slows down the frequency of the RTC to give control to the application for RTC compensation, depending on the operating environment (temperature, ...). The compensation value is given in seconds per week. It can be positive (accelerate) or negative (slow down).

NOTE: The *SetRTCDrift* function must be called only once. Each new call replaces the compensation value by the new one. The value is kept in the controller hardware while the RTC is powered by the main supply or by the battery. If both battery and power supply are removed, the RTC compensation value is not available.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variables Description

This table describes the input parameters:

Inputs	Type	Comment
<i>RtcDrift</i>	SINT (-29...+29)	Correction in seconds per week (-29 ... +29).

NOTE: The parameters *Day*, *Hour*, and *Minute* are used only to ensure backwards compatibility.

NOTE: If the value entered for *RtcDrift* exceeds the limit value, the controller firmware sets the value to its maximum value.

This table describes the output variable:

Output	Type	Comment
<i>SetRTCDrift</i>	<i>RTCSETDRIFT_ERROR</i> , page 53	Returns <i>RTC_OK</i> (00 hex) if command is correct otherwise returns the ID code of the detected error.

Example

In this example, the function is called only once during the first MAST task cycle. It accelerates the RTC by 4 seconds a week (18 seconds a month).

```

VAR
MyRTCDrift : SINT (-29...+29) := 0;
MyDay : DAY_OF_WEEK;
MyHour : HOUR;
MyMinute : MINUTE;
END_VAR
IF IsFirstMastCycle() THEN
MyRTCDrift := 4;
MyDay := 0;
MyHour := 0;
MyMinute := 0;
SetRTCDrift(MyRTCDrift, MyDay, MyHour, MyMinute);
END_IF

```

M262 User Functions

Overview

This section describes the following user functions:

Function	Description
<i>FB_GetFreeDiskSpace</i>	Retrieves the amount of free memory space of a memory medium in bytes. See <i>FB_GetFreeDiskSpace</i> : Gets the Free Memory Space Asynchronously, page 30
<i>FB_GetLabel</i>	Retrieves the label of a memory medium. See <i>FB_GetLabel</i> : Gets the Label of Memory Medium, page 31
<i>FB_GetTotalDiskSpace</i>	Retrieves the size of free memory space of a memory medium in bytes. See <i>FB_GetTotalDiskSpace</i> : Gets the Size of Memory Medium, page 32
<i>FB_CheckAllowedControllerMacAddr</i>	Check whether a specific MAC address is within the valid range for the controller. See <i>FB_CheckAllowedControllerMacAddr</i> : Check If MAC Address Allowed by Controller, page 33

Function	Description
<i>FB_ControlClone</i>	Enable or disable the controller cloning functionality. See <i>FB_ControlClone: Clone the Controller</i> , page 34
<i>DataFileCopy</i>	Copy memory data to a file and vice versa. See <i>DataFileCopy: Copy File Commands</i> , page 35
<i>ExecuteScript</i>	Run script commands. See <i>ExecuteScript: Run Script Commands</i> , page 37

FB_GetFreeDiskSpace: Gets the Free Memory Space Asynchronously

Function Block Description

This function block retrieves the amount of free memory space of a memory medium (user disk, system disk, SD card) in bytes. The name of the memory medium is transferred:

The free memory space of a remote device cannot be accessed. If a remote device is specified as parameter, then the function returns "-1".

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the input variables:

Input	Data type	Description
<i>i_xEnable</i>	BOOL	Activation entry, executes the operation when the value is TRUE .
<i>i_sVolumeName</i>	STRING[80]	Name of the device whose free memory space must be accessed: - System disk: <code>`/sys'</code></li> <li>User disk: <code>`/usr'</code> - SD Card: <code>`/sd0'</code>

This table describes the output variables:

Output	Data type	Description
<i>q_xDone</i>	BOOL	Set to TRUE when function block ended.
<i>q_xBusy</i>	BOOL	Set to TRUE when function block has been started and is still executing.
<i>q_OperationResult</i>	DINT	Result of the operation, a non zero value indicates an error.
<i>q_uliDiskSpace</i>	ULINT	Memory space in bytes.

FB_GetLabel: Gets the Label of Memory Medium

Function Block Description

This function block retrieves the label of a memory medium. If a device has no label, then an empty string is returned.

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the input variables:

Input	Data type	Description
<i>i_xEnable</i>	BOOL	Activation entry, executes the operation when the value is TRUE .
<i>i_sVolumeName</i>	STRING[80]	Name of the device whose free memory space must be accessed: - System disk: <code>`/sys'</code></li> <li>User disk: <code>`/usr'</code> - SD Card: <code>`/sd0'</code>

This table describes the output variables:

Output	Data type	Description
<i>q_xDone</i>	BOOL	Set to TRUE when function block ended.
<i>q_xBusy</i>	BOOL	Set to TRUE when function block has been started and is still executing.

Output	Data type	Description
<i>q_OperationResult</i>	DINT	Result of the operation, a non zero value indicates an error.
<i>q_sLabel</i>	STRING[11]	Label of the device.

FB_GetTotalDiskSpace: Gets the Size of Memory Medium

Function Block Description

This function block retrieves the amount of free memory space of a memory medium (user disk, system disk, SD card) in bytes.

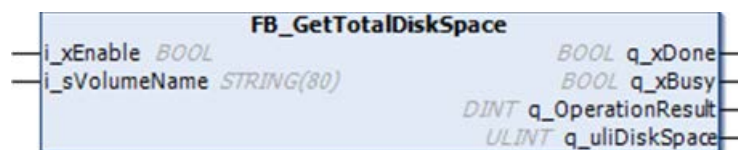
The size of a remote device cannot be accessed. If a remote device is specified as parameter, then the function returns "-1".

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the input variables:

Input	Data type	Description
<i>i_xEnable</i>	BOOL	Activation entry, executes the operation when the value is <code>TRUE</code> .
<i>i_sVolumeName</i>	STRING[80]	Name of the device whose memory size must be accessed: - System disk: <code>`/sys'</code></li> <li>User disk: <code>`/usr'</code> - SD Card: <code>`/sd0'</code>

This table describes the output variables:

Output	Data type	Description
<i>q_xDone</i>	BOOL	Set to <code>TRUE</code> when function block ended.
<i>q_xBusy</i>	BOOL	Set to <code>TRUE</code> when function block has been started and is still executing.

Output	Data type	Description
<i>q_OperationResult</i>	DINT	Result of the operation, a non zero value indicates an error.
<i>q_uliDiskSpace</i>	ULINT	Memory space in bytes.

FB_CheckAllowedControllerMacAddr: Check If MAC Address Allowed by Controller

Function Block Description

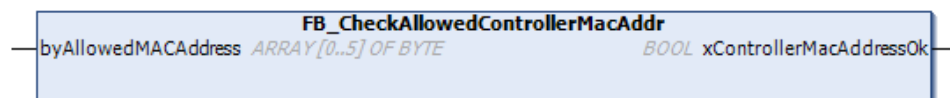
This function block checks whether a specified MAC address is within the range of MAC addresses allowed for the controller. The application only continues executing if the MAC address matches. Otherwise, the application stops and the controller goes to the HALT state and the system variable *i_wLastApplicationError*, page 15 is updated appropriately.

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variables:

Input	Type	Comment
<i>byAllowedMacAddress</i>	ARRAY[0...5] OF BYTE	MAC address to check [aa . bb . cc . dd . ee . ff]: • <i>i_byMACAddress</i>[0] = aa • ... • <i>i_byMACAddress</i>[5] = ff

The following table describes the output variables:

Output	Type	Comment
<i>xControllerMacAddressOk</i>	BOOL	TRUE = indicates that the MAC address is allowed for this controller.

FB_ControlClone: Clone the Controller

Function Block Description

Cloning is possible by SD card or **Controller Assistant**. When user rights are enabled and the View right **FrmUpdate** is denied for the **ExternalMedia** group, the cloning function is not allowed. In this case, the function block enables cloning functionality one time on the next controller power on.

NOTE: You can choose whether user rights are included in the clone on the **Clone Management** page of the Web server (see Modicon M262 Logic/Motion Controller, Programming Guide).

This table shows how to set the function block and the user rights:

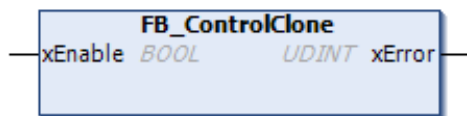
Function block setting	When user rights enabled	When user rights disabled
<i>xEnable</i> = 1	Cloning is allowed	Cloning is allowed
<i>xEnable</i> = 0	Cloning is not allowed	Cloning is not allowed

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variables:

Input	Type	Comment
<i>xEnable</i>	BOOL	If TRUE, enables the cloning functionality one time. If FALSE, disables the cloning functionality.

The following table describes the output variables:

Output	Type	Comment
<i>xError</i>	UDINT	A value of 0 indicates that no error was detected while executing the function block. A non-zero indicates that an error was detected.

DataFileCopy: Copy File Commands

Function Block Description

This function block copies memory data to a file and vice versa. The file is located either within the internal file system or an external file system (SD card).

The *DataFileCopy* function block can:

- Read data from a formatted file or
- Copy data from memory to a formatted file. For further information, refer to Non-Volatile Memory Organization (see Modicon M262 Logic/Motion Controller, Programming Guide).

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the input variables:

Input	Type	Comment
<i>xExecute</i>	BOOL	On rising edge, starts the function block execution. On falling edge, resets the outputs of the function block when any ongoing execution terminates. NOTE: With the falling edge, the function continues until it concludes its execution and updates its outputs. The outputs are retained for one cycle and reset.
<i>sFileName</i>	STRING	File name without extension (the extension <i>.DTA</i> is automatically added). Use only a...z, A...Z, 0...9 alphanumeric characters.
<i>xRead</i>	BOOL	TRUE: copy data from the file identified by <i>sFileName</i> to the internal memory of the controller. FALSE: copy data from the internal memory of the controller to the file identified by <i>sFileName</i> .
<i>xSecure</i>	BOOL	TRUE: The MAC address is always stored in the file. Only a controller with the same MAC address can read from the file. FALSE: Another controller with the same type of memory can read from the file.
<i>iLocation</i>	INT	0: the file location is <i>/usr/DTA</i> in internal file system. 1: the file location is <i>/usr/DTA</i> in external file system (SD card). NOTE: If the file does not already exist in the directory, the file is created.
<i>uiSize</i>	UINT	Indicates the size in bytes. Maximum is 65534 bytes. Only use addresses of variables conforming to IEC 61131-3 (variables, arrays, structures), for example: <code>Variable : int;</code> <code>uiSize := SIZEOF (Variable);</code>
<i>dwAdd</i>	DWORD	Indicates the address in the memory that the function will read from or write to. Only use addresses of variables conforming to IEC 61131-3 (variables, arrays, structures), for example: <code>Variable : int;</code> <code>dwAdd := ADR (Variable);</code>

WARNING

UNINTENDED EQUIPMENT OPERATION

Verify that the memory location is of the correct size and the file is of the correct type before copying the file to memory.

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

This table describes the output variables:

Output	Type	Comment
<i>xDone</i>	BOOL	TRUE = indicates that the action is successfully completed.
<i>xBusy</i>	BOOL	TRUE = indicates that the function block is running.
<i>xError</i>	BOOL	TRUE = indicates that an error is detected and the function block aborted the action.
<i>eError</i>	<i>DataFileCopyError</i> , page 46	Indicates the type of the data file copy detected error.

NOTE: If you modify data within the memory (variables, arrays, structures) used to write the file, a CRC integrity error results.

Example

This example describes how to copy file commands:

```
VAR
LocalArray : ARRAY [0..29] OF BYTE;
myFileName: STRING := 'exportfile';
EXEC_FLAG: BOOL;
DataFileCopy: DataFileCopy;
END_VAR
DataFileCopy(
  xExecute:= EXEC_FLAG,
  sFileName:= myFileName,
  xRead:= FALSE,
  xSecure:= FALSE,
  iLocation:= DFCL_INTERNAL,
  uiSize:= SIZEOF(LocalArray),
  dwAdd:= ADR(LocalArray),
  xDone=> ,
  xBusy=> ,
  xError=> ,
  eError=> );
```

ExecuteScript: Run Script Commands

Function Block Description

This function block can run the following SD card script commands:

- *Download*
- *Upload*
- *SetNodeName*
- *Delete*
- *Reboot*

For information on the required script file format, refer to Creating Script Files (see Modicon M262 Logic/Motion Controller, Programming Guide).

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

This table describes the input variables:

Input	Type	Comment
<i>xExecute</i>	BOOL	On detection of a rising edge, starts the function block execution. On detection of a falling edge, resets the outputs of the function block when any on-going execution terminates. NOTE: With the falling edge, the function continues until it concludes its execution and updates its outputs. The outputs are retained for one cycle and reset.
<i>sCmd</i>	STRING	SD card script command syntax. Simultaneous command executions are not allowed: if a command is being executed from another function block or from an SD card script then the function block queues the command and does not execute it immediately. NOTE: An SD card script executed from an SD card is considered as being executed until the SD card has been removed.

This table describes the output variables:

Output	Type	Comment
<i>xDone</i>	BOOL	TRUE indicates that the action is successfully completed.
<i>xBusy</i>	BOOL	TRUE indicates that the function block is running.
<i>xError</i>	BOOL	TRUE indicates error detection; the function block aborts the action.
<i>eError</i>	<i>ExecuteScriptError</i> , page 47	Indicates the type of the execute script detected error.

Example

This example describes how to execute an *Upload* script command:

```
VAR
EXEC_FLAG: BOOL;
ExecuteScript: ExecuteScript;
END_VAR
ExecuteScript(
xExecute:= EXEC_FLAG,
sCmd:= 'Upload "/usr/Syslog/*"',
xDone=> ,
xBusy=> ,
xError=> ,
eError=> );
```


M262 Library Data Types

What's in This Chapter

<i>PLC_RW</i> System Variables Data Types	39
DataFileCopy System Variables Data Types	46
ExecScript System Variables Data Types	47
<i>ETH_RW</i> System Variables Data Types	47
System Function Data Types	52

Overview

This chapter describes the data types of the M262 PLCSystem library.

There are 2 kinds of data types available:

- System variable data types are used by the system variables, page 12 of the M262 PLCSystem Library (*PLC_R*, *PLC_W*,...).
- System function data types are used by the read/write system functions, page 22 of the M262 PLCSystem Library.

PLC_RW System Variables Data Types

Overview

This section lists and describes the system variable data types included in the *PLC_R* and *PLC_W* structures.

PLC_R_APPLICATION_ERROR: Detected Application Error Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_APPLICATION_ERROR* enumeration data type contains the following values:

Enumerator	Value	Comment	What to do
<i>PLC_R_APP_ERR_UNKNOWN</i>	FFFF hex	Undefined error detected.	Contact your local Schneider Electric representative.
<i>PLC_R_APP_ERR_NOEXCEPTION</i>	0000 hex	No error detected.	–
<i>PLC_R_APP_ERR_WATCHDOG</i>	0010 hex	Task watchdog expired.	Verify your application, and correct if necessary. See System and Task Watchdogs (see Modicon M262 Logic/Motion Controller, Programming Guide). A reset is needed to enter RUNNING mode.
<i>PLC_R_APP_ERR_HARDWAREWATCHDOG</i>	0011 hex	The internal hardware watchdog has expired.	If the problem is reproducible, verify that there are no configured but disconnected communication ports. Otherwise, update the firmware. If the problem still persists, contact your Schneider Electric service representative
<i>PLC_R_APP_ERR_IO_CONFIG_ERROR</i>	0012 hex	Incorrect I/O configuration parameters detected.	Your application may be corrupted. To resolve this issue, reload the application from the PC. Otherwise, contact your local Schneider Electric representative.
<i>PLC_R_APP_ERR_UNRESOLVED_EXTREFS</i>	0018 hex	Undefined functions detected.	Remove the unresolved functions from the application.
<i>PLC_R_APP_ERR_IEC_TASK_CONFIG_ERROR</i>	0025 hex	Incorrect Task configuration parameters detected.	Your application may be corrupted. To resolve this issue, reload the application from the PC. Otherwise, contact your local Schneider Electric representative.
<i>PLC_R_APP_ERR_TARGET_MISMATCH</i>	0026 hex	Controller is not allowed to execute this IEC application.	The application could not be executed because it is not the correct target or controller type.
<i>PLC_R_APP_ERR_ILLEGAL_INSTRUCTION</i>	0050 hex	Undefined instruction detected.	Debug your application to resolve the problem.
<i>PLC_R_APP_ERR_ACCESS_VIOLATION</i>	0051 hex	Attempted access to reserved memory area.	Debug your application to resolve the problem.
<i>PLC_R_APP_ERR_DIVIDE_BY_ZERO</i>	0102 hex	Integer division by zero detected.	Debug your application to resolve the problem.
<i>PLC_R_APP_ERR_PROCESSORLOAD_WATCHDOG</i>	0105 hex	Processor overloaded by Application Tasks, and system watchdog has expired.	Reduce the application workload by improving the application architecture. Reduce event frequency.
<i>PLC_R_APP_ERR_DIVIDE_REAL_BY_ZERO</i>	0152 hex	Real division by zero detected.	Debug your application to resolve the problem.
<i>PLC_R_APP_ERR_EXPIO_EVENTS_COUNT_EXCEEDED</i>	4E20 hex	Too many events on expert I/Os are detected.	Reduce the number of external event tasks.
<i>PLC_R_APP_ERR_APPLICATION_VERSION_MISMATCH</i>	4E21 hex	Mismatch in the application version detected.	The application version in the controller does not match the version in EcoStruxure Machine Expert. Refer to Applications (see , Programming Guide).

PLC_R_BOOT_PROJECT_STATUS: Boot Project Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_BOOT_PROJECT_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>PLC_R_NO_BOOT_PROJECT</i>	0000 hex	Boot project does not exist in non-volatile memory.
<i>PLC_R_BOOT_PROJECT_CREATION_IN_PROGRESS</i>	0001 hex	Boot project is being created.
<i>PLC_R_DIFFERENT_BOOT_PROJECT</i>	0002 hex	Boot project in non-volatile memory is different from the project loaded in memory.
<i>PLC_R_VALID_BOOT_PROJECT</i>	FFFF hex	Boot project in non-volatile memory is the same as the project loaded in memory.

Enumerated Type Description

The *PLC_R_BOOT_PROJECT_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>PLC_R_NO_BOOT_PROJECT</i>	0000 hex	Boot project does not exist in non-volatile memory.
<i>PLC_R_BOOT_PROJECT_CREATION_IN_PROGRESS</i>	0001 hex	Boot project is being created.
<i>PLC_R_DIFFERENT_BOOT_PROJECT</i>	0002 hex	Boot project in non-volatile memory is different from the project loaded in memory.
<i>PLC_R_VALID_BOOT_PROJECT</i>	FFFF hex	Boot project in non-volatile memory is the same as the project loaded in memory.

PLC_R_IO_STATUS: I/O Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_IO_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>PLC_R_IO_OK</i>	FFFF hex	Inputs/Outputs are operational.
<i>PLC_R_IO_NO_INIT</i>	0001 hex	Inputs/Outputs are not initialized.
<i>PLC_R_IO_CONF_FAULT</i>	0002 hex	Incorrect I/O configuration parameters detected.
<i>PLC_R_IO_SHORTCUT_FAULT</i>	0003 hex	Inputs/Outputs short-circuit detected.
<i>PLC_R_IO_POWER_SUPPLY_FAULT</i>	0004 hex	Inputs/Outputs power supply error detected.

PLC_R_SDCARD_STATUS: SD Card Slot Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_SDCARD_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>NO_SDCARD</i>	0000 hex	No SD card detected in the slot or the slot is not connected.
<i>SDCARD_READONLY</i>	0001 hex	SD card is in read-only mode.
<i>SDCARD_READWRITE</i>	0002 hex	SD card is in read/write mode.
<i>SDCARD_ERROR</i>	0003 hex	Error detected in the SD card. More details are written to the file FwLog.txt.

PLC_R_STATUS: Controller Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>PLC_R_EMPTY</i>	0000 hex	Controller does not contain an application.
<i>PLC_R_STOPPED</i>	0001 hex	Controller is stopped.
<i>PLC_R_RUNNING</i>	0002 hex	Controller is running.
<i>PLC_R_HALT</i>	0004 hex	Controller is in a HALT state (see the controller state diagram in your controller programming guide (see Modicon M262 Logic/Motion Controller, Programming Guide)).
<i>PLC_R_BREAKPOINT</i>	0008 hex	Controller has paused at a breakpoint.

PLC_R_STOP_CAUSE: From RUN State to Other State Transition Cause Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_STOP_CAUSE* enumeration data type contains the following values:

Enumerator	Value	Comment	What to do
<i>PLC_R_STOP_REASON_UNKNOWN</i>	00 hex	Initial value or stop cause is indeterminable.	Contact your local Schneider Electric representative.
<i>PLC_R_STOP_REASON_HW_WATCHDOG</i>	01 hex	Stopped after hardware watchdog timeout.	Contact your local Schneider Electric representative.
<i>PLC_R_STOP_REASON_RESET</i>	02 hex	Stopped after reset.	See reset possibilities in Controller State Diagram (see Modicon M262 Logic/Motion Controller, Programming Guide).
<i>PLC_R_STOP_REASON_EXCEPTION</i>	03 hex	Stopped after exception.	Verify your application, and correct if necessary. See System and Task Watchdogs (see Modicon M262 Logic/Motion Controller, Programming Guide). A reset is needed to enter Run mode.
<i>PLC_R_STOP_REASON_USER</i>	04 hex	Stopped after a user request.	Refer to Stop Command in Commanding State Transitions (see Modicon M262 Logic/Motion Controller, Programming Guide).
<i>PLC_R_STOP_REASON_IECPROGRAM</i>	05 hex	Stopped after a program command request (for example: control command with parameter <i>PLC_W.q_wPLCControl:=PLC_W_COMMAND.PLC_W_STOP;</i>).	–
<i>PLC_R_STOP_REASON_DELETE</i>	06 hex	Stopped after a remove application command.	See the Applications tab of the Controller Device Editor (see Modicon M262 Logic/Motion Controller, Programming Guide).
<i>PLC_R_STOP_REASON_DEBUGGING</i>	07 hex	Stopped after entering debug mode.	–
<i>PLC_R_STOP_FROM_NETWORK_REQUEST</i>	0A hex	Stopped after a request from the network, USB key, or <i>PLC_W</i> command.	–
<i>PLC_R_STOP_FROM_INPUT</i>	0B hex	Stop required by a controller input.	–
<i>PLC_R_STOP_FROM_RUN_STOP_SWITCH</i>	0C hex	Stop required by the controller switch.	–
<i>PLC_R_STOP_REASON_RETAIN_MISMATCH</i>	0D hex	Stopped after an unsuccessful check context test during rebooting.	There are retained variables in non-volatile memory that do not exist in the executing application. Verify your application, correct if necessary, then reestablish the boot application.
<i>PLC_R_STOP_REASON_BOOT_APPLI_MISMATCH</i>	0E hex	Stopped after an unsuccessful compare between the boot application and the application that was in the memory before rebooting.	Create a valid boot application.
<i>PLC_R_STOP_REASON_POWERFAIL</i>	0F hex	Stopped after a power interruption.	Refer to reset possibilities in the Controller State Diagram (see Modicon M262 Logic/Motion Controller, Programming Guide).

For more information on the reasons why the controller has stopped, refer to the Controller State Description (see Modicon M262 Logic/Motion Controller, Programming Guide).

PLC_R_TERMINAL_PORT_STATUS: Programming Port Connection Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_TERMINAL_PORT_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>TERMINAL_NOT_CONNECTED</i>	00 hex	No PC is connected to the programming port.
<i>TERMINAL_CONNECTION_IN_PROGRESS</i>	01 hex	Connection is in progress.
<i>TERMINAL_CONNECTED</i>	02 hex	PC is connected to the programming port.
<i>TERMINAL_ERROR</i>	0F hex	Error detected during connection.

PLC_R_TM3_BUS_STATE: TM3 Bus Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_R_TM3_BUS_STATE* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>TM3_CONF_ERROR</i>	01 hex	Error detected due to mismatch in the physical configuration and the configuration in EcoStruxure Machine Expert.
<i>TM3_BUS_ERROR</i>	02hex	Error detected in <i>q_wIOBusErrPassiv</i> , page 65 in TM3 System library.
<i>TM3_OK</i>	03 hex	The physical configuration and the configuration in EcoStruxure Machine Expert match.
<i>TM3_POWER_SUPPLY_ERROR</i>	04 hex	Error detected in power supply.

PLC_W_COMMAND: Control Command Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *PLC_W_COMMAND* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>PLC_W_STOP</i>	0001 hex	Command to stop the controller.
<i>PLC_W_RUN</i>	0002 hex	Command to run the controller.
<i>PLC_W_RESET_COLD</i>	0004 hex	Command to initiate a Controller cold reset.
<i>PLC_W_RESET_WARM</i>	0008 hex	Command to initiate a Controller warm reset.

DataFileCopy System Variables Data Types

Overview

This section lists and describes the system variable data types included in the *DataFileCopy* structures.

DataFileCopyError: Detected Error Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *DataFileCopyError* enumeration data type contains the following values:

Enumerator	Value	Description
<i>ERR_NO_ERR</i>	00 hex	No error detected.
<i>ERR_FILE_NOT_FOUND</i>	01 hex	The file does not exist.
<i>ERR_FILE_ACCESS_REFUSED</i>	02 hex	The file cannot be opened.
<i>ERR_INCORRECT_SIZE</i>	03 hex	The request size is not the same as size read from file.
<i>ERR_CRC_ERR</i>	04 hex	The CRC is not correct and the file is assumed to be corrupted.
<i>ERR_INCORRECT_MAC</i>	05 hex	The controller attempting to read from the file does not have the same MAC address as that contained in the file.

DataFileCopyLocation: Location Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *DataFileCopyLocation* enumeration data type contains the following values:

Enumerator	Value	Description
<i>DFCL_INTERNAL</i>	00 hex	Data file with DTA extension is located in <i>/usr/Dta</i> directory.
<i>DFCL_EXTERNAL</i>	01 hex	Data file with DTA extension is located in <i>/sd0/usr/Dta</i> directory.
<i>DFCL_TBD</i>	02 hex	Not used.

ExecScript System Variables Data Types

Overview

This section lists and describes the system variable data types included in the *ExecScript* structures.

ExecuteScriptError: Detected Error Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ExecuteScriptError* enumeration data type contains the following values:

Enumerator	Value	Description
<i>CMD_OK</i>	00 hex	No error detected.
<i>ERR_CMD_UNKNOWN</i>	01 hex	The command is invalid.
<i>ERR_SD_CARD_MISSING</i>	02 hex	SD card is not present.
<i>ERR_SEE_FWLOG</i>	03 hex	There was an error detected during command execution, see <i>FwLog.txt</i> . For more information, refer to File Type (see Modicon M262 Logic/Motion Controller, Programming Guide).
<i>ERR_ONLY_ONE_COMMAND_ALLOWED</i>	04 hex	An attempt was made to execute several scripts simultaneously.
<i>CMD_BEING_EXECUTED</i>	05 hex	A script is already in progress.

ETH_RW System Variables Data Types

Overview

This section lists and describes the system variable data types included in the *ETH_R* and *ETH_W* structures.

***ETH_R_FRAME_PROTOCOL*: Frame Transmission Protocol Codes**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_FRAME_PROTOCOL* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>ETH_R_802_3</i>	00 hex	The protocol used for frame transmission is IEEE 802.3.
<i>ETH_R_ETHERNET_II</i>	01 hex	The protocol used for frame transmission is Ethernet II.

***ETH_R_IPFORWARDING*: IP Forwarding**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_IPFORWARDING* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>DISABLED</i>	00 hex	IP forwarding is disabled.
<i>ENABLED</i>	01 hex	IP forwarding is enabled.

***ETH_R_IP_MODE*: IP Address Source Codes**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_IP_MODE* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>ETH_R_STORED</i>	00 hex	Stored IP address is used.
<i>ETH_R_BOOTP</i>	01 hex	Bootstrap protocol (BOOTP) is used to get an IP address.

Enumerator	Value	Comment
<i>ETH_R_DHCP</i>	02 hex	DHCP protocol is used to get an IP address.
<i>ETH_DEFAULT_IP</i>	FF hex	Default IP address is used.

ETH_R_ITF_STRUCT: Ethernet Interface Parameters

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_ITF_STRUCT* enumeration data type contains the following values:

%MW	Var Name	Type	Comment
n/a	<i>i_byIPAddress</i>	ARRAY[0..3] OF BYTE	IP address of this Ethernet interface [aaa . bbb . ccc . ddd]: <i>i_byIPAddress</i>[0] = aaa ... <i>i_byIPAddress</i>[3] = ddd
n/a	<i>i_bySubNetMask</i>	ARRAY[0..3] OF BYTE	Subnet mask of this Ethernet interface [aaa . bbb . ccc . ddd]: <i>i_bySub-netMask</i>[0] = aaa ... <i>i_bySub-netMask</i>[3] = ddd
n/a	<i>i_byGateway</i>	ARRAY[0..3] OF BYTE	Gateway address of this Ethernet interface [aaa . bbb . ccc . ddd]: <i>i_byGateway</i>[0] = aaa ... <i>i_byGateway</i>[3] = ddd
n/a	<i>i_byMACAddress</i>	ARRAY[0..5] OF BYTE	MAC address of this Ethernet interface [aa . bb . cc . dd . ee . ff]: <i>i_byMACAddress</i>[0] = aa ... <i>i_byMACAddress</i>[5] = ff
n/a	<i>i_sDeviceName</i>	STRING(15)	Name used to get IP address from server.
n/a	<i>i_wlpMode</i>	<i>ETH_R_IP_MODE</i> , page 48	Method used to obtain an IP address.
n/a	<i>i_byFDRServerIPAddress</i>	ARRAY[0..3] OF BYTE	The IP address [aaa . bbb . ccc . ddd] of the DHCP or BootP server: <i>i_byFDRServerIPAddress</i>[0] = aaa ... <i>i_byFDRServerIPAddress</i>[3] = ddd Equals 0.0.0.0 if Stored IP or Default IP used.
n/a	<i>i_udiFramesTransmittedOK</i>	UDINT	Number of frames successfully transmitted. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiFramedReceivedOK</i>	UDINT	Number of frames successfully received. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_udiTransmitBufferErrors</i>	UDINT	Numbers of frames transmitted with detected errors. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .

%MW	Var Name	Type	Comment
n/a	<i>i_udiReceiveBufferErrors</i>	UDINT	Numbers of frames received with detected errors. Reset at Power ON or with reset command <i>ETH_W.q_wResetCounter</i> .
n/a	<i>i_wPortALinkStatus</i>	<i>ETH_R_PORT_LINK_STATUS</i> , page 51	Link of the Ethernet Port (0 = No Link, 1 = Link connected to another Ethernet device).
n/a	<i>i_wPortASpeed</i>	<i>ETH_R_PORT_SPEED</i> , page 51	Ethernet Port network speed (10Mb/s, 100Mb/s, or 1Gb/s).
n/a	<i>i_wPortADuplexStatus</i>	<i>ETH_R_PORT_DUPLEX_STATUS</i> , page 50	Ethernet Port duplex status (0 = Half or 1 = Full duplex).
n/a	<i>i_wPortAlpStatus</i>	<i>ETH_R_PORT_IP_STATUS</i> , page 50	Ethernet TCP/IP port stack status.
n/a means that there is no predefined %MW mapping for this system variable.			

ETH_R_PORT_DUPLEX_STATUS: Transmission Mode Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_PORT_DUPLEX_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>ETH_R_PORT_HALF_DUPLEX</i>	00 hex	Half duplex transmission mode is used.
<i>ETH_R_FULL_DUPLEX</i>	01 hex	Full duplex transmission mode is used.
<i>ETH_R_PORT_NA_DUPLEX</i>	03 hex	No duplex transmission mode is used.

ETH_R_PORT_IP_STATUS: Ethernet TCP/IP Port Status Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_PORT_IP_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>WAIT_FOR_PARAMS</i>	00 hex	Waiting for parameters.
<i>WAIT_FOR_CONF</i>	01 hex	Waiting for configuration.
<i>DATA_EXCHANGE</i>	02 hex	Ready for data exchange.

Enumerator	Value	Comment
<i>ETH_ERROR</i>	03 hex	Ethernet TCP/IP port error detected (cable disconnected, invalid configuration, and so on).
<i>DUPLICATE_IP</i>	04 hex	IP address already used by another device.

***ETH_R_PORT_LINK_STATUS*: Communication Link Status Codes**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_PORT_LINK_STATUS* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>ETH_R_LINK_DOWN</i>	00 hex	Communication link not available to another device.
<i>ETH_R_LINK_UP</i>	01 hex	Communication link available to another device.

***ETH_R_PORT_SPEED*: Communication Speed of the Ethernet Port Codes**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_PORT_SPEED* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>ETH_R_SPEED_NA</i>	0 dec	Network speed is 0 megabits per second.
<i>ETH_R_SPEED_10_MB</i>	10 dec	Network speed is 10 megabits per second.
<i>ETH_R_100_MB</i>	100 dec	Network speed is 100 megabits per second.
<i>ETH_R_1_GB</i>	1000 dec	Network speed is 1 gigabits per second.

***ETH_R_RUN_IDLE*: Ethernet/IP Run and Idle States Codes**

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The *ETH_R_RUN_IDLE* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>IDLE</i>	00 hex	EtherNet/IP connection is idle.
<i>RUN</i>	01 hex	EtherNet/IP connection is running.

System Function Data Types

Overview

This section describes the different system function data types of the M262 PLCSystem library.

IMMEDIATE_ERR_TYPE: GetImmediateFastInput Read Input of Embedded Expert I/O Codes

Library and Namespace

Library name: **PLCSystemBase**

Namespace: **PLCSystemBase**

Enumerated Type Description

The enumeration data type contains the following values:

Enumerator	Type	Comment
<i>IMMEDIATE_NO_ERROR</i>	Word	No errors detected.
<i>IMMEDIATE_UNKNOWN</i>	Word	The reference of <i>Immediate</i> function is incorrect or not configured.
<i>IMMEDIATE_UNKNOWN_PARAMETER</i>	Word	A parameter reference is incorrect.

RTCSETDRIFT_ERROR: SetRTCDrift Function Detected Error Codes

Enumerated Type Description

The *RTCSETDRIFT_ERROR* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>RTC_OK</i>	00 hex	RTC drift correctly configured.
<i>RTC_BAD_DAY</i>	01 hex	Not used.
<i>RTC_BAD_HOUR</i>	02 hex	Not used.
<i>RTC_BAD_MINUTE</i>	03 hex	Not used.
<i>RTC_BAD_DRIFT</i>	04 hex	RTC Drift parameter out of range.
<i>RTC_INTERNAL_ERROR</i>	05 hex	RTC Drift settings rejected on internal error detected.

SerialLine System

What’s in This Part

M262 Serial Line System Variables 55

Introduction

This part describes the M262 SerialLine Library.

M262 Serial Line System Variables

What's in This Chapter

<i>SERIAL_R</i> : Serial Line Diagnostic Variables.....	55
<i>SERIAL_W</i> : Serial Line Diagnostic Variables.....	56

Overview

This chapter describes the Serial Line system variables of the M262 PLCSystem library.

SERIAL_R: Serial Line Diagnostic Variables

Library and Namespace

Library name: **SerialLineSystem**

Namespace: **SEC_SLSYS**

Introduction

The *SERIAL_R* system variable (*SERIAL_R_ARRAY_TYPE* type) is an array of *SERIAL_R_STRUCT*.

Variable Structure

This table describes the parameters of the *SERIAL_R_STRUCT*:

Variable Name	Type	Comment
<i>i_udiFramesTransmittedOK</i>	UDINT	Indicates the number of frames transmitted since start by Serial Line bus.
<i>i_udiFramesReceivedOK</i>	UDINT	Indicates the number of frames received since start by Serial Line bus.
<i>i_udiRX_MessagesError</i>	UDINT	Indicates that the <i>RX Error Counter</i> register counts the number of frames received with errors detected for all types of frames.
<i>i_uiSlaveExceptionCount</i>	UINT	Indicates the number of bus exception errors detected.
<i>i_uiSlaveMsgCount</i>	UINT	Indicates the number of slave messages.
<i>i_uiSlaveNoRespCount</i>	UINT	Indicates the number of no responses from slaves.
<i>i_uiSlaveNakCount</i>	UINT	Indicates the number of slave NAK indications.
<i>i_uiSlaveBusyCount</i>	UINT	Indicates the number of slave busy indications.
<i>i_uiCharOverrunCount</i>	UINT	Indicates the number of character overruns encountered.

SERIAL_W: Serial Line Diagnostic Variables

Library and Namespace

Library name: **SerialLineSystem**
Namespace: **SEC_SLSYS**

Introduction

The *SERIAL_W* system variable (*SERIAL_W_ARRAY_TYPE* type) is an array of *SERIAL_W_STRUCT*.

Variable Structure

This table describes the parameters of the *SERIAL_W_STRUCT*:

Variable Name	Type	Comment
<i>q_wResetCounter</i>	WORD	Transition from 0 to 1 to reset the events counter. To reset the counter again, it is necessary to write 0 to this variable before another transition from 0 to 1 can take place.

TM3 System

What’s in This Part

TM3 System Variables	58
TM3 System Functions	59
TM3 System Data Types	64

Introduction

This part describes the TM3 System Library.

TM3 System Variables

What’s in This Chapter

TM3_MODULE_R[0...13]: TM3 Modules Read-Only System Variables..... 58

TM3_MODULE_R[0...13]: TM3 Modules Read-Only System Variables

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Introduction

The *TM3_MODULE_R* is an array of 14 *TM3_MODULE_R_STRUCT* type. Each element of the array returns diagnostic system variables for the corresponding TM3 expansion module.

For the Modicon M262 Logic/Motion Controller:

- *TM3_MODULE_R[0]* refers to the TM3 expansion module 0
- ...
- *TM3_MODULE_R[13]* refers to the TM3 expansion module 13

Variable Structure

The following table describes the parameters of the *TM3_MODULE_R[13]* system variable:

%MW	Var Name	Type	Comment
n/a	<i>i_wProductID</i>	WORD	TM3 expansion module ID.
n/a	<i>i_wModuleState</i>	<i>TM3_MODULE_STATE</i> , page 66	Describes the state of the TM3 module.

NOTE: **n/a** means that there is no predefined %MW mapping for this system variable.

TM3 System Functions

What's in This Chapter

<i>storetm3bus_w</i> : Change the TM3 Management Mode.....	59
<i>TM3_GetModuleBusStatus</i> : Get TM3 Module Bus Status.....	60
<i>TM3_GetModuleInternalStatus</i> : Get TM3 Module Internal Status	60
<i>TM3_SendDc2Cmd</i> : Send a DC2 Command to the TM3 bus	62

storetm3bus_w: Change the TM3 Management Mode

Function Description

This function allows the controller application to change the TM3 error management mode (passive/active) and to manually restart the TM3 bus.

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variable:

Input	Type	Comment
<i>pTm3Bus_W</i>	POINTER TO <i>TM3_BUS_W_STRUCT</i>	Pointer to the <i>TM3_BUS_W</i> , page 65 structure.

The following table describes the output variable:

Output	Type	Comment
<i>storetm3bus_w</i>	BOOL	<i>TRUE</i> = indicates that the action is successfully completed.

TM3_GetModuleBusStatus: Get TM3 Module Bus Status

Function Description

This function returns the bus status of the module. The index of the module is given as an input parameter.

Library and Namespace

Library name: **TM3System**
Namespace: **SEC_TM3Sys**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variable:

Input	Type	Comment
<i>ModuleIndex</i>	BYTE	Index of the expansion module (0 for the module closest to the controller, 1 for the second closest, and so on).

The following table describes the output variable:

Output	Type	Comment
<i>TM3_GetModuleBusStatus</i>	<i>TM3_ERR_CODE</i>	Returns <i>TM3_NO_ERR</i> (00 hex) if command is correct otherwise returns the ID code of the detected error.

TM3_GetModuleInternalStatus: Get TM3 Module Internal Status

Function Description

This function fills the *pStatusBuffer* with the status table of the module *ModuleIndex*.

This function selectively reads the I/O channel status of a TM3 analog or temperature module, indicated by *ModuleIndex*. The function block writes the status for each requested channel starting at the memory location pointed to by *pStatusBuffer*.

NOTE: This function block is intended to be used with analog and temperature I/O modules. To get status information for digital I/O modules, see *TM3_GetModuleBusStatus*, page 60.

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

Each analog/temperature I/O channel of the requested module requires one byte of memory. If there is not sufficient memory allocated to the buffer for the number of I/O module channel statuses requested, it is possible that the function will overwrite memory allocated for other purposes, or perhaps attempt to overwrite a restricted area of memory.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Ensure that *pStatusBuffer* is pointing to a memory area that has been sufficiently allocated for the number of channels to be read.

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

The following table describes the input variables:

Input	Type	Comment
<i>ModuleIndex</i>	BYTE	Index of the expansion module (0 for the module closest to the controller, 1 for the second closest, and so on)
<i>StatusOffset</i>	BYTE	Offset of the first status to be read in the status table.
<i>StatusSize</i>	BYTE	Number of bytes to be read in the status table.
<i>pStatusBuffer</i>	POINTER TO BYTE	Buffer containing the read status table.

The following table describes the output variable:

Output	Type	Comment
<i>TM3_GetModuleInternalStatus</i>	<i>TM3_ERR_CODE</i> , page 65	Returns <i>TM3_NO_ERR</i> (00 hex) if command is correct otherwise returns the ID code of the error. For the purposes of this function block, any returned value other than zero indicates that the module is not compatible with the status request, or that the module has other communication issues

Example

The following examples describe how to get the module internal status:

```
VAR
TM3AQ2_Channel_0_Output_Status: BYTE;
END_VAR
TM3AQ2 is on position 1
Status of channel 0 is at offset 0
We read 1 channel
TM3_GetModuleInternalStatus(1, 0, 1, ADR(TM3AQ2_Channel_0_
Output_Status));
status of channel 0 is in TM3AQ2_Channel_0_Output_Status
```

TM3AQ2 module (2 outputs)

Getting the status of first output QW0

- *StatusOffset* = 0 (0 inputs x 2)
- *StatusSize* = 1 (1 status to read)
- *pStatusBuffer* needs to be at least 1 byte

```
VAR
TM3AM6_Channels_1_2_Input_Status: ARRAY[1..2] OF BYTE;
END_VAR
TM3AM6 is on position 1
Status of channel 1 is at offset 9
We read 2 consecutive channels
TM3_GetModuleInternalStatus(1, 9, 2, ADR(TM3AM6_Channels_1_
2_Input_Status));
status of channel 1 is in TM3AM6_Channels_1_2_Input_Status
[1]
status of channel 2 is in TM3AM6_Channels_1_2_Input_Status
[2]
```

TM3AM6 module (4 inputs, 2 outputs)

Getting the status of input IW1 & IW2 (IW0 being the first one)

- *StatusOffset* = 9 (4 inputs x 2 + 1 to skip IW0 status)
- *StatusSize* = 2 (2 statuses to read)
- *pStatusBuffer* needs to be at least 2 bytes

TM3_SendDc2Cmd: Send a DC2 Command to the TM3 bus

Function Description

This function sends a module configuration command to the TM3 bus.

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation*, page 74.

I/O Variable Description

The following table describes the input variables:

Input	Type	Comment
<i>ModuleIndex</i>	BYTE	Index of the expansion module (0 for the module closest to the controller, 1 for the second closest, and so on)
<i>RWtype</i>	BYTE	Access type • 0: read • 1: write • 2: read/write
<i>ByteSize</i>	BYTE	Number of bytes to access (1 to 127)
<i>HeadAddr</i>	BYTE	Address within the block to be accessed (0 to 127)
<i>BlockNum</i>	BYTE	Block to be accessed (0 to 63)
<i>pInData</i>	POINTER TO BYTE	Pointer to the data buffer filled by the TM3 module. If no data expected: 0.
<i>pOutData</i>	POINTER TO BYTE	Pointer to the data buffer to be written to the TM3 module. If no data buffer to write: 0.

The following table describes the output variable:

Output	Type	Comment
<i>TM3_SendDc2Cmd</i>	<i>TM3_ERR_CODE</i>	Error code returned by the function. Refer to <i>TM3_NO_ERR</i> , page 65.

TM3 System Data Types

What's in This Chapter

<i>TM3_BUS_PARAM_ID</i> : TM3 Expansion Module Diagnostics	64
<i>TM3_BUS_W_IOBUSERRMOD</i> : TM3 bus error mode	64
<i>TM3_BUS_W_IOBUSINIT</i> : Reset the Bus Communication	64
<i>TM3_BUS_W</i> : TM3 Bus System Variables	65
<i>TM3_ERR_CODE</i> : TM3 Expansion Module Detected Error Codes.....	65
<i>TM3_MODULE_R_ARRAY_TYPE</i> : TM3 Expansion Module Read Array Type	66
<i>TM3_MODULE_STATE</i> : TM3 Expansion Module State Codes	66

TM3_BUS_PARAM_ID: TM3 Expansion Module Diagnostics

Enumerated Type Description

The *TM3_BUS_PARAM_ID* enumeration data type contains the following values:

Enumerator	Type	Value	Comment
<i>ID_TM3_MODULE_DIAG</i>	DWORD	10001 dec	-
<i>ID_TM3_BUS_W</i>	DWORD	10002 dec	-

TM3_BUS_W_IOBUSERRMOD: TM3 bus error mode

Enumerated Type Description

The *TM3_BUS_W_IOBUSERRMOD* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>IOBUS_ERR_ACTIVE</i>	00 hex	Active mode. The logic controller stops all I/O exchanges on the TM3 bus on detection of a permanent error. Refer to I/O Configuration General Description (see Modicon M262 Logic/Motion Controller, Programming Guide).
<i>IOBUS_ERR_PASSIVE</i>	01 hex	Passive mode. I/OSs exchanges continue on the TM3 bus even if an error is detected.

TM3_BUS_W_IOBUSINIT: Reset the Bus Communication

Enumerated Type Description

The *TM3_BUS_W_IOBUSINIT* enumeration data type contains the following values:

Enumerator	Type	Value
<i>CMD_INIT_OFF</i>	WORD	00 hex
<i>CMD_INIT_ON</i>	WORD	01 hex

TM3_BUS_W: TM3 Bus System Variables

Variable Structure

This table describes the parameters of the *TM3_BUS_W* system variable (*TM3_BUS_W_STRUCT* type):

Var Name	Type	Comment
<i>q_wIOBusErrPassiv</i>	<i>TM3_BUS_W_IOBUSERRMOD</i>	When set to <i>ERR_ACTIVE</i> (the default), bus errors detected on TM3 expansion modules stop I/O exchanges. When set to <i>ERR_PASSIVE</i> , passive I/O error handling is used: the controller attempts to continue data bus exchanges.
<i>q_wIOBusRestart</i>	<i>TM3_BUS_W_IOBUSINIT</i>	When set to 1, restarts the I/O expansion bus. This is only necessary when <i>q_wIOBusErrPassiv</i> is set to <i>ERR_ACTIVE</i> and at least one bit of <i>TM3_MODULE_R[i].i_wModuleState</i> is set to <i>TM3_BUS_ERROR</i> .

For more information, refer to I/O Configuration General Description (see Modicon M262 Logic/Motion Controller, Programming Guide).

TM3_ERR_CODE: TM3 Expansion Module Detected Error Codes

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Enumerated Type Description

The *TM3_ERR_CODE* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>TM3_NO_ERR</i>	00 hex	Last bus exchange with the expansion module was successful.
<i>TM3_ERR_FAILED</i>	01 hex	Error detected due to the last bus exchange with the expansion module was unsuccessful.
<i>TM3_ERR_PARAMETER</i>	02 hex	Parameter error detected in the last bus exchange with the module.
<i>TM3_ERR_COK</i>	03 hex	Temporary or permanent hardware error detected on one of the TM3 expansion modules.
<i>TM3_ERR_BUS</i>	04 hex	Bus error detected in the last bus exchange with the expansion module.

TM3_MODULE_R_ARRAY_TYPE: TM3 Expansion Module Read Array Type

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Description

The *TM3_MODULE_R_ARRAY_TYPE* is an array of 0...13 *TM3_MODULE_R_STRUCT*.

TM3_MODULE_STATE: TM3 Expansion Module State Codes

Library and Namespace

Library name: **TM3System**

Namespace: **SEC_TM3Sys**

Enumerated Type Description

The *TM3_MODULE_STATE* enumeration data type contains the following values:

Enumerator	Value	Comment
<i>TM3_EMPTY</i>	00 hex	No module.
<i>TM3_CONF_ERROR</i>	01 hex	Physical expansion module does not match with the one configured in EcoStruxure Machine Expert.
<i>TM3_BUS_ERROR</i>	02 hex	Bus error detected in the last exchange with the module.
<i>TM3_OK</i>	03 hex	Last bus exchange with this module was successful.
<i>TM3_POWER_SUPPLY_ERROR</i>	04_hex	Error on module external power supply.
<i>TM3_MISSING_OPT_MOD</i>	05 hex	Optional module is not physically present.

TMS System

What’s in This Part

TMS System Variables.....68

TMS System Data Types 70

Introduction

This part describes the TMS System Library.

TMS System Variables

What's in This Chapter

<i>TMS_BUS_DIAG_R</i> : TMS Bus Diagnostic Error Codes.....	68
<i>TMS_MODULE_DIAG_R</i> : TMS Expansion Module Diagnostic Error Codes	68

Overview

This chapter describes the system variables, page 12 included in the TMS System Library.

TMS_BUS_DIAG_R: TMS Bus Diagnostic Error Codes

Library and Namespace

Library name: **TMSSystem**

Namespace: **TMS**

Variable Structure

This table describes the parameters of the *TMS_BUS_DIAG_R* system variable (*STRUCT_TMS_BUS_DIAG* type):

Enumerator	Type	Comment
<i>ConfState</i>	UNIT	Describes the state of the TMS configuration: • 0: No configuration • 1: Configuration not valid (module mismatch) • 2: Configuration valid, but power booster not active • 3: Configuration valid and applied
<i>NbModules</i>	UNIT	Indicates the number of modules detected on the bus.

TMS_MODULE_DIAG_R: TMS Expansion Module Diagnostic Error Codes

Library and Namespace

Library name: **TMSSystem**

Namespace: **TMS**

Introduction

The *TMS_MODULE_DIAG_R* system variable (*TMS_MODULE_R_ARRAY_TYPE* type) is an array of *STRUCT_TMS_MODULE_DIAG*.

Variable Structure

This table describes the parameters of the *STRUCT_TMS_MODULE_DIAG*:

Enumerator	Type	Comment
<i>Name</i>	STRING(15)	Name of the TMS expansion module.
<i>MajorType</i>	WORD	Type of TMS expansion module.
<i>SubType</i>	WORD	Sub-type of TMS expansion module.
<i>Version</i>	STRING(15)	Firmware version of the TMS expansion module.
<i>ModuleState</i>	<i>TMS_MODULE_STATE</i> , page 70	State of the TMS expansion module.
<i>IpState</i>	<i>TMS_IP_STATE</i> , page 70	TCP IP accessibility of the TMS expansion module from the controller over COM_Bus .
<i>PixCmdState</i>	<i>TMS_PIXCMD_STATE</i> , page 71	State of the expansion module.

TMS System Data Types

What's in This Chapter

<i>TMS_IP_STATE</i> : TMS Expansion Module IP State	70
<i>TMS_MODULE_STATE</i> : TMS Expansion Module State Codes.....	70
<i>TMS_PIXCMD_STATE</i> : TMS Expansion Module PIXCMD State.....	71

TMS_IP_STATE: TMS Expansion Module IP State

Library and Namespace

Library name: **TMSSystem**

Namespace: **TMS**

Description

The *TMS_IP_STATE* enumeration data type contains the following values:

Name	Type	Value	Comment
<i>TMS_IP_PING_SUCCESS</i>	DWORD	0	IP interface is configured.
<i>TMS_IP_CONFIG_CMD_ERROR</i>	DWORD	1	Configuration send unsuccessful.
<i>TMS_IP_CONFIG_RESP_WAIT</i>	DWORD	2	Waiting for configuration response (transient).
<i>TMS_IP_CONFIG_RESP_ERROR</i>	DWORD	3	Configuration response error.
<i>TMS_IP_CONFIG_RESP_NONE</i>	DWORD	4	No configuration response.
<i>TMS_IP_CONFIG_SUCCESS</i>	DWORD	5	Configuration successful.
<i>TMS_IP_PING_CMD_ERROR</i>	DWORD	6	Ping command send unsuccessful.
<i>TMS_IP_PING_RESP_WAIT</i>	DWORD	7	Waiting for ping command response (transient).
<i>TMS_IP_PING_RESP_ERROR</i>	DWORD	8	Ping command response error.
<i>TMS_IP_PING_RESP_NONE</i>	DWORD	9	No ping command response.
<i>TMS_IP_NOT_CONFIGURED</i>	DWORD	10	IP interface not configured.

TMS_MODULE_STATE: TMS Expansion Module State Codes

Library and Namespace

Library name: **TMSSystem**

Namespace: **TMS**

Enumerated Type Description

The *TMS_MODULE_STATE* enumeration data type contains the following values:

Name	Type	Value	Comment
<i>TMS_MODULE_POWERED</i>	DWORD	0	Module is powered up.
<i>TMS_MODULE_INITIALIZED</i>	DWORD	1	Module is initialized and has been discovered.
<i>TMS_MODULE_CONFIGURED</i>	DWORD	2	Module is configured and is operating normally.
<i>TMS_MODULE_EXCHANGE_FAULT</i>	DWORD	3	Timeout for module detection expired.
<i>TMS_MODULE_ERROR</i>	DWORD	4	Module detected an error.
<i>TMS_MODULE_HEALTH_SEND_FAULT</i>	DWORD	5	Module health report send unsuccessful.
<i>TMS_MODULE_RCV_TIMEOUT</i>	DWORD	6	Module reception timeout.
<i>TMS_MODULE_RCV_MISC</i>	DWORD	7	Module reception error other than timeout.
<i>TMS_MODULE_RESP_ERR</i>	DWORD	8	Module response error.
<i>TMS_MODULE_DISCOVERY</i>	DWORD	9	Module discovery error.

***TMS_PIXCMD_STATE*: TMS Expansion Module PIXCMD State**

Library and Namespace

Library name: **TMSSystem**

Namespace: **TMS**

Description

The *TMS_PIXCMD_STATE* enumeration data type contains the following values:

Name	Type	Value	Comment
<i>TMS_PIXCMD_EXCHING</i>	DWORD	0	Exchange in progress.
<i>TMS_PIXCMD_CONFIG_NONE</i>	DWORD	1	No configuration.
<i>TMS_PIXCMD_CONFIG_CMD_ERROR</i>	DWORD	2	Configuration send unsuccessful.
<i>TMS_PIXCMD_CONFIG_RESP_WAIT</i>	DWORD	3	Waiting for configuration response (transient).
<i>TMS_PIXCMD_CONFIG_RESP_ERROR</i>	DWORD	4	Configuration response error.
<i>TMS_PIXCMD_CONFIG_ONLY</i>	DWORD	5	Configuration successful, no exchange occurred.
<i>TMS_PIXCMD_CONFIG_SUCCESS</i>	DWORD	6	Configuration successful (transient)
<i>TMS_PIXCMD_ENABLE_CMD_ERROR</i>	DWORD	7	Enable command error.
<i>TMS_PIXCMD_ENABLE_RESP_WAIT</i>	DWORD	8	Waiting for enable command response (transient).
<i>TMS_PIXCMD_ENABLE_RESP_ERROR</i>	DWORD	9	Enable command response error.

Name	Type	Value	Comment
<i>TMS_PIXCMD_EXCH_ERROR</i>	DWORD	10	Exchange error.
<i>TMS_PIXCMD_DISABLING</i>	DWORD	11	Exchange being disabled (transient).
<i>TMS_PIXCMD_DISABLED</i>	DWORD	12	Exchange disabled.

Appendices

What's in This Part

Function and Function Block Representation 74

Overview

This appendix extracts parts of the programming guide for technical understanding of the library documentation.

Function and Function Block Representation

What's in This Chapter

Differences Between a Function and a Function Block	74
How to Use a Function or a Function Block in IL Language	75
How to Use a Function or a Function Block in ST Language	77

Overview

Each function can be represented in the following languages:

- IL: Instruction List
- ST: Structured Text
- LD: Ladder Diagram
- FBD: Function Block Diagram
- CFC: Continuous Function Chart

This chapter provides functions and function blocks representation examples and explains how to use them for IL and ST languages.

Differences Between a Function and a Function Block

Function

A function:

- is a POU (Program Organization Unit) that returns one immediate result.
- is directly called with its name (not through an instance).
- has no persistent state from one call to the other.
- can be used as an operand in other expressions.

Examples: boolean operators (`AND`), calculations, conversion (`BYTE_TO_INT`)

Function Block

A function block:

- is a POU (Program Organization Unit) that returns one or more outputs.
- needs to be called by an instance (function block copy with dedicated name and variables).
- each instance has a persistent state (outputs and internal variables) from one call to the other from a function block or a program.

Examples: timers, counters

In the example, `Timer_ON` is an instance of the function block `TON`:

```

1  PROGRAM MyProgram_ST
2  VAR
3      Timer_ON: TON; // Function Block Instance
4      Timer_RunCd: BOOL;
5      Timer_PresetValue: TIME := T#5S;
6      Timer_Output: BOOL;
7      Timer_ElapsedTime: TIME;
8  END_VAR

1  Timer_ON(
2      IN:=Timer_RunCd,
3      PT:=Timer_PresetValue,
4      Q=>Timer_Output,
5      ET=>Timer_ElapsedTime);

```

How to Use a Function or a Function Block in IL Language

General Information

This part explains how to implement a function and a function block in IL language.

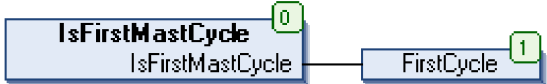
Functions `IsFirstMastCycle` and `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in IL Language

This procedure describes how to insert a function in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POU's (see EcoStruxure Machine Expert, Programming Guide).
2	Create the variables that the function requires.
3	If the function has 1 or more inputs, start loading the first input using LD instruction.
4	Insert a new line below and: - type the name of the function in the operator column (left field), or - use the Input Assistant to select the function (select Insert Box in the contextual menu).
5	If the function has more than 1 input and when Input Assistant is used, the necessary number of lines is automatically created with ??? in the fields on the right. Replace the ??? with the appropriate value or variable that corresponds to the order of inputs.
6	Insert a new line to store the result of the function into the appropriate variable: type ST instruction in the operator column (left field) and the variable name in the field on the right.

To illustrate the procedure, consider the Functions `IsFirstMastCycle` (without input parameter) and `SetRTCDrift` (with input parameters) graphically presented below:

Function	Graphical Representation
without input parameter: IsFirstMastCycle	
with input parameters: SetRTCDrift	

In IL language, the function name is used directly in the operator column:

Function	Representation in POU IL Editor		
IL example of a function without input parameter: IsFirstMastCycle	1 PROGRAM MyProgram_IL 2 VAR 3 FirstCycle: BOOL; 4 END_VAR 5 1 IsFirstMastCycle ST FirstCycle <tr><td> IL example of a function with input parameters: SetRTCDrift </td><td> 1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag </td></tr>	IL example of a function with input parameters: SetRTCDrift	1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag
IL example of a function with input parameters: SetRTCDrift	1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag		

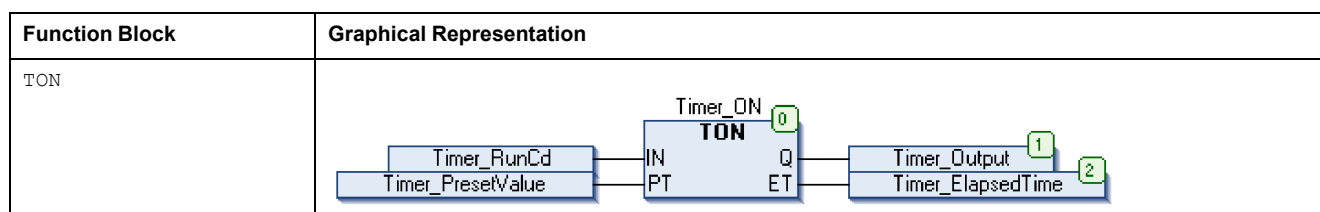
Using a Function Block in IL Language

This procedure describes how to insert a function block in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs (see EcoStruxure Machine Expert, Programming Guide).
2	Create the variables that the function block requires, including the instance name.
3	Function Blocks are called using a CAL instruction: Use the Input Assistant to select the function block (right-click and select Insert Box in the contextual menu).

Step	Action
	- Automatically, the <code>CAL</code> instruction and the necessary I/O are created. Each parameter (I/O) is an instruction: - Values to inputs are set by ":=". - Values to outputs are set by "=>".
4	In the <code>CAL</code> right-side field, replace ??? with the instance name.
5	Replace other ??? with an appropriate variable or immediate value.

To illustrate the procedure, consider this example with the `TON` Function Block graphically presented below:



In IL language, the function block name is used directly in the operator column:

Function Block	Representation in POU IL Editor
TON	<pre> 1 PROGRAM MyProgram_IL 2 VAR 3 Timer_ON: TON; // Function Block instance declaration 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 </pre> <pre> 1 CAL Timer_ON(2 IN:= Timer_RunCd, 3 PT:= Timer_PresetValue, 4 Q=> Timer_Output, 5 ET=> Timer_ElapsedTime) </pre>

How to Use a Function or a Function Block in ST Language

General Information

This part explains how to implement a Function and a Function Block in ST language.

Function `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in ST Language

This procedure describes how to insert a function in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs (see EcoStruxure Machine Expert, Programming Guide).
2	Create the variables that the function requires.
3	Use the general syntax in the POU ST Editor for the ST language of a function. The general syntax is: <code>FunctionResult:= FunctionName (VarInput1, VarInput2, .. VarInputx);</code>

To illustrate the procedure, consider the function `SetRTCDrift` graphically presented below:

Function	Graphical Representation
SetRTCDrift	

The ST language of this function is the following:

Function	Representation in POU ST Editor
SetRTCDrift	<pre> PROGRAM MyProgram_ST VAR myDrift: SINT(-29..29) := 5; myDay: DAY_OF_WEEK := SUNDAY; myHour: HOUR := 12; myMinute: MINUTE; myRTCAdjust: RTCDRIFT_ERROR; END_VAR myRTCAdjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute); </pre>

Using a Function Block in ST Language

This procedure describes how to insert a function block in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs (see EcoStruxure Machine Expert, Programming Guide).
2	Create the input and output variables and the instance required for the function block: - Input variables are the input parameters required by the function block - Output variables receive the value returned by the function block
3	Use the general syntax in the POU ST Editor for the ST language of a Function Block. The general syntax is: <code>FunctionBlock_InstanceName (Input1:=VarInput1, Input2:=VarInput2, ... Ouput1=>VarOutput1, Ouput2=>VarOutput2, ...);</code>

To illustrate the procedure, consider this example with the `TON` function block graphically presented below:

Function Block	Graphical Representation
TON	

This table shows examples of a function block call in ST language:

Function Block	Representation in POU ST Editor
TON	<pre>1 PROGRAM MyProgram_ST 2 VAR 3 Timer_ON: TON; // Function Block Instance 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 1 Timer_ON(2 IN:=Timer_RunCd, 3 PT:=Timer_PresetValue, 4 Q=>Timer_Output, 5 ET=>Timer_ElapsedTime);</pre>

Glossary

A

%:

According to the IEC standard, % is a prefix that identifies internal memory addresses in the logic controller to store the value of program variables, constants, I/O, and so on.

application:

A program including configuration data, symbols, and documentation.

ARRAY:

The systematic arrangement of data objects of a single type in the form of a table defined in logic controller memory. The syntax is as follows: ARRAY

[<dimension>] OF <Type>

Example 1: ARRAY [1..2] OF BOOL is a 1-dimensional table with 2 elements of type BOOL.

Example 2: ARRAY [1..10, 1..20] OF INT is a 2-dimensional table with 10 x 20 elements of type INT.

B

BOOL:

(*boolean*) A basic data type in computing. A BOOL variable can have one of these values: 0 (FALSE), 1 (TRUE). A bit that is extracted from a word is of type BOOL; for example, %MW10.4 is a fifth bit of memory word number 10.

Boot application:

(*boot application*) The binary file that contains the application. Usually, it is stored in the controller and allows the controller to boot on the application that the user has generated.

BOOTP:

(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address (and possibly other data) from a server. The client identifies itself to the server using the client MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its pre-configured IP address. BOOTP was originally used as a method that enabled diskless hosts to be remotely booted over a network. The BOOTP process assigns an infinite lease of an IP address. The BOOTP service utilizes UDP ports 67 and 68.

byte:

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C

CAN:

(*controller area network*) A protocol (ISO 11898) for serial bus networks, designed for the interconnection of smart devices (from multiple manufacturers) in smart systems and for real-time industrial applications. Originally developed for use in automobiles, CAN is now used in a variety of industrial automation control environments.

CFC:

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

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.

control network:

A network containing logic controllers, SCADA systems, PCs, HMI, switches, ...

Two kinds of topologies are supported:

- flat: all modules and devices in this network belong to same subnet.
- 2 levels: the network is split into an operation network and an inter-controller network.

These two networks can be physically independent, but are generally linked by a routing device.

CRC:

(*cyclical redundancy check*) A method used to determine the validity of a communication transmission. The transmission contains a bit field that constitutes a checksum. The message is used to calculate the checksum by the transmitter according to the content of the message. Receiving nodes, then recalculate the field in the same manner. Any discrepancy in the value of the 2 CRC calculations indicates that the transmitted message and the received message are different.

D**DHCP:**

(*dynamic host configuration protocol*) An advanced extension of BOOTP. DHCP is more advanced, but both DHCP and BOOTP are common. (DHCP can handle BOOTP client requests.)

DWORD:

(*double word*) Encoded in 32-bit format.

E**EtherNet/IP:**

(*Ethernet industrial protocol*) An open communications protocol for manufacturing automation solutions in industrial systems. EtherNet/IP is in a family of networks that implement the common industrial protocol at its upper layers. The supporting organization (ODVA) specifies EtherNet/IP to accomplish global adaptability and media independence.

Ethernet:

A physical and data link layer technology for LANs, also known as IEEE 802.3.

F

FB:

(function block) A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

firmware:

Represents the BIOS, data parameters, and programming instructions that constitute the operating system on a controller. The firmware is stored in non-volatile memory within the controller.

function block diagram:

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

function block:

A programming unit that has 1 or more inputs and returns 1 or more outputs. FBs are called through an instance (function block copy with dedicated name and variables) and each instance has a persistent state (outputs and internal variables) from 1 call to the other.

Examples: timers, counters

function:

A programming unit that has 1 input and returns 1 immediate result. However, unlike FBs, it is directly called with its name (as opposed to through an instance), has no persistent state from one call to the next and can be used as an operand in other programming expressions.

Examples: boolean (AND) operators, calculations, conversions (BYTE_TO_INT)

G

GVL:

(global variable list) Manages global variables within an EcoStruxure Machine Expert project.

H

hex:

(hexadecimal)

I

I/O:

(input/output)

ID:

(identifier/identification)

IEC 61131-3:

Part 3 of a 3-part IEC standard for industrial automation equipment. IEC 61131-3 is concerned with controller programming languages and defines 2 graphical and 2 textual programming language standards. The graphical programming languages are ladder diagram and function block diagram. The textual programming languages include structured text and instruction list.

IEC:

(international electrotechnical commission) A non-profit and non-governmental international standards organization that prepares and publishes international standards for electrical, electronic, and related technologies.

IEEE 802.3:

A collection of IEEE standards defining the physical layer, and the media access control sublayer of the data link layer, of wired Ethernet.

IL:

(instruction list) A program written in the 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 (refer to IEC 61131-3).

INT:

(integer) A whole number encoded in 16 bits.

IP:

(Internet protocol) Part of the TCP/IP protocol family that tracks the Internet addresses of devices, routes outgoing messages, and recognizes incoming messages.

L**LD:**

(ladder diagram) 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 (refer to IEC 61131-3).

legacy projects:

Application projects that were created with SoMachine, SoMachine Motion, or a previous version of EcoStruxure Machine Expert.

LWORD:

(long word) A data type encoded in a 64-bit format.

M**MAC address:**

(media access control address) A unique 48-bit number associated with a specific piece of hardware. The MAC address is programmed into each network card or device when it is manufactured.

MAST:

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

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

Modbus:

The protocol that allows communications between many devices connected to the same network.

%MW:

According to the IEC standard, %MW represents a memory word register (for example, a language object of type memory word).

N**network:**

A system of interconnected devices that share a common data path and protocol for communications.

NVM:

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

P**PLC:**

(*programmable logic controller*) An industrial computer used to automate manufacturing, industrial, and other electromechanical processes. PLCs are different from common computers in that they are designed to have multiple input and output arrays and adhere to more robust specifications for shock, vibration, temperature, and electrical interference among other things.

POU:

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

program:

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

protocol:

A convention or standard definition that controls or enables the connection, communication, and data transfer between 2 computing system and devices.

R**run:**

A command that causes the controller to scan the application program, read the physical inputs, and write to the physical outputs according to solution of the logic of the program.

S**STOP:**

A command that causes the controller to stop running an application program.

string:

A variable that is a series of ASCII characters.

ST:

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

system variable:

A variable that provides controller data and diagnostic information and allows sending commands to the controller.

T**task:**

A group of sections and subroutines, executed cyclically or periodically for the MAST task or periodically for the FAST task.

A task possesses a level of priority and is linked to inputs and outputs of the controller. These I/O are refreshed in relation to the task.

A controller can have several tasks.

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**UDINT:**

(*unsigned double integer*) Encoded in 32 bits.

UINT:

(*unsigned integer*) Encoded in 16 bits.

unlocated variable:

A variable that does not have an address (refer to *located variable*).

V**variable:**

A memory unit that is addressed and modified by a program.

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.

WORD:

A type encoded in a 16-bit format.

Index

C

cycle

GetExternalEventValue	26
IsFirstMastColdCycle	24
IsFirstMastCycle	24
IsFirstMastWarmCycle	26

D

data types

DataFileCopyError	46
DataFileCopyLocation	46
ETH_R_FRAME_PROTOCOL	48
ETH_R_IP_MODE	48
ETH_R_IPFORWARDING	48
ETH_R_ITF_STRUCT	49
ETH_R_PORT_DUPLEX_STATUS	50
ETH_R_PORT_IP_STATUS	50
ETH_R_PORT_LINK_STATUS	51
ETH_R_PORT_SPEED	51
ETH_R_RUN_IDLE	51
ExecuteScriptError	47
IMMEDIATE_ERR_TYPE	52
PLC_R_APPLICATION_ERROR	39
PLC_R_BOOT_PROJECT_STATUS	41
PLC_R_IO_STATUS	41
PLC_R_SDCARD_STATUS	42
PLC_R_STATUS	42
PLC_R_STOP_CAUSE	43
PLC_R_TERMINAL_PORT_STATUS	45
PLC_R_TM3_BUS_STATE	45
PLC_W_COMMAND	45
TM3_BUS_PARAM_ID	64
TM3_BUS_W_IOBUSERRMOD	64
TM3_BUS_W_IOBUSINIT	64
TM3_ERR_CODE	65
TM3_MODULE_R_ARRAY_TYPE	66
TM3_MODULE_STATE	66
TMS_IP_STATE	70
TMS_MODULE_STATE	70
TMS_PIXCMD_STATE	71

Data Types

RTCSETDRIFT_ERROR	53

DataFileCopy

copying data to or from a file	35

DataFileCopyError

data types	46

DataFileCopyLocation

data type	46

E

embedded I/O

GetImmediateFastInput	22
PhysicalWriteFastOutputs	27

ETH_R

system variable	19

ETH_R_FRAME_PROTOCOL

data type	48

ETH_R_IP_MODE

data type	48

ETH_R_IPFORWARDING

data type	48

ETH_R_ITF_STRUCT

data type	49
ETH_R_PORT_DUPLEX_STATUS	50
data type	50
ETH_R_PORT_IP_STATUS	50
data type	50
ETH_R_PORT_LINK_STATUS	51
data type	51
ETH_R_PORT_SPEED	51
data type	51
ETH_R_RUN_IDLE	51
data type	51
ETH_W	21
system variable	21
ExecuteScript	37
running script commands	37
ExecuteScriptError	47
data type	47

F

FB_CheckAllowedControllerMacAddr

function block	33

FB_ControlClone

function block	34

FB_GetFreeDiskSpace

function block	30

FB_GetLabel

function block	31

FB_GetTotalDiskSpace

function block	32

file copy commands

DataFileCopy	35

function blocks

DataFileCopy	35
ExecuteScript	37
FB_CheckAllowedControllerMacAddr	33
FB_ControlClone	34
FB_GetFreeDiskSpace	30
FB_GetLabel	31
FB_GetTotalDiskSpace	32

functions

differences between a function and a function block	74
how to use a function or a function block in IL language	75
how to use a function or a function block in ST language	77

G

GetExternalEventValue

get current value of an external event	26

GetImmediateFastInput

getting the value of a fast input	22

GetRtc

getting real time clock (RTC) value	23

I

IMMEDIATE_ERR_TYPE

data type	52

IsFirstMastColdCycle

first cold start cycle	24

IsFirstMastCycle

first mast cycle	24

IsFirstMastWarmCycle

first warm start cycle	26

M

M262 PLCSystem	
GetExternalEventValue	26
GetImmediateFastInput	22
PhysicalWriteFastOutputs	27
SetRTCDrift	28

P

PhysicalWriteFastOutputs	
writing output of an embedded expert I/O	27
PLC_R	
system variable	14
PLC_R_APPLICATION_ERROR	
data type	39
PLC_R_BOOT_PROJECT_STATUS	
data type	41
PLC_R_IO_STATUS	
data type	41
PLC_R_SDCARD_STATUS	
data type	42
PLC_R_STATUS	
data type	42
PLC_R_STOP_CAUSE	
data type	43
PLC_R_TERMINAL_PORT_STATUS	
data type	45
PLC_R_TM3_BUS_STATE	
data type	45
PLC_W	
system variable	18
PLC_W_COMMAND	
data type	45
PLCSystemBase	
DataFileCopy	35
DataFileCopyError	46
DataFileCopyLocation	46
ETH_R	19
ETH_R_FRAME_PROTOCOL	48
ETH_R_IP_MODE	48
ETH_R_IPFORWARDING	48
ETH_R_ITF_STRUCT	49
ETH_R_PORT_DUPLEX_STATUS	50
ETH_R_PORT_IP_STATUS	50
ETH_R_PORT_LINK_STATUS	51
ETH_R_PORT_SPEED	51
ETH_R_RUN_IDLE	51
ETH_W	21
ExecuteScript	37
ExecuteScriptError	47
FB_CheckAllowedControllerMacAddr	33
FB_ControlClone	34
FB_GetFreeDiskSpace	30
FB_GetLabel	31
FB_GetTotalDiskSpace	32
GetRtc	23
IMMEDIATE_ERR_TYPE	52
IsFirstMastColdCycle	24
IsFirstMastCycle	24
IsFirstMastWarmCycle	26
PLC_R	14
PLC_R_APPLICATION_ERROR	39
PLC_R_BOOT_PROJECT_STATUS	41
PLC_R_IO_STATUS	41
PLC_R_SDCARD_STATUS	42
PLC_R_STATUS	42
PLC_R_STOP_CAUSE	43
PLC_R_TERMINAL_PORT_STATUS	45

PLC_R_TM3_BUS_STATE	45
PLC_W	18
PLC_W_COMMAND	45

R

real time clock	
GetRtc	23
SetRTCDrift	28
RTC	
GetRtc	23
SetRTCDrift	28
RTCSETDRIFT_ERROR	
Data Types	53

S

script commands	
ExecuteScript	37
SERIAL_R	
system variable	55
SERIAL_W	
system variable	56
SerialLineSystem	
SERIAL_R	55
SERIAL_W	56
SetRTCDrift	
accelerating or slowing the RTC frequency	28
storetm3bus_w	
getting the bus status of a TM3 module	59
system variables	
definition	12
ETH_R	19
ETH_W	21
PLC_R	14
PLC_W	18
SERIAL_R	55
SERIAL_W	56
TM3_BUS_W	65
TM3_MODULE_R	58
TMS_BUS_DIAG_R	68
TMS_MODULE_DIAG_R	68
using	13

T

TM3 module bus status	
TM3_GetModuleBusStatus	60
TM3 module internal status	
TM3_GetModuleInternalStatus	60
TM3 SendDc2Cmd	
TM3_SendDc2Cmd	62
TM3_BUS_PARAM_ID	
data type	64
TM3_BUS_W	
system variable	65
TM3_BUS_W_IOBUSERRMOD	
data type	64
TM3_BUS_W_IOBUSINIT	
data type	64
TM3_ERR_CODE	
data type	65
TM3_GetModuleBusStatus	
getting the bus status of a TM3 module	60
TM3_GetModuleInternalStatus	
getting the internal status of a TM3 module	60
TM3_MODULE_R	

system variable.....	58
TM3_MODULE_R_ARRAY_TYPE	
data type	66
TM3_MODULE_STATE	
data type	66
TM3_SendDc2Cmd	
getting the bus status of a TM3 module.....	62
TM3System	
storetm3bus_w	59
TM3_BUS_PARAM_ID	64
TM3_BUS_W	65
TM3_BUS_W_IOBUSERRMOD	64
TM3_BUS_W_IOBUSINIT	64
TM3_ERR_CODE.....	65
TM3_GetModuleBusStatus.....	60
TM3_MODULE_R.....	58
TM3_MODULE_R_ARRAY_TYPE	66
TM3_MODULE_STATE.....	66
TM3_SendDc2Cmd	62
TMS_BUS_DIAG_R	
system variable.....	68
TMS_IP_STATE	
data type	70
TMS_MODULE_DIAG_R	
system variable.....	68
TMS_MODULE_STATE	
data type	70
TMS_PIXCMD_STATE	
data types	71
TMSSystem	
TMS_BUS_DIAG_R.....	68
TMS_IP_STATE.....	70
TMS_MODULE_DIAG_R	68
TMS_MODULE_STATE	70
TMS_PIXCMD_STATE.....	71

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2022 Schneider Electric. All rights reserved.

EIO0000003667.04

Modicon M262

CommonMotionPcrt

Library Guide

EIO0000004658.00
02/2022

Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2022 Schneider Electric. All Rights Reserved.

Table of Contents

Safety Information.....	5
Qualification of Personnel	5
Proper Use.....	6
Before You Begin.....	6
Start-up and Test	7
Operation and Adjustments	7
About the Book.....	8
Presentation of the Library	12
General Information	12
Enumerations	13
<i>ET_Result</i> - General Information.....	13
Function Blocks	14
<i>FB_PersistPosition</i>	14
<i>FB_PersistPosition</i> - General Information	14
Functions.....	17
<i>FC_EtResultToString</i> - General Information	17
Structures.....	18
<i>ST_PersistPositionData</i> - General Information	18
Index	19

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a "Danger" or "Warning" safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

Qualification of Personnel

A qualified person is one who has the following qualifications:

- Skills and knowledge related to the construction and operation of electrical equipment and the installation.
- Knowledge and experience in industrial control programming.
- Received safety-related training to recognize and avoid the hazards involved.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical,

electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Proper Use

- This product is a library to be used together with the control systems and servo amplifiers intended solely for the purposes as described in the present documentation as applied in the industrial sector.
- Always observe the applicable safety-related instructions, the specified conditions, and the technical data.
- Perform a risk evaluation concerning the specific use before using the product. Take protective measures according to the result.
- Since the product is used as a part of an overall system, you must ensure the safety of the personnel by means of the design of this overall system (for example, machine design).
- Any other use is not intended and may be hazardous.

Before You Begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

 **WARNING**

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

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

- This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.
- Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.
- In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.
- Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-

operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

Start-up and Test

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check are made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

Operation and Adjustments

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

Document Scope

This document describes the functionalities contained in the CommonMotionPcrt library.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.2.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Product Related Information

 WARNING LOSS OF CONTROL • The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart. • Separate or redundant control paths must be provided for critical control functions. • System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link. • Observe all accident prevention regulations and local safety guidelines.¹ • Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service. Failure to follow these instructions can result in death, serious injury, or equipment damage.

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POUs found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF PROGRAM ORGANIZATION UNITS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the Program Organization Units (POUs) are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POUs.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Always evaluate the return values when using POUs of a library.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update your application program as required, paying particular attention to I/O address adjustments, whenever you modify the hardware configuration.

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

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.

⚠ WARNING

UNINTENDED MOVEMENT OF THE AXIS

- Ensure the proper functioning of the functional safety equipment before commissioning.
- Ensure that you can stop axis movements at any time using functional safety equipment (limit switch, emergency stop) before and during commissioning.

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

⚠ WARNING

UNINTENDED MOVEMENT OF THE SLAVE AXIS

Deactivate the POU that instructs the slave, or disconnect the connection from the master, if the slave axis stops independently from the master.

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

Motion function blocks, except for the Homing function blocks, can only be activated after the mechanical position reference has been established. This is especially important after the start-up of the Sercos motion bus.

⚠ WARNING

INCORRECT HOMING REFERENCE TO MECHANICAL SYSTEM

Ensure that a valid mechanical position reference exists by performing commissioning tests for all operating modes.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Presentation of the Library

General Information

Description

The library CommonMotionPcrt provides enumerations, structures, and functions that allow you to restore the home position of an axis during phasing up of Sercos without having to go through a homing procedure.

Characteristics of This Library

The following table indicates the characteristics of the library:

Characteristics	Value
Library title	CommonMotionPcrt
Company	Schneider Electric
Category	System
Component	M262 (motion control)
Default namespace	CM
Language model attribute	Qualified-access-only (see EcoStruxure Machine Expert, Functions and Libraries User Guide)
Forward compatible library	No

NOTE: For this library, qualified-access-only is set. Therefore, the POUs, data structures, enumerations, and constants have to be accessed using the namespace of the library. The default namespace of the library is CM.

Enumerations

ET_Result - General Information

Overview

Type:	Enumeration type
Available as of:	V2.2.0.0

Description

This enumeration contains the return values on the status of the execution of the function block *FB_PersistPosition*, page 14.

Enumeration Elements

Name	Value (INT)	Description
<i>OK</i>	0	Function block <i>FB_PersistPosition</i> executed successfully.
<i>UnexpectedReturnValue</i>	1	Indeterminable return value from system Contact your Schneider Electric service representative
<i>AxisInvalid</i>	2	No axis is specified for the input <i>i_ifAxis</i> , or the specified axis does not provide the functionality required by the function block <i>FB_PersistPosition</i> . Connect the axis for which the function block is to be executed to the input <i>i_ifAxis</i> . Connect an axis that provide the functionality required by the function block.
<i>NotConsistantRtpData</i>	3	The read Realtime Process (RTP) data is inconsistent. Home the axis. Retry to execute the function block <i>FB_PersistPosition</i> .
<i>AxisNotHomed</i>	4	The axis specified via the input <i>i_ifAxis</i> is not homed. Home the axis. Retry to execute the function block <i>FB_PersistPosition</i> .
<i>ChecksumInvalid</i>	5	The checksum of the axis position value and the encoder increment value in the structure <i>ST_PersistPositionData</i> , page 18 is invalid. Set the data in the structure <i>ST_PersistPositionData</i> to 0. Home the axis. Retry to execute the function block <i>FB_PersistPosition</i> . Contact your Schneider Electric service representative if the condition persists.
<i>InvalidParameters</i>	6	Contact your Schneider Electric service representative.
<i>DeviceNotSupported</i>	7	An attempt was made to execute the function block for a device that is not supported. Refer to Requirements and Limitations, page 15 for details.

Function Blocks

FB_PersistPosition

FB_PersistPosition - General Information

Overview

Type:	Function block
Available as of:	V2.2.0.0
Inherits from:	-
Implements:	-

Task

This function block allows you to restore the home position of an axis during phasing up of Sercos without having to go through a homing procedure.

Description

This function block allows you to restore the home position of an axis based on an axis position value and the corresponding encoder increment value previously stored in persistent memory. This restoration is performed during phasing up of Sercos so that the axis does not need to be homed again.

An axis is homed if the relationship between an axis position X (*LXM32.AxisIrPosition*) and its mechanical position Y (as determined by the increments of its motor encoder) is known. This relationship is established by homing.

For an axis already homed, the relationship for a given axis position value can be calculated based on the known axis position value and the encoder increment value if these two values originate from the same Realtime Process (RTP) cycle. The relationship remains fixed as long as the values X and Y do not exceed the limits of their position ranges (modulo axis for X , encoder overflow or underflow for Y). The fixed relationship also persists beyond a power removal and rebooting of the controller under the condition that the axis has not been moved, for example by external forces, while power was off. This means that the homing relationship between a position value X_1 of a homed axis and an encoder increment value Y_1 can be used to determine the relationship between a read encoder increment value Y_2 and the corresponding axis position value X_2 .

The function block stores the homing relationship of an axis via the tuple of the values of the axis position and the encoder increment in a persistent variable. After phasing up of Sercos, the home position of the axis can be set without a homing procedure, based on the stored relationship, the read encoder increment value and the calculated axis position value.

To use the function block, create a structure of type *ST_PersistPositionData* in an area of the persistent memory that is used exclusively for this function block. This structure is used for the axis position value and the corresponding motor increment value. Create a function block *FB_PersistPosition*. Connect the structure *ST_PersistPositionData* to the input/output *iq_stHomedData* of the function block. Connect an axis to the input *i_ifAxis* of the function block.

Home the axis via the function block *MC_HOME*. Call the function block by setting the value at its input *i_xEnable* to TRUE.

If the home position is to be restored after a controller bootup, ensure that the axis is set to the original values before the function block is called in the application.

If the axis is correctly homed, each subsequent call of the function block *FB_PersistPosition* stores the axis position value and the corresponding motor

encoder increment value in the structure *ST_PersistPositionData*. The value at the output *q_xPositionStored* is TRUE and the value at the output *q_etResult* is OK.

If the specified axis is not homed when the function block is called:

- The value at the output *q_xPositionStored* is FALSE.
- The value at the output *q_etResult* is *AxisNotHomed*.
- The value at the output *q_xError* is FALSE, that is, no error detected.

If the specified axis is invalid (for example, axis of a drive that is not supported):

- The value at the output *q_xPositionStored* is FALSE.
- The value at the output *q_etResult* is *AxisInvalid*.
- The value at the output *q_xError* is TRUE, that is, error detected.

Once the function block *FB_PersistPosition* has saved the data in *ST_PersistPositionData* and Sercos is phased up again (for example, after a controller reboot), calling an enabled function block *FB_PersistPosition* reads the encoder increment value. The stored homing relationship is used to calculate the home position of the axis corresponding to the encoder increment value read. This home position is set to the axis and the value at the output *q_xPositionRestored* is set to TRUE. No homing procedure is required.

In addition to the axis position value and the encoder increment value, the function block stores the checksum of these two values for each cycle. When the home position is restored during phasing up of Sercos, the function block verifies the consistency of the two values on the basis of the checksum. If a checksum error is detected during the restoration of the home position (*q_xError* is TRUE), the output *q_etResult* is set to *ChecksumInvalid*. In this case, no values are stored or restored. Set the data in the structure *ST_PersistPositionData* to 0. Home the axis. Retry to execute the function block *FB_PersistPosition*. Contact your Schneider Electric service representative if the condition persists.

The checksum verification is not performed if the values in the structure *ST_PersistPositionData* are zero.

Requirements and Limitations

The following requirements and limitations apply to the use of the function block *FB_PersistPosition*:

- The function block supports LXM32S drives with the approved motors with multiturn motor encoders with an encoder working range of 4096 turns with 131072 increments per turn (refer to the user guide of the drive for details).
- The function block does not support machine encoders.
- While the function block is being executed, the values of axis parameters must not be modified.
- The following parameters of the LXM32S drive require the following settings:
 - *SimAbsolutePos*: OFF (value 0, no simulation)
 - *ShiftEncWorkRang*: OFF (value 0, no shifting of encoder working range)
 - *InvertDirOfMove*: OFF (value 0, no inversion of direction of movement)
- If the motor is replaced, or in the event of any other type of uncoupling of the motor from the mechanical system (for example, during maintenance), the axis needs to be re-homed before the function block can be used again.
- The memory area for the persistent data must be used exclusively for this function block.

Interface

Input	Data type	Description
<i>i_xEnable</i>	BOOL	Starts (value TRUE) or terminates (value FALSE) the execution of the function block.
<i>i_ifAxis</i>	<i>MOIN.IF_Axis</i>	Reference to the axis for which the function block is to be executed.

Output	Data type	Description
<i>q_xPositionStored</i>	BOOL	Indicates whether storing of the home position was successful (value TRUE) or not (value FALSE).
<i>q_xPositionRestored</i>	BOOL	Indicates whether restoring of the home position was successful (value TRUE) or not (value FALSE).
<i>q_xError</i>	BOOL	Indicates whether the last execution of the function block was successful (value FALSE = no error detected) or not (value TRUE = error detected).
<i>q_etResult</i>	<i>ET_Result</i> , page 13	Provides information on the execution of the function block.

Input/Output	Data type	Description
<i>iq_stHomedData</i>	<i>ST_PersistPositionData</i> , page 18	Structure for the axis position value and the corresponding motor increment value.

Functions

FC_EtResultToString - General Information

Overview

Type:	Function
Available as of:	V2.2.0.0

Task

Converts an enumeration element of type `ET_Result` to a variable of type `STRING`.

Description

Using the function *FC_EtResultToString*, you can convert an enumeration element of type `ET_Result` to a variable of type `STRING`.

Interface

Input	Data type	Description
<i>i_etResult</i>	<code>ET_Result</code>	Enumeration element to be converted to a string.

Output	Data type	Description
<i>q_xError</i>	<code>BOOL</code>	If this output is set to <code>TRUE</code> , an error was detected. Refer to <i>ET_Result</i> .
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 13.

Return Value

Data type	Description
<code>STRING(80)</code>	String converted from the <i>ET_Result</i> element used as input value.

Structures

ST_PersistPositionData - General Information

Overview

Type:	Data structure
Available as of:	V2.2.0.0
Inherits from:	-

Description

This structure contains the axis position value and the corresponding encoder increment value used by the function block *FB_PersistPosition*, page 14 to restore the home position of an axis.

Structure Elements

Variable	Data type	Description
<i>lrAxisPosition</i>	LREAL	Axis position value
<i>dwEncoderIncrements</i>	DWORD	Encoder increment value
<i>dwChecksum</i>	DWORD	Checksum of axis position value and corresponding encoder increment value

Index

C

CommonMotionPcrt

ET_Result	13
FB_PersistPosition.....	14
FC_EtResultToString.....	17
ST_PersistPositionData.....	18

E

ET_Result	13

F

FB_PersistPosition	14
FC_EtResultToString	17

S

ST_PersistPositionData	18

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2022 – Schneider Electric. All rights reserved.

EIO0000004671.00

Modicon M262

Logic/Motion Controller

Encoder Library Guide

2/2019



The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

© 2019 Schneider Electric. All rights reserved.

Table of Contents



	Safety Information	5
	About the Book	7
Chapter 1	Encoder Modes Principles	13
	Incremental Mode Principle Description	14
	SSI Mode Principle Description	17
Chapter 2	M262 Logic/Motion Controller Encoder Function Blocks	19
	FB_Encoder_M262: Enable and Monitor the Encoder	20
	FB_EncoderPreset_M262: Preset the Encoder	23
	FB_EncoderCapture_M262: Capture the Encoder Value	25
	FB_EncoderReadScalingParam_M262: Read the Scaling Parameter	27
Chapter 3	M262 Logic/Motion Controller Library Data Types	29
	ET_ENC_CAP_EDGE_M262: Encoder Capture Codes	30
	ET_ENC_ERROR_M262: Encoder Error Codes	31
	ET_ENC_INPUT_M262: Encoder Input Codes	32
	ET_ENC_PRESET_MODE_M262: Encoder Preset Mode Codes	33
Appendices		35
Appendix A	Function and Function Block Representation	37
	Differences Between a Function and a Function Block	38
	How to Use a Function or a Function Block in IL Language	39
	How to Use a Function or a Function Block in ST Language	43
Glossary		45
Index		47

Safety Information



Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in death** or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in death** or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result** in minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book



At a Glance

Document Scope

This document will acquaint you with the encoder functions and variables offered within the M262 Logic/Motion Controller. The M262 Logic/Motion Controller Encoder library contains functions and variables to get information from and send commands to the encoder system.

This document describes the data type functions and variables of the M262 Logic/Motion Controller Encoder library.

The following knowledge is required:

- Basic information on the functionality, structure, and configuration of the M262 Logic/Motion Controller.
- Programming in the FBD, LD, ST, IL, or CFC language.
- system variables (global variables).

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V1.2.

Related Documents

Title of Documentation	Reference Number
EcoStruxure Machine Expert - Programming Guide	<i>EIO0000002854 (ENG);</i> <i>EIO0000002855 (FRE);</i> <i>EIO0000002856 (GER);</i> <i>EIO0000002858 (SPA);</i> <i>EIO0000002857 (ITA);</i> <i>EIO0000002859 (CHS)</i>
Modicon M262 Logic/Motion Controller - Hardware Guide	<i>EIO0000003659 (ENG);</i> <i>EIO0000003660 (FRE);</i> <i>EIO0000003661 (GER);</i> <i>EIO0000003662 (SPA);</i> <i>EIO0000003663 (ITA);</i> <i>EIO0000003664 (CHS);</i> <i>EIO0000003665 (POR);</i> <i>EIO0000003666 (TUR)</i>
Modicon M262 Logic/Motion Controller - Programming Guide	<i>EIO0000003651 (ENG);</i> <i>EIO0000003652 (FRE);</i> <i>EIO0000003653 (GER);</i> <i>EIO0000003654 (SPA);</i> <i>EIO0000003655 (ITA);</i> <i>EIO0000003656 (CHS);</i> <i>EIO0000003657 (POR);</i> <i>EIO0000003658 (TUR)</i>

You can download these technical publications and other technical information from our website at <https://www.se.com/ww/en/download/> .

WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Chapter 1

Encoder Modes Principles

Overview

This chapter describes how to use an encoder in incremental mode or in SSI (Synchronous Serial Interface) mode.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Incremental Mode Principle Description	14
SSI Mode Principle Description	17

Incremental Mode Principle Description

Overview

This section describes the use of the incremental mode to connect incremental encoders.

Principle

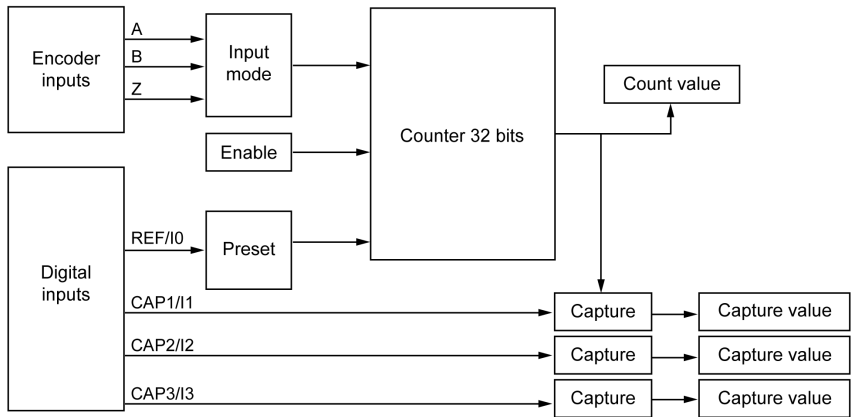
The incremental mode behaves like a standard up/down counter, using pulses and counting these pulses.

Positions must be preset and counting must be initialized to implement and manage the incremental mode.

The counter value can be stored in the capture register by configuring an external event.

Principle Diagram

The following diagram provides an overview of the encoder in incremental mode:



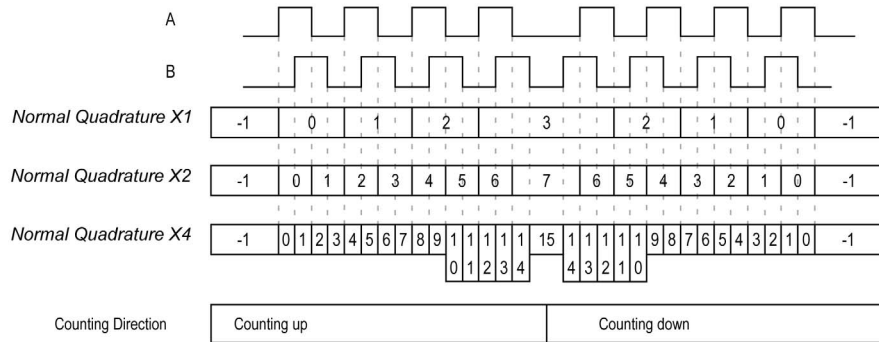
Axis Types

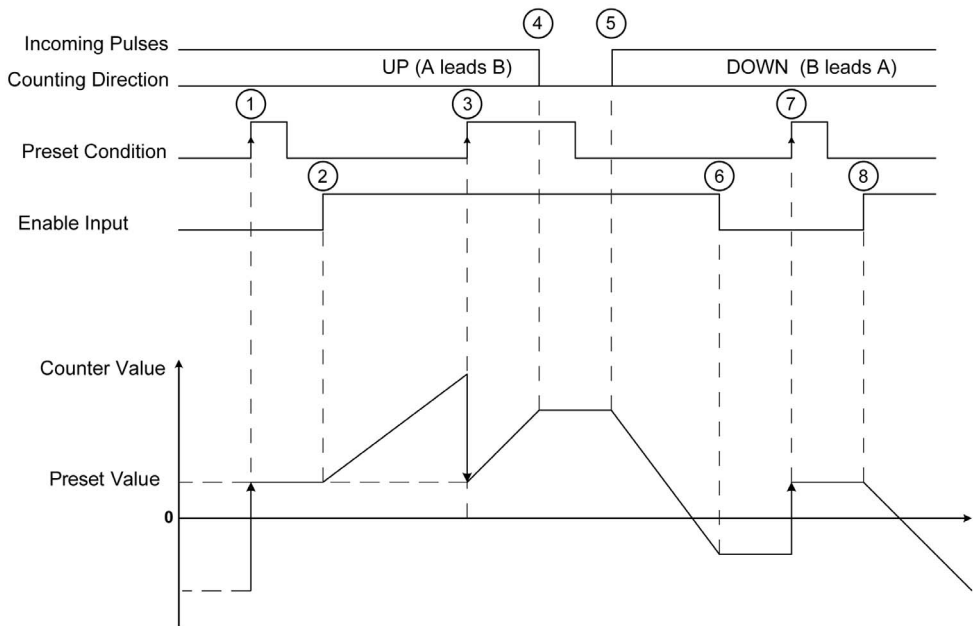
The following table presents the two available axis types and corresponding counting modes:

Axis Type	Comment
Linear	This mode acts as a finite counter.
Rotary	This mode acts as an infinite counter.

Principle Diagram

The input mode in incremental mode is always quadrature:





Stage	Action
1	On the rising edge of Preset condition, the counter value is set to the preset value and the counter is activated.
2	When the Enable condition = 1, the counter starts to increment when the counting direction is up.
3	The rising edge on the Preset condition loads the Preset value.
4	When the incoming pulses stop, the counter maintains its value.
5	When the Enable condition = 1, the counter starts to decrements when the counting direction is down.
6	When the Enable condition = 0, the counter ignores the pulses applied to the counting inputs A/B.
7	The rising edge on the Preset condition loads the preset value.
8	When the Enable condition = 1, the counter starts to decrements when the counting direction is down.

NOTE: Enable and Preset conditions depend on the configuration. These are described in the Enable ([see page 20](#)) and Preset ([see page 23](#)) function.

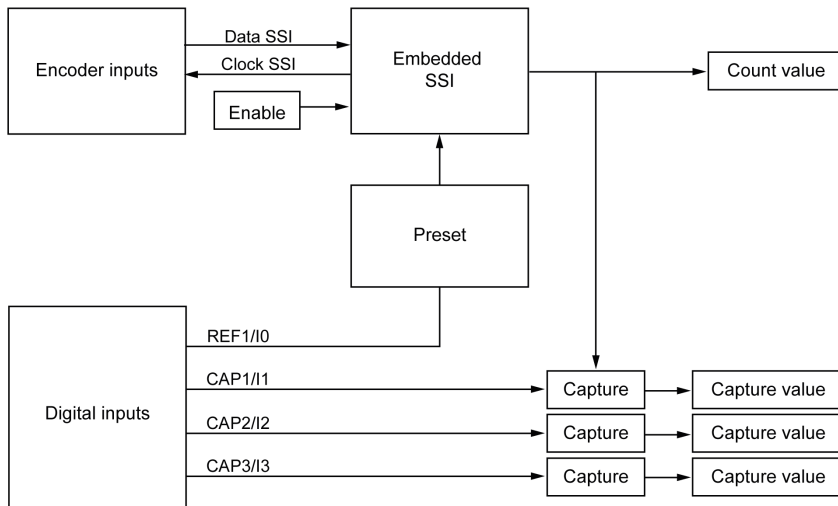
SSI Mode Principle Description

General

The SSI (Synchronous Serial Interface) mode allows the connection of an absolute encoder. The position of the absolute encoder is read by an SSI link.

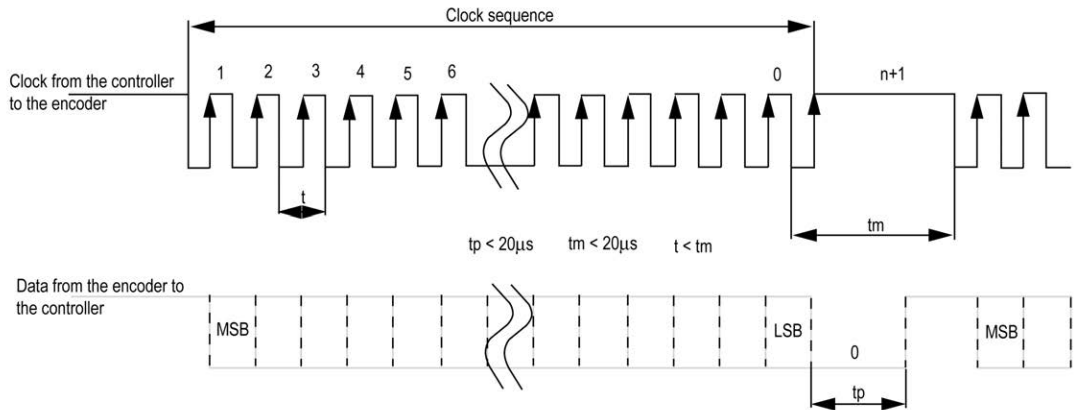
Principle Diagram

The following diagram provides an overview of the encoder in SSI mode:



Principle Diagram

The figure below represents an SSI frame:



Data Information

The data content can be configured to adjust the information from the absolute encoder:

Parameter	Range	Comment
Transmission speed	100 kHz, or 250 kHz, or 500 kHz	–
Number of bits per frame	8...64 bits	Length of frame = implicit number of header bits (0 to 4) + number of data bits (8 to 32) + number of status bits (0 to 4) + number of parity bit (0 or 1).
Number of data bits	8...32 bits	The least significant bits (8...32) indicate resolution per turn and the most significant bits (0...24) indicate the number of turns.
Number of data data/turn	8...16 bits	–
Number of status bits	0...4 bits	–
Parity	None Odd Even	–
Resolution reduction	0...17 bits	This parameter allows to filter data. The least significant bits are ignored.
Binary coding	Binary Gray	Binary or gray code.

Chapter 2

M262 Logic/Motion Controller Encoder Function Blocks

Overview

This chapter describes the function blocks included in the M262 Encoder Library. Adding an encoder adds automatically the Encoder Library to your controller.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
FB_Encoder_M262: Enable and Monitor the Encoder	20
FB_EncoderPreset_M262: Preset the Encoder	23
FB_EncoderCapture_M262: Capture the Encoder Value	25
FB_EncoderReadScalingParam_M262: Read the Scaling Parameter	27

FB_Encoder_M262: Enable and Monitor the Encoder

Function Block Description

This function block is used to enable and monitor the encoder, in incremental or SSI mode. You can only use one instance of this function block which is called once. Use the cyclic calls to refresh the values.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation* (see page 37).

I/O Variable Description

This table describes the input variables:

Input	Type	Default	Comments
ENC_REF_M262	ENC_REF_M262	–	Reference of the encoder instance.
xEnable	BOOL	FALSE	TRUE enables the function block. On a rising edge, the values of the following scaling parameters are taken into account: • udiScaling_NbOfIncs • udiScaling_NbOfUnits • udiScaling_IncPerTurn If you modify these values, trigger a rising edge on xEnable to take them into account.
udiScaling_NbOfIncs	UDINT	0	0 indicates that the scaling is disabled. The value in user units diCurrentValue_Unit equals the value in pulses diCurrentValue. > 0 indicates that the scaling is enabled. The value in user units is calculated from the value in pulses diCurrentValue, such as: $\text{diCurrentValue\_Unit} = \text{diCurrentValue} \times (\text{udiScalingNbOfUnits} / \text{udiScalingNbOfIncs}).$
udiScaling_NbOfUnits	UDINT	0	0 indicates that the scaling is disabled. The value in user units diCurrentValue_Unit equals the value in pulses diCurrentValue. If no scaling, then udiScalingNbOfUnits=udiScalingNbOfIncs. > 0 indicates that the scaling is enabled. The value in user units is calculated from the value in pulses diCurrentValue, such as: $\text{diCurrentValue\_Unit} = \text{diCurrentValue} \times (\text{udiScalingNbOfUnits} / \text{udiScalingNbOfIncs}).$
udiScaling_IncPerTurn	UDINT	0	When equal to 0, the axis type has a counter mode of linear. The counting range is: - 2 147 483 648...2 147 483 647. If the number of increments is > 0, the axis type has a counter mode of rotary. udiScaling_IncPerTurn value defines the modulo value, at which the counter rolls over (the modulo value is never reached). The counting range is: 0...diScaling_IncPerTurn -1.

This table describes the output variables:

Output	Type	Default	Comment
xValid	BOOL	FALSE	TRUE indicates that the output values on the function block are valid. If the function block is disabled, the output is set to FALSE .
xError	BOOL	FALSE	TRUE indicates that an error is detected. You can trigger a rising edge on xEnable to reset the error.
etErrorId	ET_ENC_ERROR_M262	ENC_ERROR_NO	Indicates the code of the detected error when xError is TRUE.
diNbTurns	DINT	0	Indicates the modulo value of the encoder. In incremental mode, it is incremented when the counter rolls over its up-limit. It is decremented when the counter rolls over its down-limit. In SSI mode $diNbTurns = \text{raw (SSI value - preset value)} / \text{udiScaling\_IncPerTurn}$ The raw SSI value comes directly from SSI, without any transformation.
diCurrentValue	DINT	0	In linear mode, indicates the value of the position of device in pulses. Range of value is: - 2 147 483 648...2 147 483 647. In rotary mode, indicates the value of the position in pulses for each turn of the mechanics. Range of value for diCurrentValue is $0...diScaling\_IncPerTurn - 1$.
lrCurrentValue_Unit	LREAL	0	Indicates the value of the encoder in units in turns of the mechanics. $diCurrentValue\_Unit = diCurrentValue / \text{udiScaling\_IncPerUnit}$ when $udiScaling\_IncPerUnit \geq 1$.

FB_EncoderPreset_M262: Preset the Encoder

Function Block Description

This function block is used to preset the encoder, in incremental or SSI mode.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation* ([see page 37](#)).

I/O Variable Description

This table describes the input variables:

Input	Type	Default	Comments
ENC_REF_M262	ENC_REF_M262	–	Reference of the encoder instance.
xEnable	BOOL	FALSE	TRUE enables the encoder preset function, via: • The preset mode using REF on I0 and Z on the encoder • The xForce input of the function block
xForce	BOOL	FALSE	On rising edge, presets and starts the counter if xEnable is TRUE.
etREF_Input	ET_ENC_INPUT_M262	ENC_INPUT_REF_I0	Defines the REF input. The only valid value is I0 (see page 32).
etMode	ET_ENC_PRESET_MODE_M262	ENC_PRESET_NO	Selects the conditions to preset the counting function with REF and Z inputs (see page 33).
diPresetValue	DINT	0	Defines the value loaded in the encoder actual value at preset event.

This table describes the output variables:

Output	Type	Default	Comment
xValid	B00L	FALSE	TRUE indicates that the output values on the function block are valid.
xError	B00L	FALSE	TRUE indicates that an error is detected. You can trigger a rising edge on xEnable to reset the error.
etErrorId	ET_ENC_ERROR_M262	ENC_ERROR_NO	Indicates the code of the detected error when xError is TRUE (<i>see page 31</i>).
xPresetFlag	B00L	FALSE	Set to TRUE for one cycle by the preset of the encoder.

FB_EncoderCapture_M262: Capture the Encoder Value

Function Block Description

This function block is used to capture the encoder value, in incremental or SSI mode.

To configure several instances of this function block, define different etCAP_Input.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation* ([see page 37](#)).

I/O Variable Description

This table describes the input variables:

Input	Type	Default	Comment
ENC_REF_M262	ENC_REF_M262	–	Reference of the encoder instance.
xEnable	BOOL	FALSE	TRUE enables the encoder capture function, via the capture input specified by the etCAP_Input input.
etCAP_Input	ET_ENC_INPUT_M262	ENC_INPUT_CAP_I1	Defines the input used for the capture function (see page 32).
etCAP_Edge	ET_ENC_CAP_EDGE_M262	ENC_CAP_EDGE_RISING	Indicates the edge detection for capture input (see page 30).

This table describes the output variables:

Output	Type	Default	Comment
xValid	BOOL	FALSE	TRUE indicates that the output values on the function block are valid.
xError	BOOL	FALSE	TRUE indicates that an error is detected. You can trigger a rising edge on xEnable to reset the error.
etErrorId	ET_ENC_ERROR_M262	ENC_ERROR_NO	Indicates the code of the detected error when xError is TRUE (see page 31).
xCaptureFlag	BOOL	FALSE	TRUE indicates that a cycle is defined by the encoder capture event. xCaptureFlag is therefore TRUE for only one cycle.
diCapturedValue	DINT	0	Indicates the captured value in pulses, valid at xCaptureFlag rising edge. Captured value remains until next xCaptureFlag occurs. Captured value is reset to 0 when xEnable set to FALSE.
lrCapturedValue_Units	LREAL	0.0	Indicates the captured value in units, valid at xCaptureFlag rising edge. Captured value remains until next xCaptureFlag occurs. Captured value is reset to 0 when xEnable set to FALSE.

FB_EncoderReadScalingParam_M262: Read the Scaling Parameter

Function Block Description

This function block is used to read the active values of the scaling parameter used to compute the unit value, in incremental or SSI mode.

Graphical Representation



IL and ST Representation

To see the general representation in IL or ST language, refer to the chapter *Function and Function Block Representation* (see page 37).

I/O Variable Description

This table describes the input variables:

Input	Type	Default	Comment
ENC_REF_M262	ENC_REF_M262	—	Reference of the encoder instance.
xEnable	BOOL	FALSE	TRUE enables the encoder function block reading active values of the scaling parameter used to compute <code>lrCurrentValue_Unit</code> . FALSE disables the function block.

This table describes the output variables:

Output	Type	Default	Comment
xValid	BOOL	FALSE	TRUE indicates that the output values on the function block are valid.
xError	BOOL	FALSE	TRUE indicates that an error is detected. You can trigger a rising edge on <code>xEnable</code> to reset the error.
udiScalingNbOfIncs	UDINT	0	Indicates the active value of <code>udiScalingNbOfIncs</code> to compute <code>lrCurrentValue_Unit</code> .
udiScalingNbOfUnits	UDINT	0	Indicates the active value of <code>udiScalingNbOfUnits</code> to compute <code>lrCurrentValue_Unit</code> .
udiScaling_IncPerTurn	UDINT	0	Indicates the active value of <code>udiScaling_IncPerTurn</code> to compute <code>lrCurrentValue_Unit</code> .

Chapter 3

M262 Logic/Motion Controller Library Data Types

Overview

This chapter describes the data types of the M262 encoder library.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
ET_ENC_CAP_EDGE_M262: Encoder Capture Codes	30
ET_ENC_ERROR_M262: Encoder Error Codes	31
ET_ENC_INPUT_M262: Encoder Input Codes	32
ET_ENC_PRESET_MODE_M262: Encoder Preset Mode Codes	33

ET_ENC_CAP_EDGE_M262: Encoder Capture Codes

Enumerated Type Description

This enumeration describes the types of edges which can be used for reference and capture on an encoder function block.

The ET_ENC_CAP_EDGE_M262 enumeration data type contains the following values:

Parameter Name	Value	Description
ENC_CAP_EDGE_RISING	0	Capture on input rising edge.
ENC_CAP_EDGE_FALLING	1	Capture on input falling edge.
ENC_CAP_EDGE_BOTH	2	Capture on input both edges.

ET_ENC_ERROR_M262: Encoder Error Codes

Enumerated Type Description

This enumeration describes the types of errors which can occur on an encoder function block.

The ET_ENC_ERROR_M262 enumeration data type contains the following values:

Parameter Name	Value	Description
ENC_ERROR_NO	0	No error detected.
ENC_ERROR_REF	1	The encoder reference is incorrect or not configured.
ENC_ERROR_PARAMETER_INVALID	3	The value of a parameter is incorrect.
ENC_ERROR_COM	4	A communication error is detected with the encoder.
ENC_ERROR_SUPPLY	11	Encoder supply not detected.
ENC_ERROR_I0_EVT_CONFIGURED	12	I0 is configured as an event and cannot be used for preset.
ENC_ERROR_RESERVED	13	FB_Encoder_M262 function block is reserved.

ET_ENC_INPUT_M262: Encoder Input Codes

Enumerated Type Description

This enumeration describes the types of inputs which can be used for reference and capture on an encoder function block.

The ET_ENC_INPUT_M262 enumeration data type contains the following values:

Parameter Name	Value	Description
ENC_INPUT_REF_I0	0	REF input on I0 for preset.
ENC_INPUT_CAP_I1	1	Capture input on I1.
ENC_INPUT_CAP_I2	2	Capture input on I2.
ENC_INPUT_CAP_I3	3	Capture input on I3.

ET_ENC_PRESET_MODE_M262: Encoder Preset Mode Codes

Enumerated Type Description

This enumeration describes the different types of preset mode which can be used for an encoder function block.

The ET_ENC_PRESET_MODE_M262 enumeration data type contains the following values:

Parameter Name	Value	Description
ENC_PRESET_NO	0	No preset configured.
ENC_PRESET_Z_EDGE_RISING	1	Preset on Z rising edge (incremental encoder only).
ENC_PRESET_Z_EDGE_FALLING	2	Preset on Z falling edge (incremental encoder only).
ENC_PRESET_Z_EDGE_BOTH	3	Preset on Z both edges (incremental encoder only).
ENC_PRESET_REF_RISING	4	Preset on REF rising edge.
ENC_PRESET_REF_FALLING	5	Preset on REF falling edge.
ENC_PRESET_REF_BOTH	6	Preset on REF both edges.
ENC_PRESET_Z_EDGE_RISING_AND_REF	7	Preset on Z rising edge and REF (incremental encoder only).
ENC_PRESET_EDGE_RISING_Z_FIRST_AND_REF	8	Preset on the first Z rising edge and REF (incremental encoder only).
ENC_PRESET_EDGE_RISING_Z_FIRST_AND_NO_REF	9	Preset on the first Z rising edge and no REF (incremental encoder only).

Appendices



Appendix A

Function and Function Block Representation

Overview

Each function can be represented in the following languages:

- IL: Instruction List
- ST: Structured Text
- LD: Ladder Diagram
- FBD: Function Block Diagram
- CFC: Continuous Function Chart

This chapter provides functions and function blocks representation examples and explains how to use them for IL and ST languages.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Differences Between a Function and a Function Block	38
How to Use a Function or a Function Block in IL Language	39
How to Use a Function or a Function Block in ST Language	43

Differences Between a Function and a Function Block

Function

A function:

- is a POU (Program Organization Unit) that returns one immediate result.
- is directly called with its name (not through an instance).
- has no persistent state from one call to the other.
- can be used as an operand in other expressions.

Examples: boolean operators (AND), calculations, conversion (BYTE_TO_INT)

Function Block

A function block:

- is a POU (Program Organization Unit) that returns one or more outputs.
- needs to be called by an instance (function block copy with dedicated name and variables).
- each instance has a persistent state (outputs and internal variables) from one call to the other from a function block or a program.

Examples: timers, counters

In the example, Timer_ON is an instance of the function block TON:

```
1  PROGRAM MyProgram_ST
2  VAR
3      Timer_ON: TON; // Function Block Instance
4      Timer_RunCd: BOOL;
5      Timer_PresetValue: TIME := T#5S;
6      Timer_Output: BOOL;
7      Timer_ElapsedTime: TIME;
8  END_VAR
```

```
1  Timer_ON(
2      IN:=Timer_RunCd,
3      PT:=Timer_PresetValue,
4      Q=>Timer_Output,
5      ET=>Timer_ElapsedTime);
```

How to Use a Function or a Function Block in IL Language

General Information

This part explains how to implement a function and a function block in IL language.

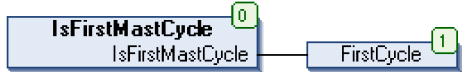
Functions `IsFirstMastCycle` and `SetRTCDrift` and Function Block `TON` are used as examples to show implementations.

Using a Function in IL Language

This procedure describes how to insert a function in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to <i>Adding and Calling POUs (see EcoStruxure Machine Expert, Programming Guide)</i> .
2	Create the variables that the function requires.
3	If the function has 1 or more inputs, start loading the first input using LD instruction.
4	Insert a new line below and: - type the name of the function in the operator column (left field), or - use the Input Assistant to select the function (select Insert Box in the context menu).
5	If the function has more than 1 input and when Input Assistant is used, the necessary number of lines is automatically created with ??? in the fields on the right. Replace the ??? with the appropriate value or variable that corresponds to the order of inputs.
6	Insert a new line to store the result of the function into the appropriate variable: type ST instruction in the operator column (left field) and the variable name in the field on the right.

To illustrate the procedure, consider the Functions `IsFirstMastCycle` (without input parameter) and `SetRTCDrift` (with input parameters) graphically presented below:

Function	Graphical Representation
without input parameter: <code>IsFirstMastCycle</code>	
with input parameters: <code>SetRTCDrift</code>	

In IL language, the function name is used directly in the operator column:

Function	Representation in POU IL Editor		
IL example of a function without input parameter: IsFirstMastCycle	1 PROGRAM MyProgram_IL 2 VAR 3 FirstCycle: BOOL; 4 END_VAR 5 1 IsFirstMastCycle ST FirstCycle <tr><td>IL example of a function with input parameters: SetRTCDrift</td><td> 1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag </td></tr>	IL example of a function with input parameters: SetRTCDrift	1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag
IL example of a function with input parameters: SetRTCDrift	1 PROGRAM MyProgram_IL 2 VAR 3 myDrift: SINT (-29..29) := 5; 4 myDay: DAY_OF_WEEK := SUNDAY; 5 myHour: HOUR := 12; 6 myMinute: MINUTE; 7 myDiag: RTCSETDRIFT_ERROR; 8 END_VAR 9 1 LD myDrift SetRTCDrift myDay myHour myMinute ST myDiag		

Using a Function Block in IL Language

This procedure describes how to insert a function block in IL language:

Step	Action
1	Open or create a new POU in Instruction List language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POU's (see <i>EcoStruxure Machine Expert, Programming Guide</i>).
2	Create the variables that the function block requires, including the instance name.
3	Function Blocks are called using a CAL instruction: ● Use the Input Assistant to select the FB (right-click and select Insert Box in the context menu). ● Automatically, the CAL instruction and the necessary I/O are created. Each parameter (I/O) is an instruction: ● Values to inputs are set by "=". ● Values to outputs are set by "=>".
4	In the CAL right-side field, replace ??? with the instance name.
5	Replace other ??? with an appropriate variable or immediate value.

To illustrate the procedure, consider this example with the TON Function Block graphically presented below:

Function Block	Graphical Representation
TON	The diagram shows a central TON (Timer On Delay) function block. It has two inputs: IN and PT. It has two outputs: Q and ET. The IN input is connected to a block labeled 'Timer_RunCd'. The PT input is connected to a block labeled 'Timer_PresetValue'. The Q output is connected to a block labeled 'Timer_Output'. The ET output is connected to a block labeled 'Timer_ElapsedTime'. The outputs are labeled with green boxes containing numbers 0, 1, and 2. The TON block is labeled 'Timer_ON' and 'TON'.

In IL language, the function block name is used directly in the operator column:

Function Block	Representation in POU IL Editor
TON	1 PROGRAM MyProgram_IL 2 VAR 3 Timer_ON: TON; // Function Block instance declaration 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 9 1 CAL Timer_ON(IN:= Timer_RunCd, PT:= Timer_PresetValue, Q=> Timer_Output, ET=> Timer_ElapsedTime)

How to Use a Function or a Function Block in ST Language

General Information

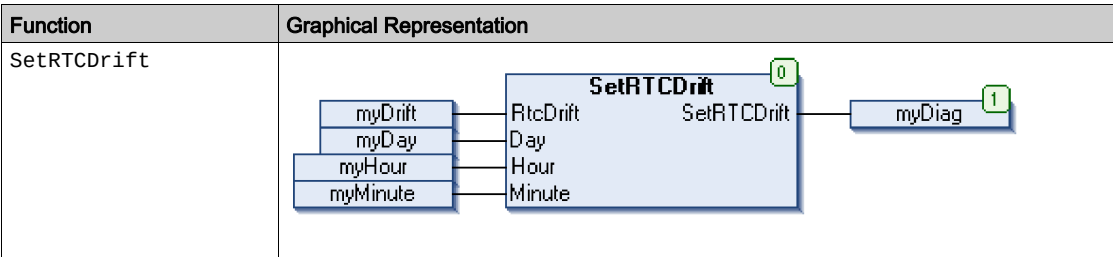
This part explains how to implement a Function and a Function Block in ST language.
 Function SetRTCDrift and Function Block TON are used as examples to show implementations.

Using a Function in ST Language

This procedure describes how to insert a function in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. NOTE: The procedure to create a POU is not detailed here. For more information, refer to Adding and Calling POUs (<i>see EcoStruxure Machine Expert, Programming Guide</i>).
2	Create the variables that the function requires.
3	Use the general syntax in the POU ST Editor for the ST language of a function. The general syntax is: <code>FunctionResult:= FunctionName(VarInput1, VarInput2, .. VarInputx);</code>

To illustrate the procedure, consider the function SetRTCDrift graphically presented below:



The ST language of this function is the following:

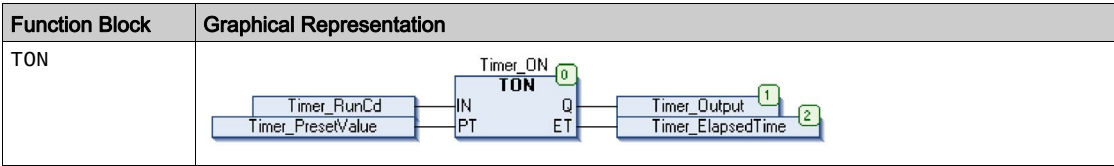
Function	Representation in POU ST Editor
SetRTCDrift	<pre> PROGRAM MyProgram_ST VAR myDrift: SINT(-29..29) := 5; myDay: DAY_OF_WEEK := SUNDAY; myHour: HOUR := 12; myMinute: MINUTE; myRTCAjust: RTCDRIFT_ERROR; END_VAR myRTCAjust:= SetRTCDrift(myDrift, myDay, myHour, myMinute); </pre>

Using a Function Block in ST Language

This procedure describes how to insert a function block in ST language:

Step	Action
1	Open or create a new POU in Structured Text language. NOTE: The procedure to create a POU is not detailed here. For more information on adding, declaring and calling POUs, refer to the related documentation (<i>see EcoStruxure Machine Expert, Programming Guide</i>).
2	Create the input and output variables and the instance required for the function block: • Input variables are the input parameters required by the function block • Output variables receive the value returned by the function block
3	Use the general syntax in the POU ST Editor for the ST language of a Function Block. The general syntax is: FunctionBlock_InstanceName(Input1:=VarInput1, Input2:=VarInput2, ... Ouput1=>VarOutput1, Ouput2=>VarOutput2, ...);

To illustrate the procedure, consider this example with the TON function block graphically presented below:



This table shows examples of a function block call in ST language:

Function Block	Representation in POU ST Editor
TON	<pre>1 PROGRAM MyProgram_ST 2 VAR 3 Timer_ON: TON; // Function Block Instance 4 Timer_RunCd: BOOL; 5 Timer_PresetValue: TIME := T#5S; 6 Timer_Output: BOOL; 7 Timer_ElapsedTime: TIME; 8 END_VAR 1 Timer_ON(2 IN:=Timer_RunCd, 3 PT:=Timer_PresetValue, 4 Q=>Timer_Output, 5 ET=>Timer_ElapsedTime);</pre>



B

byte

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C

CFC

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

F

FB

(*function block*) A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

function block diagram

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

I

IL

(*instruction list*) A program written in the 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 (refer to IEC 61131-3).

INT

(*integer*) A whole number encoded in 16 bits.

L

LD

(*ladder diagram*) 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 (refer to IEC 61131-3).

P

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.

S

ST

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

V

variable

A memory unit that is addressed and modified by a program.



D

data types

- ET_ENC_CAP_EDGE_M262, *30*
- ET_ENC_ERROR_M262, *31*
- ET_ENC_INPUT_M262, *32*
- ET_ENC_PRESET_MODE_M262, *33*

E

encoder modes

- incremental, *14*
- SSI absolute, *17*
- ET_ENC_CAP_EDGE_M262
 - data type, *30*
- ET_ENC_ERROR_M262
 - data type, *31*
- ET_ENC_INPUT_M262
 - data type, *32*
- ET_ENC_PRESET_MODE_M262
 - data type, *33*

F

- FB_Encoder_M262
 - function block, *20*
- FB_EncoderCapture_M262
 - function block, *25*
- FB_EncoderPreset_M262
 - function block, *23*
- FB_EncoderReadScalingParam_M262
 - function block, *27*
- function blocks
 - FB_Encoder_M262, *20*
 - FB_EncoderCapture_M262, *25*
 - FB_EncoderPreset_M262, *23*
 - FB_EncoderReadScalingParam_M262, *27*
- functions
 - differences between a function and a

- function block, *38*

- how to use a function or a function block in IL language, *39*
- how to use a function or a function block in ST language, *43*

I

incremental

- encoder modes, *14*

S

SSI absolute

- encoder modes, *17*

Modicon M262

MotionInterface

Library Guide

EIO0000004353.01
03/2021

Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

© 2021 Schneider Electric. All rights reserved.

Table of Contents

Safety Information.....	5
Qualification of Personnel	5
Intended Use.....	6
Before You Begin.....	6
Start-up and Test	7
Operation and Adjustments	7
About the Book.....	8
Presentation of the Library	13
General Information.....	13
Enumerations	14
<i>ET_AxisState</i> - General Information	14
<i>ET_Buffer_Mode</i> - General Information	14
<i>ET_CaptureEdge</i> - General Information	15
<i>ET_Direction</i> - General Information	15
<i>ET_ErrorSource</i> - General Information	16
<i>ET_InterpolationMode</i> - General Information	17
<i>ET_JobState</i> - General Information	18
<i>ET_Master_Start_Mode</i> - General Information	18
<i>ET_MotionInterfaceType</i> - General Information	19
<i>ET_OperationMode</i> - General Information	19
<i>ET_Result</i> - General Information.....	20
<i>ET_Slave_Start_Mode</i> - General Information.....	25
Function Blocks.....	27
<i>FB_ControlledAxis</i>	27
<i>FB_ControlledAxis</i> - General Information	27
<i>FB_ControlledAxis</i> - <i>SetAxisTypeLinearWithLimits</i> (Method)	28
<i>FB_ControlledAxis</i> - <i>SetAxisTypeLinearWithoutLimits</i> (Method)	29
<i>FB_ControlledAxis</i> - <i>SetAxisTypeModulo</i> (Method).....	30
<i>FB_ControlledAxis</i> - <i>SetErrorStopRamp</i> (Method).....	30
<i>FB_CustomJobBase</i>	31
<i>FB_CustomJobBase</i> - General Information	31
<i>FB_CustomJobBase</i> - <i>CalculateMovement</i> (Method).....	33
<i>FB_CustomJobBase</i> - <i>Prepare</i> (Method)	33
Functions.....	35
<i>FC_EtJobStateToString</i> - General Information	35
<i>FC_EtResultToString</i> - General Information	35
Interfaces	37
<i>IF_Axis</i>	37
<i>IF_Axis</i> - General Information	37
<i>IF_Axis</i> - <i>SetAxisTypeLinearWithLimits</i> (Method)	38
<i>IF_Axis</i> - <i>SetAxisTypeLinearWithoutLimits</i> (Method).....	39
<i>IF_Axis</i> - <i>SetAxisTypeModulo</i> (Method)	40
<i>IF_Axis</i> - <i>SetErrorStopRamp</i> (Method)	40
Structures.....	42
<i>ST_AxisError</i> - General Information	42
<i>ST_CustomJobCalculateParameter</i> - General Information.....	42
<i>ST_CustomJobPrepareParameter</i> - General Information.....	43

<i>ST_InterpolationParameter</i> - General Information	43
<i>ST_InterpolationPointXYVA</i> - General Information	44
<i>ST_MovementValues</i> - General Information	45
Index	47

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a "Danger" or "Warning" safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

Qualification of Personnel

A qualified person is one who has the following qualifications:

- Skills and knowledge related to the construction and operation of electrical equipment and the installation.
- Knowledge and experience in industrial control programming.
- Received safety-related training to recognize and avoid the hazards involved.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical,

electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Intended Use

This product is a library to be used together with the control systems and servo amplifiers intended solely for the purposes as described in the present documentation as applied in the industrial sector.

Always observe the applicable safety-related instructions, the specified conditions, and the technical data.

Perform a risk evaluation concerning the specific use before using the product. Take protective measures according to the result.

Since the product is used as a part of an overall system, you must ensure the safety of the personnel by means of the design of this overall system (for example, machine design).

Any other use is not intended and may be hazardous.

Before You Begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠ WARNING
UNGUARDED EQUIPMENT • Do not use this software and related automation equipment on equipment which does not have point-of-operation protection. • Do not reach into machinery during operation. Failure to follow these instructions can result in death, serious injury, or equipment damage.

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-

operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

Start-up and Test

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check are made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

Operation and Adjustments

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

Document Scope

This document describes the functionalities contained in the MotionInterface library.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Related Documents

Title of documentation	Reference number
Modicon M262 Logic/Motion Controller - Hardware Guide	EIO0000003659 (eng)
	EIO0000003660 (fre)
	EIO0000003661 (ger)
	EIO0000003662 (spa)
	EIO0000003663 (ita)
	EIO0000003664 (chi)
	EIO0000003665 (por)
	EIO0000003666 (tur)

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POU's found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF PROGRAM ORGANIZATION UNITS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the Program Organization Units (POUs) are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POU's.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Always evaluate the return values when using POU's of a library.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update your application program as required, paying particular attention to I/O address adjustments, whenever you modify the hardware configuration.

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

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.

⚠ WARNING

UNINTENDED MOVEMENT OF THE AXIS

- Ensure the proper functioning of the functional safety equipment before commissioning.
- Ensure that you can stop axis movements at any time using functional safety equipment (limit switch, emergency stop) before and during commissioning.

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

⚠ WARNING

UNINTENDED MOVEMENT OF THE SLAVE AXIS

Deactivate the POU that instructs the slave, or disconnect the connection from the master, if the slave axis stops independently from the master.

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

Motion function blocks, except for the Homing function blocks, can only be activated after the mechanical position reference has been established. This is especially important after the start-up of the Sercos motion bus.

⚠ WARNING

INCORRECT HOMING REFERENCE TO MECHANICAL SYSTEM

Ensure that a valid mechanical position reference exists by performing commissioning tests for all operating modes.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Presentation of the Library

General Information

Description

The library MotionInterface comprises the *IF_Axis* type used as input for PLCopen function blocks. It includes enumerations and records (data structures) for functions and states of axes.

The library contains the function block *FB_ControlledAxis* that can be used as virtual axes for applications.

The library includes the function block *FB_CustomJobBase* that user code has to derive from in order to implement a custom job algorithm.

Characteristics of This Library

The following table summarizes the characteristics of the library:

Characteristics	Value
Library title	MotionInterface
Company	Schneider Electric
Category	System
Component	CoreLibraries
Default namespace	MOIN
Language model attribute	Qualified-access-only (see EcoStruxure Machine Expert, Functions and Libraries User Guide)
Forward compatible library	No

NOTE: For this library, qualified-access-only is set. Therefore, the POU's, data structures, enumerations, and constants have to be accessed using the namespace of the library. The default namespace of the library is MOIN.

Enumerations

ET_AxisState - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration describes the axis states according to the PLCopen state machine.

Enumeration Elements

Name	Value (UDINT)	Description
<i>ErrorStop</i>	0	An emergency stop is active for the axis.
<i>Disabled</i>	1	The axis is disabled.
<i>Standstill</i>	2	The axis is not moving.
<i>Stopping</i>	3	The axis is stopping or has stopped.
<i>Homing</i>	4	The axis is being homed.
<i>DiscreteMotion</i>	5	The axis executes a movement for a limited period of time.
<i>ContinuousMotion</i>	6	The axis executes a movement for an unlimited period of time.
<i>SynchronizedMotion</i>	7	The axis executes a movement synchronously with a master.

ET_Buffer_Mode - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used as an input option of motion function blocks. It defines the method for the start of a new/buffered movement with regard to the ongoing movement.

Enumeration Elements

Name	Value (INT)	Description
<i>Aborting</i>	0	The ongoing movement is aborted and the new movement is executed immediately in the next possible real-time cycle.
<i>Buffered</i>	1	The new/buffered movement is executed as soon as the ongoing movement has reached its steady-state, which corresponds to the function block output <i>Done</i> , <i>InVelocity</i> , <i>InSync</i> , or <i>EndOfProfile</i> , depending on the ongoing movement. The buffered job becomes active immediately in the real-time-cycle when the previous job reaches its steady-state. It does not wait for the outputs to become true in the next application task cycle afterwards.
<i>BlendingLow</i>	2	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the lower of

Name	Value (INT)	Description
		the two velocity values of the ongoing movement and the new/buffered movement.
<i>BlendingPrevious</i>	3	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the velocity value of the ongoing movement.
<i>BlendingNext</i>	4	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the velocity value of the new/buffered movement.
<i>BlendingHigh</i>	5	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the higher of the two velocity values of the ongoing movement and the new/buffered movement.

ET_CaptureEdge - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is an option for *IF_Trigger* of *MC_TouchProbe* selecting the input edge to be used for activating position capture.

Enumeration Elements

Name	Value (UDINT)	Description
<i>FallingEdge</i>	0	<i>MC_TouchProbe</i> activates position capture with a falling edge of the selected input.
<i>RisingEdge</i>	1	<i>MC_TouchProbe</i> activates position capture with a rising edge of the selected input.
<i>BothEdges</i>	2	<i>MC_TouchProbe</i> activates position capture with a both falling edge and a rising edge of the selected input.

ET_Direction - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration describes the direction of movement for an *MC_MoveAbsolute* with a modulo axis.

Enumeration Elements

Name	Value (UDINT)	Description
<i>PositiveDirection</i>	0	Positive direction of movement.
<i>NegativeDirection</i>	1	Negative direction of movement.
<i>ShortestWay</i>	2	The direction of movement depends on whether the positive direction of movement or the negative direction of movement has the shortest distance to the target position.

ET_ErrorSource - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used as an output of an axis to describe the source of a detected error (function block or system component).

Enumeration Elements

Name	Value (UDINT)	Description
<i>NoErrorSource</i>	0	No detected error with determinable source
<i>UndefinedErrorSource</i>	1	Indeterminable error source
<i>McCamIn</i>	2	Potential source of detected error: command via <i>MC_CamIn</i>
<i>McGearIn</i>	3	Potential source of detected error: command via <i>MC_GearIn</i>
<i>McPhasingAbsolute</i>	4	Potential source of detected error: command via <i>MC_PhasingAbsolute</i>
<i>McAbortTrigger</i>	5	Potential source of detected error: command via <i>MC_AbortTrigger</i>
<i>McCustomJob</i>	6	Potential source of detected error: command via <i>MC_CustomJob</i>
<i>McHalt</i>	7	Potential source of detected error: command via <i>MC_Halt</i>
<i>McHome</i>	8	Potential source of detected error: command via <i>MC_Home</i>
<i>McMoveAbsolute</i>	9	Potential source of detected error: command via <i>MC_MoveAbsolute</i>
<i>McMoveAdditive</i>	10	Potential source of detected error: command via <i>MC_MoveAdditive</i>
<i>McMoveRelative</i>	11	Potential source of detected error: command via <i>MC_MoveRelative</i>
<i>McMoveVelocity</i>	12	Potential source of detected error: command via <i>MC_MoveVelocity</i>
<i>McPower</i>	13	Potential source of detected error: command via <i>MC_Power</i>
<i>McReset</i>	14	Potential source of detected error: command via <i>MC_Reset</i>
<i>McSetPosition</i>	15	Potential source of detected error: command via <i>MC_SetPositoIn</i>
<i>McStop</i>	16	Potential source of detected error: command via <i>MC_Stop</i>
<i>McTouchProbe</i>	17	Potential source of detected error: command via <i>MC_TouchProbe</i>
<i>AxisLimits</i>	18	Potential source of detected error: command conflicting with the movement limits of the axis
<i>AxisModulo</i>	19	Potential source of detected error: command conflicting with the modulo definition of the axis
<i>ErrorStopRamp</i>	20	Potential source of detected error: execution of an error stop
<i>AbsolutePositioning</i>	21	Potential source of detected error: movement with an absolute position

Name	Value (UDINT)	Description
<i>InternalFirmware</i>	22	Potential source of detected error: internal firmware functionality
<i>RealTimeTask</i>	23	Potential source of detected error: timing conflict
<i>PlcApplication</i>	24	Potential source of detected error: incorrect behavior of the logic controller application
<i>McMoveSuperimposed</i>	25	Potential source of detected error: command via <i>MC_MoveSuperimposed</i>
<i>PositioningJob</i>	26	Potential source of detected error: positioning movement
<i>MotionJobNotClassified</i>	27	Indeterminable error source
<i>ErrorStop</i>	28	Potential source of detected error: error stop
<i>StoppingJob</i>	29	Potential source of detected error: stop movement
<i>Encoder</i>	30	Potential source of detected error: signal of an encoder
<i>MultiAxisGroup</i>	31	Potential source of detected error: command via MultiAxisGroup

ET_InterpolationMode - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

⚠ WARNING
UNINTENDED EQUIPMENT OPERATION - Verify that the number of interpolation points you specify for the input <i>InterpolationPoints</i> is the same value you specify for <i>udiNumCamPoints</i> of the structure <i>ST_InterpolationParameter</i> used for the input <i>InterpolationParameter</i> if you use an interpolated cam. - Verify that the values for X of the structure <i>ST_InterpolationPointXYVA</i> increase strictly monotonically if you use an interpolated cam with cam law Poly5. - Verify that the data in the array of cam points is not modified while the cam is buffered or being executed. - Verify that no online modifications are triggered while the cam is being executed. Failure to follow these instructions can result in death, serious injury, or equipment damage.

This enumeration is used as a parameter for *MC_CamIn* to define the type of interpolation between the given cam points if the cam is defined via an array of interpolation points at the input *InterpolationPoints*. Refer to the description of *MC_CamIn* in the Synchronized Motion Library Guide for details on interpolated cams.

Enumeration Elements

Name	Value (UDINT)	Description
<i>YArrayLinear</i>	0	Straight line between two points is used for interpolation.
<i>XYVAArrayPoly5</i>	1	General Poly5 between two points consisting of master position, slave position, velocity, and acceleration between two points is used for interpolation.

ET_JobState - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used as an output of *FB_CustomJobBase* (algorithm of a custom job) to describe its state.

Enumeration Elements

Name	Value (UDINT)	Description
<i>Idle</i>	0	Algorithm is not commanded to be executed.
<i>Linked</i>	1	Command triggered to execute algorithm, but not yet started.
<i>Executing</i>	2	Algorithm is executing a movement with a defined final condition.
<i>Steady</i>	3	Algorithm is executing a movement without defined final condition.
<i>Done</i>	4	Algorithm has finished execution.
<i>Aborted</i>	5	Algorithm was replaced by another algorithm before final condition was reached.
<i>Error</i>	6	Execution of algorithm was superseded by an error response.
<i>ScheduledToBeAborted</i>	7	Command triggered for a different algorithm to abort this algorithm, but yet not effective.
<i>ScheduledToBeDone</i>	8	Algorithm has reached the final condition in this RealTimeCycle and will be replaced in the next cycle.

ET_Master_Start_Mode - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used as an input option of *MC_CamIn*. It selects if the absolute master position is taken for cam calculation or if it is ignored and assumed to be the described master position at cam start.

This enumeration is used as an input of *MC_CamIn*. In the first cycle when the cam is started, it specifies how the value of the master (current X value) as seen by slave of the cam is determined, based on the current position of the master axis.

Enumeration Elements

Name	Value (INT)	Description
<i>Absolute</i>	0	The cam starts at the X coordinate equal to the absolute master axis position at the time the cam gets active. If the cam is buffered after another cam with the same master, the master as seen by slave of the first cam is used instead.
<i>Relative</i>	1	The cam starts at the X coordinate equal to the lowest X coordinate of the cam points. The resulting offset between the position of the master axis and the master as seen by slave is kept during the movement of the cam.

ET_MotionInterfaceType - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration specifies the type of an axis. It is used as type of the property *etInterfaceType* of every axis.

Enumeration Elements

Name	Value (UDINT)	Description
<i>SimpleMotion</i>	0	Simplified Axis for executing superimposed and phasing movements
<i>Feedback</i>	1	External driven axis such as an encoder axis which cannot execute movement commands
<i>Coordinate</i>	2	Axis with feedback and movement commands driven by drives

ET_OperationMode - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration specifies the operating mode for the function block *MC_MoveVelocity*. See *MC_MoveVelocity* in the *M262 Synchronized Motion Control Library Guide* for details on the operating mode Cyclic Synchronous Velocity.

Enumeration Elements

Name	Value (UDINT)	Description
<i>Position</i>	0	Velocity control with active position control loop in drive (Cyclic Synchronous Position).
<i>Velocity</i>	1	Cyclic Synchronous Velocity, pure velocity control.

ET_Result - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used to return identifiers of detected errors for functions and function blocks.

Enumeration Elements

Name	Value (UDINT)	Description
<i>Ok</i>	0	No error detected
<i>UnexpectedReturnValue</i>	1	Indeterminable return value from system. Contact your Schneider Electric representative.
<i>SemaphoreError</i>	2	A multi-tasking conflict could not be solved via a semaphore.
<i>LimitModeInvalid</i>	3	The specified limit mode is not valid
<i>NoBusCommunication</i>	4	Communication via the motion bus is interrupted
<i>PositionOutsideModulo</i>	5	The target position is outside of modulo range of the axis. Set the target position to a value within the modulo range (from 0 to modulo value of the axis).
<i>JerkOutOfRange</i>	6	The value at the input <i>Jerk</i> is less than zero. Use a positive value or zero at the input <i>Jerk</i> .
<i>AccelerationOutOfRange</i>	7	The value at the input <i>Acceleration</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Acceleration</i> .
<i>DecelerationOutOfRange</i>	8	The value at the input <i>Deceleration</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Deceleration</i> .
<i>VelocityOutOfRange</i>	9	The value at the input <i>Velocity</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Velocity</i> .
<i>AlgorithmInvalid</i>	10	The defined algorithm is not valid.
<i>BufferModeInvalid</i>	11	A value different from <i>MC_Buffer_Mode.Aborting</i> or <i>MC_Buffer_Mode.Buffered</i> was supplied at the input <i>BufferMode</i> . Supply the value of <i>MC_Buffer_Mode.Aborting</i> or <i>MC_Buffer_Mode.Buffered</i> as <i>BufferModeInput</i> (if no value has previously been supplied, <i>MC_Buffer_Mode.Aborting</i> is used).
<i>AxisIsDisabled</i>	12	Function block cannot be executed because the axis is in the operating state Disabled. Verify that the axis is not in the operating state Disabled when attempting to start a new function block.
<i>AxisIsStopping</i>	13	Function block cannot be executed because an <i>MC_Stop</i> function block is active and the axis is in the operating state Stopping. Verify that the axis is not in the operating state Stopping when attempting to start a new function block.
<i>AxisNotHomed</i>	14	The axis is not homed (flag <i>xHomed</i> of axis is FALSE).

Name	Value (UDINT)	Description
		Home the axis to obtain a valid zero point to start a movement relative to the zero point.
<i>AxisInErrorStop</i>	15	Function block cannot be executed because an axis error has been detected and the axis is in operating state <i>ErrorStop</i> . Verify that the axis is not in the operating state <i>ErrorStop</i> when attempting to start a new function block.
<i>BufferSaturated</i>	16	The maximum number of function blocks that can be buffered for the axis has been reached. Buffer only one function block for a given axis at any point in time.
<i>BufferNotSupported</i>	17	Buffering of this command or combination (e.g. blending) is not permissible.
<i>PLCopenStateInvalid</i>	18	The <i>PLCopen</i> operating state is not valid.
<i>JobInvalid</i>	19	The defined job is not valid.
<i>MasterInvalid</i>	20	The object at the input <i>Master</i> is invalid. Provide a valid reference to the axis for which the function block is to be executed (object from <i>Devices</i> tree such as an axis or an encoder).
<i>OutOfMemory</i>	21	Insufficient memory for movement command. Reduce the memory required by your application.
<i>NoAccessToData</i>	22	The required data could not be read.
<i>LimitsInvalid</i>	23	The limits are invalid.
<i>AxisNotDisabled</i>	24	The command can only be executed when the axis is in the operating state <i>Disabled</i> .
<i>InvalidMasterAddress</i>	25	The specified master is not valid.
<i>InvalidRatioNumerator</i>	26	The value at the input <i>RatioNumerator</i> is zero. Use a value other than zero for the numerator.
<i>InvalidRatioDenominator</i>	27	The value at the input <i>RatioDenominator</i> is zero. Use a value other than zero for the denominator.
<i>AxisInvalid</i>	28	No axis is specified for the input <i>Axis</i> or specified axis does not support the required function. For <i>MC_Touchprobe</i> and <i>MC_AbortTrigger</i> : Specified axis does not support capture. Connect the axis for which the function block is to be executed to the input <i>Axis</i> . For <i>MC_Touchprobe</i> and <i>MC_AbortTrigger</i> : Use an axis that supports capture.
<i>DriveInvalid</i>	29	The specified drive is invalid.
<i>DriveNotDisabled</i>	30	The command can only be executed when the axis is in the operating state <i>Disabled</i> .
<i>ExistingConnection</i>	31	A connection is already existing.
<i>ModuloAxisNotSupported</i>	32	The command cannot be executed with a modulo axis.
<i>NotSupportedWithDrive</i>	33	The command cannot be executed with the specified drive type.
<i>PowerStateError</i>	34	Error detected relating to power state of device.
<i>DriveInError</i>	35	The drive is in the operating state error. Use the function block <i>MC_Reset</i> to reset the detected error.
<i>HomingIsAlreadyActive</i>	36	The axis is being homed. Verify that the axis is in operating mode "Standstill" before executing this function block.

Name	Value (UDINT)	Description
<i>AxisNotInStandstill</i>	37	The axis was not in operating state Standstill when homing was attempted to start. Verify that the axis is in operating mode Standstill before executing this function block.
<i>JobStartedWhileAxisIsHoming</i>	38	The command cannot be executed while the axis is in the operating state Homing.
<i>AxisResetInExecutingState</i>	39	Drive has detected an error while axis was executing.
<i>InvalidCamTableID</i>	40	The <i>CamTableID</i> is invalid. Verify that a correct cam table is provided for <i>MC_CamIn</i> via the input <i>CamTableID</i> .
<i>MasterIsNotModulo</i>	41	The specified master has to be defined as a modulo axis.
<i>LastMovementIsInvalid</i>	42	The ongoing job has caused an invalid movement.
<i>InvalidLambda</i>	43	One of the points of the electronic cam has an invalid Lambda value. Lambda is the value of the next cam segment preceding the inflection point. Permissible values for Lambda: $0 < \text{Lambda} < 1$.
<i>InvalidC</i>	44	One of the points of the electronic cam has an invalid C value. C is the value of the next curved segment of the electronic cam. Permissible values for C: $0 < C \leq 1$.
<i>InvalidM</i>	45	One of the points of the electronic cam has an invalid M value. M is the slope of the electronic cam at the position for which M is defined.
<i>InvalidK</i>	46	One of the points of the electronic cam has an invalid K value. K is the curvature of the electronic cam at the position for which K is defined. The value must be 0 for a simple sine (<i>ET_CamType</i> = <i>SimplSin</i>) and for a general fifth degree polynomial (<i>ET_CamType</i> = <i>Poly5Com</i>).
<i>InvalidCustomJob</i>	47	The specified custom job is invalid.
<i>InvalidFloatingValue</i>	48	A specified REAL/LREAL is not a valid number (for example NaN (not a number), infinite).
<i>MemAllocFailed</i>	49	No more controller memory available.
<i>EventDeleteFailure</i>	50	A system event could not be deregistered.
<i>ModuloRangeInvalid</i>	51	The specified modulo range is invalid. Use a value greater than zero.
<i>InvalidCaptureSource</i>	52	The specified capture source does not exist. Verify that the capture source is supported by the device.
<i>DeviceAccessFailed</i>	53	Error detected writing/reading via the service channel in Sercos phase 4. Reduce the frequency of access to the service channel with <i>FB_WriteIDN</i> and/or <i>FB_ReadIDN</i> .
<i>CaptureSourceAlreadyInUse</i>	54	The same capture source is used for two function blocks <i>MC_TouchProbe</i> . Only use a single <i>MC_TouchProbe</i> with a given capture source at a time.
<i>InvalidConfiguration</i>	55	The configuration for <i>MC_TouchProbe</i> is invalid. Verify the configuration for <i>MC_Touchprobe</i> .
<i>NoCamInJobOnSlaveAxis</i>	56	<i>MC_CamIn</i> is not active for the slave axis specified. <i>MC_Phasing</i> can only be executed if <i>MC_CamIn</i> is active for the specified axis
<i>MasterAxisNotHomed</i>	57	The master axis has not been homed.

Name	Value (UDINT)	Description
		Running <i>MC_CamIn</i> with <i>mcAbsolute</i> for <i>MC_Master_Start_Mode</i> requires a homed master axis.
<i>RealTimeConfigurationOfParameterFailed</i>	58	The IDNs could not be mapped in the real-time channel. Verify that cyclic data can be used and that it is possible to map the IDNs for this device.
<i>DrivePowerLoss</i>	59	Power outage at connected drive.
<i>NotSupportedWithFeedbackAxis</i>	60	Command not permitted in conjunction with an axis of type feedback, such as an encoder axis. Provide a correct axis type at the input <i>Axis</i> .
<i>ErrorInEncoderCallbackResultDetected</i>	61	Communication to encoder is interrupted.
<i>InvalidFeedResolution</i>	62	Feed resolution is invalid.
<i>InvalidFeedConstant</i>	63	Feed constant is invalid.
<i>NoEncoderSupplyDetected</i>	64	No encoder power supply Ensure correct encoder power supply.
<i>InvalidDigitalInputConfiguration</i>	65	The configuration of the digital input of the controller for the encoder is invalid. Verify correct configuration of the digital input for the encoder.
<i>InvalidDeviceHandle</i>	66	No device with the specified handle.
<i>ErrorSettingOutputs</i>	67	The outputs cannot be changed.
<i>StartAdditiveJobDuringSuperimpose</i>	68	Additive jobs cannot start while the axis performs a superimposed movement.
<i>HomingNotStarted</i>	69	Homing could not be started.
<i>InvalidDirection</i>	70	The specified direction parameter is invalid.
<i>InternalErrorInLockingMovementChange</i>	71	Internal multi-task handling error detected.
<i>InternalErrorInLockingDrive</i>	72	Internal multi-task handling error detected.
<i>InternalErrorInLockingHoming</i>	73	Internal multi-task handling error detected.
<i>InternalErrorInResettingAxis</i>	74	Internal multi-task handling error detected.
<i>PositionOutOfSetLimits</i>	75	Position generation error detected.
<i>AxisInInvalidState</i>	76	The axis is not in a valid state.
<i>PLCApplcationStoppedWhileAxisExecutingJob</i>	77	The movement was interrupted by a stop of the logic controller application.
<i>AxisNotHomedAndHasInvalidLastMovement</i>	78	No valid position for axis could be determined.
<i>InternalErrorInCyclicCalculation</i>	79	<i>FB_CustomJobBase</i> has returned an invalid LREAL value. Correct your implementation of <i>FB_CustomJobBase</i> so that it does not return invalid LREAL values (infinite and NaN (not a number) are invalid LREAL values).
<i>HomingFailed</i>	80	Error detected during homing.
<i>BlendingOvershootsFirstJob</i>	81	Blending movement would need to move further than the target of position movement.
<i>JobTypeNotAllowedToBeBlended</i>	82	The movements cannot be blended.
<i>PreactiveJobNotAllowedToBeFollowedByBlending</i>	83	The job becoming active cannot be followed by a blended movement.
<i>ActiveJobNotAllowedToBeFollowedByBlending</i>	84	The active job cannot be followed by a blended movement.
<i>FBBusyBufferModeNotPossible</i>	85	Starting a buffered command is not possible while the function block is busy.
<i>TimeNotRecorded</i>	86	No time stamps received from Sercos slave during phase up.
<i>PhaseUpForbiddenDueToLicense</i>	87	Phase-up not permitted, for example, too many axes configured.

Name	Value (UDINT)	Description
<i>MasterMovementDataNotValid</i>	88	No valid master data available, for example, caused by communication interruption.
<i>EncoderCommunicationError</i>	89	Encoder data cannot be read.
<i>EncoderPowerError</i>	90	No encoder power supply.
<i>ExpertIoError</i>	91	Expert I/O error detected in encoder module.
<i>InvalidCustomJobStateTransition</i>	92	Custom job has sent a job state which does not match the previously sent job state.
<i>ExceededTxMaxRamSize</i>	93	Insufficient memory for storing connection data to the Sercos slaves
<i>NoSynchronousMotionToDeregister</i>	94	Error in internal master slave registering handling detected.
<i>AxisUsedAsMasterForSynchronousMotion</i>	95	This command cannot be executed for an axis which is master for a different axis.
<i>MasterAxisCurrentlyHoming</i>	96	This command cannot be executed when the master axis is being homed.
<i>InvalidBus</i>	97	The bus connection is not supported.
<i>AxisAlreadyUsed</i>	98	The axis is already used by another function block (for example, slave channel)
<i>InvalidOperationForActiveMultiAxisJob</i>	99	This command cannot be executed when multi-axis job is executed with the axis.
<i>InvalidHandle</i>	100	No device or object connected to the specified handle.
<i>NullObject</i>	101	The specified interface or pointer is zero and not connected to an object.
<i>MultiAxisGroupIsExecuting</i>	102	This command cannot be executed when multi-axis job is executed with the axis.
<i>NoSlaveChannelAddedToMultiAxisJob</i>	103	Functions on a slave channel can only be executed when the slave channel is connected to a multi-axis group.
<i>ASlaveChannelOfMultiAxisGroupWasUnableToStart</i>	104	The axis of one of the slave channels could not start the job for the slave channel.
<i>UserSpecifiedErrorStop</i>	105	An error stop was triggered by the user.
<i>MultiAxisGroupChannelLock</i>	106	Internal multi-task handling error detected.
<i>CannotWriteMovementValuesToSlaveChannel</i>	107	The axis of the slave channel could not be set to the specified position.
<i>SlaveChannelInvalid</i>	108	The specified slave channel is not valid
<i>MasterChannelInvalid</i>	109	The specified master channel is not valid.
<i>InvalidMultiAxisGroupCallback</i>	110	The multi-axis group could not register to the internal event.
<i>MaxNumberOfSlaveChannelsExceeded</i>	111	The maximum number of slave channels is exceeded for the controller.
<i>MaxNumberOfMasterChannelsExceeded</i>	112	The maximum number of master channels is exceeded for the controller.
<i>MultiAxisGroupNotStarted</i>	113	Commands for the multi-axis group can only be executed if the multi-axis group is running.
<i>EmergencyStopRequiredByPlcApplication</i>	114	An emergency stop was triggered by the logic controller application.
<i>InvalidCaptureEdge</i>	115	The specified capture edge is not valid.
<i>JobAborted</i>	116	The running job was aborted by another job.
<i>AxisHoming</i>	117	The command cannot be executed while the axis is being homed.
<i>SlaveChannelNotAddedToMultiAxisGroup</i>	118	Functions on a slave channel can only be executed when the slave channel is connected to a multi-axis group.
<i>MasterChannelNotAddedToMultiAxisGroup</i>	119	Functions on a master channel can only be executed when the slave channel is connected to a multi-axis group

Name	Value (UDINT)	Description
<i>XValuesNotStrictlyMonotonic</i>	120	X values do not increase strictly monotonically throughout the cam profile. Define a cam profile with X values that increase strictly monotonically.
<i>OperationModeChangeNotAllowedForAxisNotInStandstill</i>	121	An attempt was made to change the operating mode for an axis that is not in the state Standstill. Ensure that the axis is in the state Standstill before changing the operating mode.
<i>OperationModeIDNsNotMapped</i>	122	A job with operating mode "Velocity" was started for a drive for which no velocity IDNs are mapped. Verify that the operating mode "Velocity" is activated in the drive feature configuration (checkbox VelocityOperationMode).
<i>MasterSlaveCascadeFormsLoop</i>	123	Master/slave function blocks create a loop (the master itself as slave, or a following slave with the initial master as slave). Resolve the loop in the chain.
<i>OperationModeChangeNotAllowedForMasterAxisNotInStandstill</i>	124	An attempt was made to change the operating mode to "Velocity" for an axis that is used as master for a synchronous movement and that is not in the state Standstill. Ensure that the master axis is in the state Standstill before changing the operating mode.

ET_Slave_Start_Mode - General Information

Overview

Type:	List type
Available as of:	V1.1.75.6318

Description

This enumeration is used as input option of *MC_CamIn*. It selects if, at start, the cam output position is used as slave axis position or if the slave axis position does not change.

When the cam gets active after a positive edge at the input *Execute* of *MC_CamIn*, this enumeration defines how to handle a position delta between the first calculated position of the cam law and the position the slave axis has when the cam gets active.

NOTE: Discrepancies between the physical position of the cam and the position in the cam definition may provoke position jumps if you use the slave start mode *Absolute*. If there is a position difference between the actual position of the slave and its calculated start position and this start position can be reached in spite of the position difference, this movement may be made in the form of a sudden position jump.

WARNING

UNINTENDED EQUIPMENT OPERATION

Verify that the physical position of the slave axis at the start of the cam matches the position in the cam definition.

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

Enumeration Elements

Name	Value (INT)	Description
<i>Absolute</i>	0	For starting the cam, the position of the slave axis is directly set to the first calculated Y value. The slave axis position is calculated on the basis of the cam definition and "master as seen by slave". As opposed to the slave start modes <i>Relative</i> and <i>RampIn</i> where there are no offsets and superimposed movements. The reference velocity and the acceleration are calculated on the basis of the cam definition. If there is a position difference between the actual position of the slave and its calculated start position (Y value) for the cam, and if this start position cannot be reached within one task scan, an error is detected. However, if this start position can be reached in spite of the position difference, this movement may be made in the form of a sudden position jump.
<i>Relative</i>	1	After the master as seen by slave for the start is determined according to <i>ET_MasterStartMode</i>, the current cam position is calculated. For the duration of the cam running, the cam is shifted by the difference between this calculated cam position at start and the axis position at this time. This prevents a position jump at the start of the cam while respecting the shape of the cam.
<i>RampIn</i>	2	It is assumed that the absolute slave axis position is equal to the cam Y coordinate for the cam to be in sync. At the beginning, the cam starts like slave relative, meaning f(X start) is correlated with the absolute slave axis position when the cam starts. Then a ramp-in movement is performed which offsets the slave axis to align the coordinate system of the axis position with the coordinate system of Y.

Function Blocks

FB_ControlledAxis

FB_ControlledAxis - General Information

Overview

Type:	Function block
Available as of:	V1.1.75.6318
Inherits from:	-
Implements:	<i>IF_Axis</i>

Task

This function block represents an axis which can perform movements.

Description

By defining a variable of type *FB_ControlledAxis*, you can create a virtual axis. A variable of this type can be used as input of *MC_MoveRelative* or other motion function blocks. The movement is represented by the value changes of the property *IrPosition*.

Methods

Name	Description
<i>SetAxisTypeLinearWithLimits</i>	Sets axis type to linear with limited movement range.
<i>SetAxisTypeLinearWithoutLimits</i>	Sets axis type to linear without limited movement range.
<i>SetAxisTypeModulo</i>	Sets axis type to modulo.
<i>SetErrorStopRamp</i>	Sets deceleration and jerk for stopping if an error is detected.

Properties

Name	Data type	Access	Description
<i>etAxisState</i>	<i>ET_AxisState</i>	Read	Operating state of the axis according to PLCopen state machine
<i>etInterfaceType</i>	<i>ET_MotionInterfaceType</i>	Read	Type of the axis
<i>IrAcceleration</i>	LREAL	Read	Acceleration of the axis in units/s ²
<i>IrErrorStopDec</i>	LREAL	Read	Maximum deceleration in units/s ² for an error stop movement of the axis
<i>IrErrorStopJerk</i>	LREAL	Read	Jerk in units/s ³ for an error stop movement of the axis
<i>IrModuloPeriod</i>	LREAL	Read	Modulo period of the axis. If axis limits are defined as modulo, the axis position is kept in a range of [0... <i>IrModuloPeriod</i>]. Otherwise, the value of this parameter is zero.
<i>IrNegativeDirectionLimit</i>	LREAL	Read	If the axis is defined as linear with limited movement range, the negative position limit is provided. Otherwise, the value is 0. The axis performs an emergency stop if the position of the axis is moved in a negative direction beyond this limit. If the axis position value is less than this limit, only movement commands which increase the axis position values are allowed.

Name	Data type	Access	Description
<i>lrPosition</i>	LREAL	Read	Position of the axis in units
<i>lrPositiveDirectionLimit</i>	LREAL	Read	If the axis is defined as linear with limited movement range, the positive position limit is provided. Otherwise, the value is 0. The axis performs an emergency stop if the position of the axis is moved in a positive direction beyond this limit. If the axis position value is less than this limit, only movement commands which decrease the axis position values are allowed.
<i>lrVelocity</i>	LREAL	Read	Velocity of the axis in units/s
<i>stAxisError</i>	REFERENCE TO <i>ST_AxisError</i>	Read	Detected error of an axis.
<i>stMotionOfMaster</i>	REFERENCE TO <i>ST_MovementValues</i>	Read	If a cam is performed for the axis, position in units, velocity in units/s and acceleration in units/s ² of the master as seen by slave is shown, otherwise all values are 0.
<i>stMotionOfSuperimposed</i>	REFERENCE TO <i>ST_MovementValues</i>	Read	Position in units, velocity in units/s, and acceleration in units/s ² of a superimposed movement of the axis.
<i>udiHandle</i>	UDINT	Read	Handle which is connected to the axis
<i>xisHomed</i>	BOOL	Read/write	If TRUE, the position of the axis is defined as a correct representation of the position of the mechanical system it moves. Movement commands based on the absolute position of the axis such as <i>MC_MoveAbsolute</i> require a homed axis.
<i>xisLimited</i>	BOOL	Read	If TRUE, the axis limits are defined as linear axis with limited movement range. If the axis position value exceeds the limits, an error stop movement is executed.
<i>xisModulo</i>	BOOL	Read	If TRUE, the axis limits are defined as modulo. If the axis position value falls below 0, it is increased by the axis period value. If the axis position becomes equal to or greater than the axis period value, it is reduced by the axis period value. This position jump has no effect on the physical movement of the drive that controls the axis.

FB_ControlledAxis - SetAxisTypeLinearWithLimits (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a linear axis type with limited movement range.

Description

This method sets the axis to a linear axis type with a limited movement range in positive and negative directions of movement. If the axis position exceeds the position limits, an error stop is triggered. If the axis has moved to a position beyond one of the limits, only movements in the opposite direction (in the direction towards the valid movement range) can be started.

The axis type can only be modified when the axis is disabled.

The limitation of the movement range is only active if the axis is homed (*xisHomed* = True).

Interface

Input	Data type	Description
<i>i_lrNegativeDirectionLimit</i>	LREAL	Minimum value of the movement range of the axis.
<i>i_lrPositiveDirectionLimit</i>	LREAL	Maximum value of the movement range of the axis.

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

FB_ControlledAxis - SetAxisTypeLinearWithoutLimits (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a linear axis type without limited movement range.

Description

This method sets the axis to a linear axis type without a limited movement range. There are no restrictions with regard to the movement range.

The axis type can only be modified when the axis is disabled.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Take all measures required to restrict movements to the movement range identified as permissible in your machine design and your risk assessment.
- Implement application functionality to keep the absolute position of the axis from exceeding the value appropriate for your machine.
- Take into account the precision limitations of the data type LREAL and floating-point numbers in your application.

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

Interface

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

FB_ControlledAxis - SetAxisTypeModulo (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a modulo axis type.

Description

This method sets the axis to a modulo axis type. The axis position is kept in a range of $[0 \dots i\_IrPeriod]$. If the axis position value falls below 0, it is increased by the axis period value. If the axis position becomes equal to or greater than the axis period value, it is reduced by the axis period value. This position jump has no effect on the physical movement of the drive that controls the axis. It does not affect the velocity or the acceleration of the axis.

The axis type can only be modified when the axis is disabled.

Interface

Input	Data type	Description
<i>i_IrPeriod</i>	LREAL	With a value of $\neq 0$, the position is kept within the period value $(0 \dots i\_IrPeriod)$. If a period value is too high or too low, the position is manipulated by the period value. Only positive values are valid.
Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

FB_ControlledAxis - SetErrorStopRamp (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets deceleration and jerk for stopping in response to a detected error.

Description

This method sets the deceleration and jerk for stopping in response to a detected error.

Settings can only be modified when the axis is disabled.

Interface

Input	Data type	Description
<i>i_lrDeceleration</i>	LREAL	Specifies the deceleration in units/s ² .
<i>i_lrJerk</i>	LREAL	Specifies the jerk in units/s ³ .

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the values for deceleration and jerk are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

FB_CustomJobBase

FB_CustomJobBase - General Information

Overview

Type:	Function block
Available as of:	V1.1.75.6318
Inherits from:	-
Implements:	<i>CmpEventManager.ICmpEventCallback</i>

Task

This function block allows you to implement user-defined movements that cannot be performed with the available function.

Description

If you want to define an algorithm for a specific movement which is not provided by any of the available function blocks, you need to implement your own user-defined function block. This user-defined function block inherits from *FB_CustomJobBase* (by using the EXTENDS key word). Only the methods *Prepare* and *Calculate* are allowed to be overridden or called.

When a movement is started via *MC_MoveCustomJob* with the user-defined function block as parameter, the movement is activated in the first real-time cycle and the system calls the *Prepare* method of the user-defined function block. In this call, the system provides the information on the current state of the axis and the optional master axis. This information can be used to initialize the user-defined algorithm and, for example, enable it to be started during movement of the axis. It is not required to execute tasks in *Prepare* if not needed by the user-defined algorithm.

In the same cycle in which *Prepare* is called, the *CalculateMovement* method is called by the system for the first time. After that, *CalculateMovement* is called in every real-time cycle. In this method, the position, velocity and acceleration for the axis have to be provided for this cycle. In addition, an appropriate job state has to be set:

- *ET_JobState.Executing* as long as a discrete movement is being executed
- *ET_JobState.Steady* as long as a continuous movement is being executed
- *ET_JobState.Done* when the movement is finished

Methods

Name	Description
<i>CalculateMovement</i>	Called by system every real-time cycle to calculate axis movement values.
<i>Prepare</i>	Called by system once to initialize the movement algorithm.

Properties

Name	Data type	Accessing	Description
<i>etAxisState</i>	<i>ET_AxisState</i>	Read/write	The operating state of the PLCopen state machine the axis is to be in while the custom job is running. Only <i>DiscreteMotion</i> , <i>ContinuousMotion</i> and <i>SynchronousMotion</i> are permissible.

Example

```

PROGRAM SR_Main
VAR
fbCustomIncrementPosition : FB_CustomIncrementPosition;
fbCustomJob : PLCO.MC_CustomJob;
fbPower : PLCO.MC_Power;
fbReset : PLCO.MC_Reset;
END_VAR

fbReset(Axis := DRV_Lexium32S.Axis);
fbPower(Axis := DRV_Lexium32S.Axis);

fbCustomJob(Axis := DRV_Lexium32S.Axis,
            CustomJob := fbCustomIncrementPosition);
FUNCTION_BLOCK FB_CustomIncrementPosition EXTENDS MOIN.FB_
CustomJobBase
VAR
udiPrepared : UDINT;
udiCalculated : UDINT;
lrPosition : LREAL;
uiInc : UINT;
END_VAR

```

In addition to calculating the position, acceleration, and jerk of the movement, the function block can set the state. If you set it to *ET_JobState.Steady*, the output *InSteadyState* of *PLCO.MC_CustomJob* is activated, as shown in the following example:

```

METHOD CalculateMovement
VAR_IN_OUT
iq_stParameter : MOIN.ST_CustomJobCalculateParameter;
END_VAR
VAR_OUTPUT
q_lrPosition : LREAL;
q_lrVelocity : LREAL;
q_lrAcceleration : LREAL;
q_etJobState : MOIN.ET_JobState;
END_VAR

uiInc := uiInc + 1;
IF uiInc > 4096 THEN
uiInc := 0;
END_IF
udiCalculated := udiCalculated + 1;
IF iq_stParameter.xMasterDefined THEN
lrPosition := iq_stParameter.lrMasterPositon;
ELSE
lrPosition := lrPosition + 1.0;
END_IF

```



```

q_etJobState := MOIN.ET_JobState.Steady;
q_lrPosition := lrPosition;
METHOD Prepare
VAR_IN_OUT
iq_stParameter : MOIN.ST_CustomJobPrepareParameter;
END_VAR

lrPosition := iq_stParameter.lrSlavePositionLastCycle;
udiPrepared := udiPrepared + 1;

```

FB_CustomJobBase - CalculateMovement (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Provides movement values for an axis.

Description

This method must be overwritten by the user-defined function block *FB_CustomJob*. When *FB_CustomJob* is executed, the system calls this method every real-time cycle to get the values for position, velocity and acceleration of the axis. These values are extracted from the output variables of this method.

Interface

Output	Data type	Description
<i>q_lrPosition</i>	LREAL	Position of the axis for this real-time cycle in units.
<i>q_lrVelocity</i>	LREAL	Velocity of the axis for this real-time cycle in units/s.
<i>q_lrAcceleration</i>	LREAL	Acceleration of the axis for this real time cycle in units/s ² .
<i>q_etJobState</i>	<i>ET_JobState</i>	Execution state of the movement: <i>ET_JobState.Executing</i> as long as a discrete movement is being executed <i>ET_JobState.Steady</i> as long as a continuous movement is being executed <i>ET_JobState.Done</i> when the movement is finished

Input/Output	Data type	Description
<i>iq_stParameter</i>	<i>ST_CustomJobCalculateParameter</i>	Additional parameter to calculate the movement values of the axis. Refer to <i>ST_CustomJobCalculate</i> , page 42 for details.

FB_CustomJobBase - Prepare (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Initializes the user-defined algorithm.

Description

This method can be overwritten by the user-defined function block. When *FB_CustomJob* is executed, the system calls the prepare method in the first real time cycle, before the *CalculateMovement* method is called. No actions have to be executed in *Prepare*, but information on the starting condition of the axis is provided and the method can be used to initialize the user-defined algorithm.

Interface

Input/Output	Data type	Description
<i>iq_stParameter</i>	<i>ST_CustomJobCalculateParameter</i>	Additional parameter for starting condition of the axis. Refer to <i>ST_CustomJobPrepare</i> , page 43 for details.

Functions

FC_EtJobStateToString - General Information

Overview

Type:	Function
Available as of:	V1.1.75.6318

Task

Converts an enumeration element of type ET_JobState to a variable of type STRING.

Description

Using the function *FC_EtJobStateToString*, you can convert an enumeration element of type ET_JobState to a variable of type STRING.

Interface

Input	Data type	Description
<i>i_etResult</i>	ET_JobState	Enumeration with the job state.

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> .
<i>q_etResult</i>	ET_JobState	Result of the function block execution. Refer to the ET_Result Enumeration Elements, page 20.

Return Value

Data type	Description
STRING(80)	The ET_JobState converted to text.

FC_EtResultToString - General Information

Overview

Type:	Function
Available as of:	V1.1.75.6318

Task

Converts an enumeration element of type ET_Result to a variable of type STRING.

Description

Using the function *FC_EtResultToString*, you can convert an enumeration element of type ET_Result to a variable of type STRING.

Interface

Input	Data type	Description
<i>i_etResult</i>	ET_Result	Enumeration with the result.

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> .
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the ET_Result Enumeration Elements, page 20.

Return Value

Data type	Description
STRING(80)	The ET_Result converted to text.

Interfaces

IF_Axis

IF_Axis - General Information

Overview

Type:	Interface
Available as of:	V1.1.75.6318
Inherits from:	CMI.IF_AxisIdentificaiton

Task

This interface is the basic representation of any axis. The state and behavior of the axis can be monitored via this interface. It is used as input of any movement function block.

Description

This interface describes generic properties and methods to be provided by any axis type. It also serves as a generic type for any axis qualifying for use as function block input type.

Methods

Name	Description
<i>SetAxisTypeLinearWithLimits</i>	Sets axis type to linear with limited movement range.
<i>SetAxisTypeLinearWithoutLimits</i>	Sets axis type to linear without limited movement range.
<i>SetAxisTypeModulo</i>	Sets axis type to modulo.
<i>SetErrorStopRamp</i>	Sets deceleration and jerk for stopping if an error is detected.

Properties

Name	Data type	Access	Description
<i>etAxisState</i>	<i>ET_AxisState</i>	Read	Operating state of the axis according to PLCopen state machine
<i>etInterfaceType</i>	<i>ET_MotionInterfaceType</i>	Read	Type of the axis
<i>lrAcceleration</i>	LREAL	Read	Acceleration of the axis in units/s ²
<i>lrErrorStopDec</i>	LREAL	Read	Maximum deceleration in units/s ² for an error stop movement of the axis
<i>lrErrorStopJerk</i>	LREAL	Read	Jerk in units/s ³ for an error stop movement of the axis
<i>lrModuloPeriod</i>	LREAL	Read	Modulo period of the axis. If axis limits are defined as modulo, the axis position is kept in a range of [0... <i>lrModuloPeriod</i>]. Otherwise, the value of this parameter is zero.
<i>lrNegativeDirectionLimit</i>	LREAL	Read	If the axis is defined as linear with limited movement range, the negative position limit is provided. Otherwise, the value is 0. The axis performs an emergency stop if the position of the axis is moved in a negative direction beyond this limit. If the axis position value is less than this limit, only movement commands which increase the axis position values are allowed.

Name	Data type	Access	Description
<i>lrPosition</i>	LREAL	Read	Position of the axis in units
<i>lrPositiveDirectionLimit</i>	LREAL	Read	If the axis is defined as linear with limited movement range, the positive position limit is provided. Otherwise, the value is 0. The axis performs an emergency stop if the position of the axis is moved in a positive direction beyond this limit. If the axis position value is less than this limit, only movement commands which decrease the axis position values are allowed.
<i>lrVelocity</i>	LREAL	Read	Velocity of the axis in units/s
<i>stAxisError</i>	REFERENCE TO <i>ST_AxisError</i>	Read	Detected error of an axis.
<i>stMotionOfMaster</i>	REFERENCE TO <i>ST_MovementValues</i>	Read	If a cam is performed for the axis, position in units, velocity in units/s and acceleration in units/s ² of the master as seen by slave is shown, otherwise all values are 0.
<i>stMotionOfSuperimposed</i>	REFERENCE TO <i>ST_MovementValues</i>	Read	Position in units, velocity in units/s, and acceleration in units/s ² of a superimposed movement of the axis.
<i>udiHandle</i>	UDINT	Read	Handle which is connected to the axis
<i>xisHomed</i>	BOOL	Read/write	If TRUE, the position of the axis is defined as a correct representation of the position of the mechanical system it moves. Movement commands based on the absolute position of the axis such as <i>MC_MoveAbsolute</i> require a homed axis.
<i>xisLimited</i>	BOOL	Read	If TRUE, the axis limits are defined as linear axis with limited movement range. If the axis position value exceeds the limits, an error stop movement is executed.
<i>xisModulo</i>	BOOL	Read	If TRUE, the axis limits are defined as modulo. If the axis position value falls below 0, it is increased by the axis period value. If the axis position becomes equal to or greater than the axis period value, it is reduced by the axis period value. This position jump has no effect on the physical movement of the drive that controls the axis.

IF_Axis - SetAxisTypeLinearWithLimits (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a linear axis type with limited movement range.

Description

This method sets the axis to a linear axis type with a limited movement range in positive and negative directions of movement. If the axis position exceeds the position limits, an error stop is triggered. If the axis has moved to a position beyond one of the limits, only movements in the opposite direction (in the direction towards the valid movement range) can be started.

The axis type can only be modified when the axis is disabled.

The limitation of the movement range is only active if the axis is homed (*xisHomed* = True).

Interface

Input	Data type	Description
<i>i_lrNegativeDirectionLimit</i>	LREAL	Minimum value of the movement range of the axis.
<i>i_lrPositiveDirectionLimit</i>	LREAL	Maximum value of the movement range of the axis.

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

IF_Axis - SetAxisTypeLinearWithoutLimits (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a linear axis type without limited movement range.

Description

This method sets the axis to a linear axis type without a limited movement range. There are no restrictions with regard to the movement range.

The axis type can only be modified when the axis is disabled.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Take all measures required to restrict movements to the movement range identified as permissible in your machine design and your risk assessment.
- Implement application functionality to keep the absolute position of the axis from exceeding the value appropriate for your machine.
- Take into account the precision limitations of the data type LREAL and floating-point numbers in your application.

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

Interface

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

IF_Axis - SetAxisTypeModulo (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets the axis to a modulo axis type.

Description

This method sets the axis to a modulo axis type. The axis position is kept in a range of $[0 \dots i\_IrPeriod]$. If the axis position value falls below 0, it is increased by the axis period value. If the axis position becomes equal to or greater than the axis period value, it is reduced by the axis period value. This position jump has no effect on the physical movement of the drive that controls the axis. It does not affect the velocity or the acceleration of the axis.

The axis type can only be modified when the axis is disabled.

Interface

Input	Data type	Description
<i>i_IrPeriod</i>	LREAL	With a value of $\neq 0$, the position is kept within the period value $(0 \dots i\_IrPeriod)$. If a period value is too high or too low, the position is manipulated by the period value. Only positive values are valid.
Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the position limits of the axis are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

IF_Axis - SetErrorStopRamp (Method)

Overview

Type:	Method
Available as of:	V1.1.75.6318

Task

Sets deceleration and jerk for stopping in response to a detected error.

Description

This method sets the deceleration and jerk for stopping in response to a detected error.

Settings can only be modified when the axis is disabled.

Interface

Input	Data type	Description
<i>i_lrDeceleration</i>	LREAL	Specifies the deceleration in units/s ² .
<i>i_lrJerk</i>	LREAL	Specifies the jerk in units/s ³ .

Output	Data type	Description
<i>q_xError</i>	BOOL	If this output is set to TRUE, an error was detected. Refer to <i>ET_Result</i> . If the method is not successful, the values for deceleration and jerk are not modified.
<i>q_etResult</i>	<i>ET_Result</i>	Result of the function block execution. Refer to the <i>ET_Result</i> Enumeration Elements, page 20.

Structures

ST_AxisError - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

This structure defines the error of an axis. *IF_Axis* and *FB_ControlledAxis* have a property *stAxisError* of this type.

Structure Elements

Variable	Data type	Description
<i>etID</i>	ET_Result	Error ID naming the detected error.
<i>etSource</i>	ET_ErrorSource	Enumeration identifying the function block or component from which the error originates.

ST_CustomJobCalculateParameter - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

This structure is used as input for the *Calculate* method of *MC_CustomJobBase*. You can use the structure to create synchronized motion by calculating the axis movement based on values of a different axis (master axis). For this, a master axis has to be provided to an instance of *MC_CustomJob*.

Structure Elements

Variable	Data type	Description
<i>xMasterDefined</i>	BOOL	Indicates whether <i>MC_CustomJob</i> has been called with a master axis.
<i>xMasterHomed</i>	BOOL	Indicates whether the master axis is homed.
<i>lrMasterPositionChangeLastCycle</i>	LREAL	Position offset of the master axis between the present cycle and the previous cycle.
<i>lrMasterPositon</i>	LREAL	Position of master axis.
<i>lrMasterVelocity</i>	LREAL	Velocity of master axis.
<i>lrMasterAcceleration</i>	LREAL	Acceleration of master axis.

ST_CustomJobPrepareParameter - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

This structure is used as input for the *Prepare* method of *MC_CustomJobBase*. This structure initializes the movement to be performed via *MC_CustomJobBase* by using the axis movement values of its axis before starting.

Structure Elements

Variable	Data type	Description
<i>IrSlavePositionLastCycle</i>	LREAL	Last position of axis executing <i>MC_CustomJob</i> (current position is set during <i>Calculate</i> in this cycle).
<i>IrSlaveVelocityLastCycle</i>	LREAL	Last velocity of axis executing <i>MC_CustomJob</i> .
<i>IrSlaveAccelerationLastCycle</i>	LREAL	Last acceleration of axis executing <i>MC_CustomJob</i> .

ST_InterpolationParameter - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

This structure is used to parameterize an interpolated cam.

⚠ WARNING
UNINTENDED EQUIPMENT OPERATION - Verify that the number of interpolation points you specify for the input <i>InterpolationPoints</i> is the same value you specify for <i>udiNumCamPoints</i> of the structure <i>ST_InterpolationParameter</i> used for the input <i>InterpolationParameter</i> if you use an interpolated cam. - Verify that the values for X of the structure <i>ST_InterpolationPointXYVA</i> increase strictly monotonically if you use an interpolated cam with cam law Poly5. - Verify that the data in the array of cam points is not modified while the cam is buffered or being executed. - Verify that no online modifications are triggered while the cam is being executed. Failure to follow these instructions can result in death, serious injury, or equipment damage.

Structure Elements

Variable	Data type	Description
<i>udiNumCamPoints</i>	UDINT	Number of array entries filled with cam points.
<i>lrMinMasterPosition</i>	LREAL	Minimum position of the position range of the master. The value is ignored if <i>ET_InterpolationMode</i> is set to <i>XYVAArrayPoly5</i> .
<i>lrMaxMasterPosition</i>	LREAL	Maximum position of the position range of the master. The value is ignored if <i>ET_InterpolationMode</i> is set to <i>XYVAArrayPoly5</i> .
<i>etInterpolationMode</i>	ET_ InterpolationMode	Type of interpolated cam (linear or Poly5, refer to ET_ InterpolationMode, page 17 for details).

ST_InterpolationPointXYVA - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

This structure is used to specify the interpolation data for an interpolated cam with cam law Poly5.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the number of interpolation points you specify for the input *InterpolationPoints* is the same value you specify for *udiNumCamPoints* of the structure *ST_InterpolationParameter* used for the input *InterpolationParameter* if you use an interpolated cam.
- Verify that the values for X of the structure *ST_InterpolationPointXYVA* increase strictly monotonically if you use an interpolated cam with cam law Poly5.
- Verify that the data in the array of cam points is not modified while the cam is buffered or being executed.
- Verify that no online modifications are triggered while the cam is being executed.

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

Structure Elements

Variable	Data type	Description
X	LREAL	Master position of cam point.
Y	LREAL	Slave position of cam point.
V	LREAL	Velocity at cam point (corresponds to slope).
A	LREAL	Acceleration at cam point (corresponds to curvature).

ST_MovementValues - General Information

Overview

Type:	Data structure
Available as of:	V1.1.75.6318
Inherits from:	—

Description

Movement vector/tuple (position, velocity and acceleration) defining a movement state at a given point in time.

Structure Elements

Variable	Data type	Description
<i>IrPosition</i>	LREAL	Position of the axis.
<i>IrVelocity</i>	LREAL	Velocity of the axis.
<i>IrAcceleration</i>	LREAL	Acceleration of the axis.

Index

C

CalculateMovement 33

E

ET_AxisState 14
 ET_Buffer_Mode 14
 ET_CaptureEdge 15
 ET_Direction 15
 ET_ErrorSource 16
 ET_InterpolationMode 17
 ET_JobState 18
 ET_Master_Start_Mode 18
 ET_MotionInterfaceType 19
 ET_OperationMode 19
 ET_Result 20
 ET_Slave_Start_Mode 25

F

FB_ControlledAxis 27
 SetAxisTypeLinearWithLimits 28
 SetAxisTypeLinearWithoutLimits 29
 SetAxisTypeModulo 30
 SetErrorStopRamp 30
 FB_CustomJobBase 31
 CalculateMovement 33
 Prepare 33
 FC_EtJobStateToString 35
 FC_EtResultToString 35

I

IF_Axis 37
 SetAxisTypeLinearWithLimits 38
 SetAxisTypeLinearWithoutLimits 39
 SetAxisTypeModulo 40
 SetErrorStopRamp 40

M

MotionInterface
 ET_AxisState 14
 ET_Buffer_Mode 14
 ET_CaptureEdge 15
 ET_Direction 15
 ET_ErrorSource 16
 ET_InterpolationMode 17
 ET_JobState 18
 ET_Master_Start_Mode 18
 ET_MotionInterfaceType 19
 ET_OperationMode 19
 ET_Result 20
 ET_Slave_Start_Mode 25
 FB_ControlledAxis 27
 FB_CustomJobBase 31
 FC_EtJobStateToString 35
 FC_EtResultToString 35
 IF_Axis 37
 ST_AxisError 42
 ST_CustomJobCalculateParameter 42
 ST_CustomJobPrepareParameter 43
 ST_InterpolationParameter 43

ST_InterpolationPointXYVA 44
 ST_MovementValues 45

P

Prepare 33

S

SetAxisTypeLinearWithLimits 28, 38
SetAxisTypeLinearWithoutLimits 29, 39
SetAxisTypeModulo 30, 40
SetErrorStopRamp 30, 40
 ST_AxisError 42
 ST_CustomJobCalculateParameter 42
 ST_CustomJobPrepareParameter 43
 ST_InterpolationParameter 43
 ST_InterpolationPointXYVA 44
 ST_MovementValues 45

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2021 – Schneider Electric. All rights reserved.

EIO0000004353.01

Modicon M262

SercosMaster

Library Guide

EIO00000004624.00

11/2021

Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2021 Schneider Electric. All Rights Reserved.

Table of Contents

Safety Information.....	5
Qualification of Personnel	5
Proper Use.....	6
Before You Begin.....	6
Start-up and Test	7
Operation and Adjustments	7
About the Book.....	8
Presentation of the Library	12
General Information.....	12
Enumerations	13
<i>ET_IdentificationMode</i> - General Information	13
<i>ET_IpConfigMode</i> - General Information	13
<i>ET_OperationModeStatus</i> - General Information.....	13
<i>ET_PhysicalConnectionState</i> - General Information	14
<i>ET_Result</i> - General Information.....	14
<i>ET_SercosPhase</i> - General Information	19
<i>ET_SercosState</i> - General Information.....	20
<i>ET_ServiceChannelAccessingMode</i> - General Information	20
<i>ET_SlaveCommunicationState</i> - General Information.....	21
<i>ET_WorkingMode</i> - General Information	21
Function Blocks.....	23
<i>FB_ReadIDN</i> - General Information	23
<i>FB_WriteIDN</i> - General Information.....	24
Functions.....	27
<i>FC_EtResultToString</i> - General Information	27
<i>FC_EtSercosPhaseToString</i> - General Information.....	27
<i>FC_EtSercosStateToString</i> - General Information	28
<i>FC_SlaveGetCommunicationState</i> - General Information	29
Internal Functions	30
<i>FC_DriveGetError</i>	30
<i>FC_GetIdleTimeOnSercosInLastCycle</i>	30
<i>FC_GetMotionCycleTaskLoadOfLastCycle</i>	31
<i>FC_GetPhysicalConnectionState</i>	31
<i>FC_GetScaledFeedbackAcceleration</i>	32
<i>FC_GetScaledFeedbackVelocity</i>	33
<i>FC_ReadPositionFeedbackValue</i>	33
<i>FC_ReadScaledPositionFeedback</i>	34
<i>FC_ResetDiagnostic</i>	34
<i>FC_SercosGetConfiguration</i>	35
<i>FC_SercosGetCycleCount</i>	36
<i>FC_SercosGetSlaveCount</i>	36
Structures.....	38
<i>ST_SercosConfiguration</i> - General Information	38
<i>ST_SercosConfigurationDevice</i> - General Information.....	38
<i>ST_SercosTime</i> - General Information	38
<i>ST_Slave</i> - General Information.....	39
<i>ST_TypePlate</i> - General Information	39

Global Elements.....41

 Global Constants List (GCL).....41

Index 43

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a "Danger" or "Warning" safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

Qualification of Personnel

A qualified person is one who has the following qualifications:

- Skills and knowledge related to the construction and operation of electrical equipment and the installation.
- Knowledge and experience in industrial control programming.
- Received safety-related training to recognize and avoid the hazards involved.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical,

electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Proper Use

This product is a library to be used together with the control systems and servo amplifiers intended solely for the purposes as described in the present documentation as applied in the industrial sector.

Always observe the applicable safety-related instructions, the specified conditions, and the technical data.

Perform a risk evaluation concerning the specific use before using the product. Take protective measures according to the result.

Since the product is used as a part of an overall system, you must ensure the safety of the personnel by means of the design of this overall system (for example, machine design).

Any other use is not intended and may be hazardous.

Before You Begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠ WARNING UNGUARDED EQUIPMENT • Do not use this software and related automation equipment on equipment which does not have point-of-operation protection. • Do not reach into machinery during operation. Failure to follow these instructions can result in death, serious injury, or equipment damage.

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-

operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

Start-up and Test

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check are made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

Operation and Adjustments

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

Document Scope

The present document describes the functions provided by the SercosMaster library.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.2.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Product Related Information

 WARNING LOSS OF CONTROL • The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart. • Separate or redundant control paths must be provided for critical control functions. • System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link. • Observe all accident prevention regulations and local safety guidelines.¹ • Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service. Failure to follow these instructions can result in death, serious injury, or equipment damage.

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POUs found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF PROGRAM ORGANIZATION UNITS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the Program Organization Units (POUs) are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POUs.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Always evaluate the return values when using POUs of a library.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update your application program as required, paying particular attention to I/O address adjustments, whenever you modify the hardware configuration.

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

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.

⚠ WARNING

UNINTENDED MOVEMENT OF THE AXIS

- Ensure the proper functioning of the functional safety equipment before commissioning.
- Ensure that you can stop axis movements at any time using functional safety equipment (limit switch, emergency stop) before and during commissioning.

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

⚠ WARNING

UNINTENDED MOVEMENT OF THE SLAVE AXIS

Deactivate the POU that instructs the slave, or disconnect the connection from the master, if the slave axis stops independently from the master.

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

Motion function blocks, except for the Homing function blocks, can only be activated after the mechanical position reference has been established. This is especially important after the start-up of the Sercos motion bus.

⚠ WARNING

INCORRECT HOMING REFERENCE TO MECHANICAL SYSTEM

Ensure that a valid mechanical position reference exists by performing commissioning tests for all operating modes.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Presentation of the Library

General Information

Description

The library SercosMaster provides enumerations, structures, and functions for controlling and monitoring Sercos on Modicon M262 Motion Controllers. The library lets you perform Sercos operations such as reading, writing, resetting errors, and execute procedure commands. In addition, you can obtain information on the Sercos bus configuration and on devices on the Sercos bus.

Characteristics of This Library

The following table indicates the characteristics of the library:

Characteristics	Value
Library title	SercosMaster
Company	Schneider Electric
Category	System
Component	M262
Default namespace	S3M
Language model attribute	Qualified-access-only (see EcoStruxure Machine Expert, Functions and Libraries User Guide)
Forward compatible library	No

NOTE: For this library, qualified-access-only is set. Therefore, the POUs, data structures, enumerations, and constants have to be accessed using the namespace of the library. The default namespace of the library is S3M.

Enumerations

ET_IdentificationMode - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the type of addressing on the Sercos bus.

Enumeration Elements

Name	Value (DINT)	Description
<i>TopoAddress</i>	0	The address of the devices is based on the topology address. The topology address corresponds to the physical position of the device on the Sercos bus.
<i>SercosAddress</i>	1	The address of the devices is based on the Sercos address.

ET_IpConfigMode - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the options for setting the IP Address of a Sercos device. The IP address is configured either manually or automatically via the firmware.

Enumeration Elements

Name	Value (DINT)	Description
<i>AutomaticIpAddressAssignment</i>	0	The IP address of the device is configured automatically via the firmware during phase-up.
<i>ManualIpAddressAssignmentHardware</i>	1	IP address of the device is configured manually.

ET_OperationModeStatus - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the active operating mode of the drive.

Enumeration Elements

Name	Value (DINT)	Description
<i>Position</i>	0	The active operating mode of the drive is Cyclic Synchronous Position.
<i>Velocity</i>	1	The active operating mode of the drive is Cyclic Synchronous Velocity.
<i>Torque</i>	2	The active operating mode of the drive is Cyclic Synchronous Torque.
<i>NotAvailable</i>	3	Possible reasons: • The active operating mode is not a cyclic synchronous operating mode (for example Homing). • The drive has detected an error. • The drive is disabled. • The drive is not in Sercos communication phase 4.

ET_PhysicalConnectionState - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the connection states of a physical Sercos device. The values identify whether or not the firmware was able match a configured device to a device connected to the Sercos bus.

Enumeration Elements

Name	Value (INT)	Description
<i>ConnectionToPhysicalSlaveExistent</i>	0	A matching physical device for the configured device was found during phase-up.
<i>ConnectionToPhysicalSlaveNotExistent</i>	1	No matching physical device for the configured device was found during phase-up.

ET_Result - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the return values for Sercos function calls triggered by the Sercos master.

Enumeration Elements

Name	Value (INT)	Description
<i>Success_NoError</i>	0	Function call successful.
<i>InvalidSercosPhase</i>	1	Function call not successful due to invalid Sercos communication phase.
<i>InvalidSercosState</i>	2	Function call not successful due to invalid Sercos state.
<i>InvalidInputSI</i>	3	Reserved
<i>InvalidInputSE</i>	4	Reserved
<i>InvalidInputPS</i>	5	Reserved
<i>InvalidInputDBN</i>	6	Reserved
<i>InvalidResult</i>	7	Reserved
<i>Success_Pending</i>	8	Request is being processed.
<i>DeviceManagerInvalidDeviceHandle</i>	-1	The device handle parameter could not be matched to an existing device in the firmware.
<i>DeviceManagerNoPhysicalSlaveConnected</i>	-2	A matching device in the firmware for the handle was found, but no corresponding device on the Sercos bus (device may be simulated or the Sercos bus is not in the correct communication phase.)
<i>DeviceManagerExceptionOccurred</i>	-3	Reserved
<i>DeviceManagerDeviceInUse</i>	-4	An attempt was made to connect a device already connected on the Sercos bus to a device created in the program, or an attempt was made to delete the reference to a device on the Sercos bus that was still in use.
<i>ConnectionIDNAAlreadyConfigured</i>	-5	An attempt has been made to map an already mapped IDN to a device.
<i>ConnectionDataTooLong</i>	-6	Insufficient space in the telegram to add an additional IDN.
<i>ConnectionConfigFull</i>	-7	The maximum number of IDNs to map has been reached.
<i>ConnectionIDInvalid</i>	-8	The connection ID specified as a parameter value could not be found.
<i>ConnectionConfigurationLocked</i>	-9	An attempt was made to access a semaphore that was locked.
<i>NoTopologicalAddressConfigured</i>	-10	An attempt was made to access a device using a topological address or to get a topological address for a device while no topology address was configured before
<i>TopologicalAddressOutOfBounds</i>	-11	An attempt was made to set a topological address using an invalid value, or to access a device using a topological address without a device assigned to it.
<i>ServiceChannelProcedureCommandTimeout</i>	-12	The execution of the procedure command took longer than the specified timeout.
<i>TimeoutValueInvalid</i>	-13	Reserved
<i>ServiceChannelAccessInInvalidPhase</i>	-14	An attempt was made to access the service channel in a Sercos communication phase less than 2.
<i>ServiceChannelIDnInvalid</i>	-15	A procedure command was started with an IDN that is not intended for a procedure command, or an IDN was used incorrectly.
<i>DeviceIsNoSafeLogicDevice</i>	-16	An attempt was made to add a SafeNode to a device that does not support SafeNodes.
<i>SLCCreateInInvalidPhase</i>	-17	An attempt was made to create an SLC in Sercos communication phase 2 or greater.
<i>SLCAAlreadyCreated</i>	-18	An attempt was made to create an SLC, but an SLC already existed.
<i>SafeNodeModuleIndexInvalid</i>	-19	An attempt was madet to use a SafeModule index value less than 1 or greater than 255.
<i>SafeNodeModuleIndexInUse</i>	-20	An attempt was made to create a SafeNode with a module index value already used by a different SafeNode.

Name	Value (INT)	Description
<i>SafeNodeSafetyAddressInvalid</i>	-21	An attempt was made to use a SafetyAddress value less than 2 or greater than 102.4
<i>SafeNodeSafetyAddressInUse</i>	-22	An attempt was made to create a SafeNode with a SafetyAddress value already used by a different SafeNode.
<i>SafetyDomainNumberInvalid</i>	-23	An attempt was made to use a SafetyDomain number less than 1 or greater than 1024.
<i>SafeNodeConnectionInputLengthInvalid</i>	-24	An attempt was made to use a SafeNode connection input length less than 0 or greater than 1500.
<i>SafeNodeConnectionOutputLengthInvalid</i>	-25	An attempt was made to use a SafeNode connection output length less than 0 or greater than 1500.
<i>SafeNodeSercosAddressInvalid</i>	-26	An attempt was made to use a Sercos address value less than 0 or greater than 511.
<i>DeviceManagerInvalidSafeDeviceHandle</i>	-27	An attempt was made to access an SLC, but no SLC had been created, or the handle to access the SLC was invalid.
<i>InvalidIdn</i>	-28	Reserved
<i>EmptyProcessData</i>	-29	An attempt was made to access cyclic telegram data for a cyclic telegram that did not exist.
<i>NoDeviceFacadeSet</i>	-30	Reserved
<i>DriveAccessInInvalidPhase</i>	-31	An attempt was made to interact with the drive during a Sercos communication phase less than 4.
<i>CyclicDataLockChangeLock</i>	-32	Changing the lock of a semaphore was not successful.
<i>DrivesNotEnabled</i>	-33	Error detected after an attempt to move a drive that is not enabled.
<i>DrivesConnctedToAxis</i>	-34	fc_convertincrementsstounits was called for a drive with no axis connected, or fc_drivesetenable was called for a drive that is connected to an axis.
<i>DriveInvalidFeedConstant</i>	-35	The position resolution value set is either not a numeric value or less than 1.
<i>DrivesEnabled</i>	-36	Error detected after an attempt to modify values of scaling parameters for an enabled drive.
<i>DriveInvalidGearValue</i>	-37	The gear value set is invalid.
<i>DevicesStatic</i>	-38	Device could not be deleted because it is static.
<i>SercosAddressNotAssigned</i>	-39	Sercos address could not be assigned.
<i>NoSercosAddressConfigured</i>	-40	An attempt was made to access a device via the Sercos address, but no Sercos address had been configured for this device.
<i>UnidentifiedDeviceAddressingMode</i>	-41	An unsupported method was selected to address a Sercos device (instead of Sercos address or topological address).
<i>ConfigurationChangeNotAllowed</i>	-42	An attempt was made to modify the configuration of a device, but modifications to the configuration are not permissible in the current situation or for the device.
<i>ServiceChannelBusy</i>	-43	Service channel access not possible because the service is busy.
<i>InvalidSercosCycleTime</i>	-44	Selected Sercos cycle time is invalid.
<i>SafeLogicControllerNotFound</i>	-45	fc_safelogicsercoslavehandleget was called with no existing safelogicdevice.
<i>NoDeviceForTopologicalAddressFound</i>	-46	No device found for the specified topological address.
<i>CouldNotAccessCycleTimeViolationCounter</i>	-47	Access to the cycle time violation counter not successful.
<i>CouldNotResetCycleTimeViolationCounter</i>	-48	Reserved
<i>NotConnectedToAxis</i>	-49	An attempt was made to convert user units to increments for a motion device without connected axis.
<i>DriveAxisIsUsedAsMasterForSynchronousMotion</i>	-50	An attempt was made to modify the scaling parameters for a device that is used as the master for synchronous motion.
<i>HomingIsActive</i>	-51	Reserved

Name	Value (INT)	Description
<i>InvalidIncrementResolution</i>	-52	Invalid value for the increment resolution.
<i>ErrorInEncoderCallbackResultDetected</i>	-53	Either the encoder initialization timed out or an encoder error was detected.
<i>EncoderCommunicationError</i>	-54	Error detected in communication with the encoder.
<i>EncoderPowerError</i>	-55	Error detected in encoder power supply.
<i>ExpertIoError</i>	-56	Expert I/O error detected in encoder module.
<i>SercosEncoderInvalidDevice</i>	-57	Encoder initialization for Sercos not successful due to incorrect drive handle.
<i>SercosEncoderAlreadyConfigured</i>	-58	Encoder initialization for Sercos not successful because the encoder was already configured with a different drive.
<i>InvalidFilterValue</i>	-59	The specified filter value is not in the permissible value range.
<i>InvalidDeadTimeValue</i>	-60	The specified deadtime value is not in the permissible value range.
<i>DeviceInfoDoesNotMatchTypePlate</i>	-61	The nameplate of the configured device does not match the nameplate of the device on the Sercos bus.
<i>InvalidStallTorque</i>	-64	An attempt was made to convert the torque from percentage to Nm without a valid stall torque value available for the drive.
<i>FatalException</i>	-100	Unrecoverable error detected. Contact your Schneider Electric service representative.
<i>ProcedureCommandInvalidStateChange</i>	-200	Reserved
<i>ProcedureCommandInvalidErrorCode</i>	-201	A procedure command returned an invalid error code.
<i>ProcedureCommandInvalidCommandDataStatus</i>	-202	The procedure command returned a value that could not be processed in the firmware.
<i>ProcCmdStatusNotSetAndNotEnabled</i>	-221	A procedure command returned the diagnostics value "NotSetAndNotEnabled"
<i>ProcCmdStatusErrorCommandExecImpossible</i>	-222	A procedure command returned the diagnostics value "ErrorCommandExecutionImpossible"
<i>ProcCmdStatusCmdExecInterrupted</i>	-223	A procedure command returned the diagnostics value "CommandExecutionInterrupted"
<i>SvcCmdAccepted</i>	298	The service channel command to be executed was accepted.
<i>TimeOut</i>	-301	The service channel command was not executed due to a timeout.
<i>InterruptedByHighPriorityRequest</i>	-302	The service channel command was aborted by a different command with a higher priority.
<i>NoServiceCommunicationDuringHotPlug</i>	-303	Reserved
<i>DataLengthInvalid</i>	-304	The Sercos stack reported an incorrect data length for a service channel request.
<i>DeviceIndexInvalid</i>	-305	A service channel request was not successful because of an incorrect device index.
<i>AccessInInvalidPhase</i>	-306	An attempt was made to use the service channel during an invalid Sercos communication phase.
<i>SvcBusy</i>	-307	A service channel request was added while the service channel was not able to accept new requests.
<i>SvcQueueFull</i>	-308	A service channel request was added, but the service channel queue was full.
<i>RequestIdNotFound</i>	-309	An attempt was made to get the status of a service channel request that was already processed completely.
<i>GeneralError</i>	-399	A function returned a detected error that could not be identified in the firmware.
<i>AxisInvalid</i>	-402	Invalid axis handle specified for the function.
<i>ConnectingMembersNotDisabled</i>	-430	Reserved
<i>DeviceInvalid</i>	-431	Reserved

Name	Value (INT)	Description
<i>ExistingConnection</i>	-432	An attempt was made to connect an axis to a device to which an axis had already been connected.
<i>SvcNotOpen</i>	-900	Service channel not open.
<i>SvcCloseInvalid</i>	-909	Invalid access to closing the service channel.
<i>IdnInvalid</i>	-1001	IDN not available.
<i>NoAccessToElementIdn</i>	-1009	1009 invalid access to element 1.
<i>NoAccessToElementName</i>	-2001	Name not available.
<i>NamesTooShort</i>	-2002	Name transmission too short.
<i>NamesTooLong</i>	-2003	Name transmission too long.
<i>NamesReadOnly</i>	-2004	Name cannot be modified (read only).
<i>NamesReadOnlyTemporary</i>	-2005	Name is write-protected at this time.
<i>AttributeIsTooShort</i>	-3002	Attribute transmission too short.
<i>AttributeIsTooLong</i>	-3003	Attribute transmission too long.
<i>AttributeIsReadOnly</i>	-3004	Attribute cannot be modified (read only).
<i>AttributeIsReadOnlyTemporary</i>	-3005	Attribute is write-protected at this time.
<i>NoAccessToUnit</i>	-4001	Unit not available.
<i>UnitIsTooShort</i>	-4002	Unit transmission too short.
<i>UnitIsTooLong</i>	-4003	Unit transmission too long.
<i>UnitIsReadOnly</i>	-4004	Unit cannot be modified (read only).
<i>UnitIsReadOnlyTemporary</i>	-4005	Unit is write-protected at this time.
<i>NoAccessToMinimum</i>	-5001	Minimum input value not available.
<i>MinimumIsTooShort</i>	-5002	Minimum input value transmission too short.
<i>MinimumIsTooLong</i>	-5003	Minimum input value transmission too long.
<i>MinimumIsReadOnly</i>	-5004	Minimum input value cannot be modified (read only).
<i>MinimumIsReadOnlyTemporary</i>	-5005	Minimum input value is write-protected at this time.
<i>NoAccessToMaximum</i>	-6001	Maximum input value not available.
<i>MaximumIsTooShort</i>	-6002	Maximum input value transmission too short.
<i>MaximumIsTooLong</i>	-6003	Maximum input value transmission too long.
<i>MaximumIsReadOnly</i>	-6004	Maximum input value cannot be modified (read only).
<i>MaximumIsReadOnlyTemporary</i>	-6005	Maximum input value is write-protected at this time.
<i>OperationDataIsTooShort</i>	-7002	Operation data transmission too short.
<i>OperationDataIsTooLong</i>	-7003	Operation data transmission too long.
<i>OperationDataIsReadOnly</i>	-7004	Operation data cannot be modified (read only).
<i>OperationDataIsReadOnlyTemporary</i>	-7005	Operation data is write-protected during this Sercos communication phase.
<i>OperationDataIsTooSmall</i>	-7006	Operation data is less than the minimum input value.
<i>OperationDataIsTooBig</i>	-7007	Operation data is greater than the maximum input value.
<i>OperationDataInvalid</i>	-7008	Invalid operation data (configured IDN is supported; invalid bit number or bit combination).
<i>OperationDataNeedsPassword</i>	-7009	Operation data is password-protected.
<i>OperationDataIsReadOnlyCyclic</i>	-7010	Operation data is write-protected. It is configured cyclically. The DN is configured in the MDT or AT. Therefore, writing via the service channel is not permissible.
<i>InvalidIndirectAddressing</i>	-7011	Invalid indirect addressing (for example, data container, list handling).

Name	Value (INT)	Description
<i>OperationDataIsSetReadOnly</i>	-7012	Operation data is write-protected due to other settings (for example, parameter, operating mode, sub-device is enabled).
<i>FloatingPointNumberInvalid</i>	-7013	Invalid floating point number.
<i>OperationDataReadOnlyParam</i>	-7014	Operation data is write-protected at parameterization level.
<i>OperationDataReadOnlyLevel</i>	-7015	Operation data is write-protected at operating level.
<i>ProcedureCommandAlreadyActive</i>	-7016	Procedure command already active.
<i>ProcedureCommandNoInterrupt</i>	-7017	Procedure command not interruptible.
<i>ProcedureCommandNoAccessTemp</i>	-7018	Procedure command cannot be executed at this point in time (for example, during this Sercos communication phase, the procedure command cannot be activated).
<i>ProcedureCommandNoExecute</i>	-7019	Procedure command cannot be executed (invalid or incorrect parameters).
<i>ListLengthUnexpected</i>	-7020	The received length of list parameter does not match the expectation (7101 IDN in S-0-0394 not valid).
<i>ListWritingProhibited</i>	-7102	Empty list in S-0-0397 not permissible for write access.
<i>ListIsTooLong</i>	-7103	Maximum length of the list in S-0-0394 is exceeded by list segment.
<i>ListIsReadOnly</i>	-7104	Read access only: The length of the list segment as per list index exceeds the length of the list in S-0-0394.
<i>IdnIsWriteProtected</i>	-7105	IDN in S-0-0394 is write-protected.
<i>OperationDataInListIsTooSmall</i>	-7106	Operation data in list segment is less than the minimum input value.
<i>OperationDataInListIsTooBig</i>	-7107	Operation data in list segment is greater than the maximum input value.
<i>ListIndexInvalid</i>	-7108	Invalid list index in S-0-0395.
<i>ParameterHasNoVariableLength</i>	-7109	Parameter in IDN S-0-0394 does not have variable length.
<i>IdnNotPermittedAsData</i>	-7110	IDN S-0-0397 not permitted as data in S-0-0394.
<i>UnexpectedFeedback</i>	-8000	A function call returned an invalid value.

ET_SercosPhase - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains the possible Sercos communication phases for the Sercos master.

Enumeration Elements

Name	Value (DINT)	Description
<i>NRT</i>	-1	In the state NRT, Sercos communication is disabled.
<i>Phase0</i>	0	Communication phase CP0.
<i>Phase1</i>	1	Communication phase CP1.
<i>Phase2</i>	2	Communication phase CP2.

Name	Value (DINT)	Description
<i>Phase3</i>	3	Communication phase CP3.
<i>Phase4</i>	4	Communication phase CP4.

ET_SercosState - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains Sercos communication phases for the Sercos master and internal states for the Sercos Master Library.

Enumeration Elements

Name	Value (DINT)	Description
<i>NRT</i>	-1	In the state NRT, Sercos communication is disabled.
<i>Phase0</i>	0	Communication phase CP0.
<i>Phase1</i>	1	Communication phase CP1.
<i>Phase2</i>	2	Communication phase CP2.
<i>Phase3</i>	3	Communication phase CP3.
<i>Phase4</i>	4	Communication phase CP4.
<i>Init</i>	10	State PhaseInit (Sercos initialization state is changed after the first call of the realtime cycle; during regular operation, the enumeration does not contain this value)
<i>Error</i>	11	State PhaseError

ET_ServiceChannelAccessingMode - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains types of access for function calls via the service channel.

Enumeration Elements

Name	Value (UINT)	Description
<i>TopoAddress</i>	0	Topological address is used to access the service channel.
<i>SlaveHandle</i>	1	Device handle is used to access the service channel.

ET_SlaveCommunicationState - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains communication states for a Sercos device, depending on the Sercos state.

Enumeration Elements

Name	Value (UINT)	Description
<i>Operational</i>	0	Device is in communication phase 4.
<i>PhasingUp</i>	1	Device is in communication phase 0, 1, 2, or 3.
<i>NoCommunication</i>	2	Device is in communication phase 4, but has not returned a valid state for at least two Sercos cycles, or the Sercos bus is in state Error.
<i>NoData</i>	3	Device is in communication phase 4, but an error has been detected.
<i>NotRunning</i>	4	Sercos is in state NRT.

ET_WorkingMode - General Information

Overview

Type:	Enumeration type
Available as of:	V2.0.108.9437

Description

This enumeration contains settings for the working mode of a Sercos device.

Enumeration Elements

Name	Value (UINT)	Description
<i>Simulated</i>	0	The working mode Simulated is intended for use during development, for example, for testing an application before the drive is connected to the logic/motion controller. The system treats the simulated device like a device that is physically present and connected. However, it neither searches for a simulated device nor does it perform any communication with it. The working mode Simulated is not intended to be used as a virtual axis. To set up a virtual axis (for example, for a virtual shaft as an abstract representation of the machine velocity, or as an intermediate axis for complex master/slave reference value chains), use FB_ControlledAxis (refer to the MotionInterface library guide for details). While the axis of a device in working mode Simulated behaves like a virtual axis, the simulated device also causes an additional performance impact. Further, it is counted towards the maximum number of Sercos devices, which is not the case if you declare a virtual axis by using FB_ControlledAxis.
<i>Activated</i>	1	The working mode Activated is intended for regular machine operation when the device is connected. By default, devices added to the Sercos master tree are activated.

Function Blocks

FB_ReadIDN - General Information

Overview

Type:	Function block
Available as of:	V2.0.108.9437
Inherits from:	-
Implements:	-

Task

Reads an IDN asynchronously so the controller task is not blocked.

Description

This function block reads an IDN on a device in an asynchronous way so that the controller task is not blocked. To be able to process multiple requests in a row, a queue for the service channel requests is used. The maximum number of elements in the queue is 20. Once the queue is full, an attempt to add a new read request results in the error code "Failure_ServiceChannelBusy". New requests are processed in a first-in-first-out order. If a request times out before it is finished, it is removed from the queue and aborted if is active.

First, the access mode is set to either topology or device handle. Once the value at the input *i_xExecute* is TRUE, the asynchronous call is started.

The data context is specified via the four inputs *i_dwParameterIdn*, *i_usParameterElement*, *i_pbDataPointer*, and *i_uiDataLength*. If the request times out as specified via the input *i_timTimeOut* (that is, it is not completed), the value at the output *q_xError* is set to TRUE. This does not affect the value at the input *i_pbDataPointer*. If the request is accepted, the value at the output *q_xActive* is set to TRUE.

Interface

Input	Data type	Description
<i>i_etSVCAccessMode</i>	<i>ET_ServicChannelAccessingMode</i>	Determines whether the topological address or the device handle is used to access the device. Default value: <i>ET_ServicChannelAccessingMode.TopoAddress</i>
<i>i_xExecute</i>	BOOL	Setting the value at this input to TRUE starts the asynchronous call. The outputs of the function block are not reset until the next time this input is set to TRUE.
<i>i_uiTopologicalAddress</i>	UINT	The value at this input identifies the device if the value at the input <i>i_etSVCAccessMode</i> has been set to <i>TopoAddress</i> .
<i>i_stSlave</i>	<i>ST_Slave</i>	The value at this input identifies the device if the value at the input <i>i_etSVCAccessMode</i> has been set to <i>SlaveHandle</i> .
<i>i_dwParameterIdn</i>	DWORD	Specifies the IDN to be read. To generate the value for this input, use <i>SERC.FC_BuildIDN</i> or calculate the value. For additional information, refer to Sercos

Input	Data type	Description
		Parameter Identification Number (IDN) in the SercosCommunication library.
<i>i_usParameterElement</i>	USINT	Specifies the element of the IDN to be read. Default value: 7
<i>i_pbDataPointer</i>	POINTER TO BYTE	Specifies the pointer to where the read data is to be saved.
<i>i_uiDataLength</i>	UINT	Specifies the data length to be read.
<i>i_timTimeOut</i>	LTIME	Specifies the timeout for the request. Initial value: <i>LTIME#1s0ms0us0ns</i>

Output	Data type	Description
<i>q_xDone</i>	BOOL	If this output is set to TRUE, the execution has been completed successfully.
<i>q_xError</i>	BOOL	If the value at this output is TRUE, an error has been detected. See <i>q_diErrorId</i> for details.
<i>q_diErrorId</i>	DINT	Error ID with more information on the detected error. See <i>etResult</i> .
<i>q_xActive</i>	BOOL	If the function block is active, this output is set to TRUE.

Usage of Variables of Type POINTER TO ... or REFERENCE TO ...

The function block provides inputs and/or inputs/outputs of type *POINTER TO...* or *REFERENCE TO...*. With the use of such a pointer or reference, the function block accesses the addressed memory area. In case of an online change event, it may happen that memory areas are moved to new addresses, and, in consequence, a pointer or reference becomes invalid. To help avoid errors associated with invalid pointers, variables of type *POINTER TO...* or *REFERENCE TO...* must be updated cyclically or at least at the beginning of the cycle in which they are used.

⚠ CAUTION

INVALID POINTER

Do not use the “Online Change” command or the “Log in with online change” option as long as one of the function blocks of this library indicates *Active* in your running application.

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

FB_WriteIDN - General Information

Overview

Type:	Function block
Available as of:	V2.0.108.9437
Inherits from:	-
Implements:	-

Task

Writes an IDN asynchronously so the controller task is not blocked.

Description

This function block writes an IDN to a device in an asynchronous way so that the controller task is not blocked. To be able to process multiple requests in a row, a queue for the service channel requests is used. The maximum number of elements in the queue is 20. Once the queue is full, an attempt to add a new write request results in the error code "Failure_ServiceChannelBusy". New requests are processed in a first-in-first-out order. If a request times out before it is finished, it is removed from the queue and aborted if it is active.

First, the access mode is set to either topology or device handle. Once the value at the input *i_xExecute* is TRUE, the asynchronous call is started.

The data context is specified via the four inputs *i_dwParameterIdn*, *i_usParameterElement*, *i_pbDataPointer*, and *i_uiDataLength*. If the request times out as specified via the input *i_timTimeOut* (that is, it is not completed), the value at the output *q_xError* is set to TRUE. This does not affect the value at the input *i_pbDataPointer*. If the request is accepted, the value at the output *q_xActive* is set to TRUE.

Interface

Input	Data type	Description
<i>i_etSVCAccessMode</i>	<i>ET_ServicChannelAccessingMode</i>	Determines whether the topological address or the device handle is used to access the device. Default value: <i>ET_ServicChannelAccessingMode.TopoAddress</i>
<i>i_xExecute</i>	BOOL	Setting the value at this input to TRUE starts the asynchronous call. The outputs of the function block are not reset until the next time this input is set to TRUE.
<i>i_uiTopologicalAddress</i>	UINT	The value at this input identifies the device if the value at the input <i>i_etSVCAccessMode</i> has been set to <i>TopoAddress</i> .
<i>i_stSlave</i>	<i>ST_Slave</i>	The value at this input identifies the device if the value at the input <i>i_etSVCAccessMode</i> has been set to <i>SlaveHandle</i> .
<i>i_dwParameterIdn</i>	DWORD	Specifies the IDN to be written. To generate the value for this input, use <i>SERC.FC_BuildIDN</i> or calculate the value. For additional information, refer to Sercos Parameter Identification Number (IDN) in the SercosCommunication library.
<i>i_pbDataPointer</i>	POINTER TO BYTE	Specifies the pointer to where the write data is to be saved.
<i>i_uiDataLength</i>	UINT	Specifies the data length to be written.
<i>i_timTimeOut</i>	LTIME	Specifies the timeout for the request. Initial value: <i>LTIME#1s0ms0us0ns</i>

Output	Data type	Description
<i>q_xDone</i>	BOOL	If this output is set to TRUE, the execution has been completed successfully.
<i>q_xError</i>	BOOL	If the value at this output is TRUE, an error has been detected. See <i>q_diErrorId</i> for details.
<i>q_diErrorId</i>	DINT	Error ID with more information on the detected error. See <i>etResult</i> .
<i>q_xActive</i>	BOOL	If the function block is active, this output is set to TRUE.

Usage of Variables of Type **POINTER TO ...** or **REFERENCE TO ...**

The function block provides inputs and/or inputs/outputs of type *POINTER TO...* or *REFERENCE TO...*. With the use of such a pointer or reference, the function block accesses the addressed memory area. In case of an online change event, it may happen that memory areas are moved to new addresses, and, in consequence, a pointer or reference becomes invalid. To help avoid errors associated with invalid pointers, variables of type *POINTER TO...* or *REFERENCE TO...* must be updated cyclically or at least at the beginning of the cycle in which they are used.

⚠ CAUTION

INVALID POINTER

Do not use the “Online Change” command or the “Log in with online change” option as long as one of the function blocks of this library indicates *Active* in your running application.

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

Functions

FC_EtResultToString - General Information

Overview

Type:	Function
Available as of:	V2.0.108.9437
Inherits from:	-
Implements:	-

Task

Converts an enumeration element of type *ET_Result* to a variable of type STRING.

Functional Description

Using the function *FC_EtResultToString*, you can convert an enumeration element of type *ET_Result* to a variable of type STRING.

Interface

Input	Data type	Description
<i>i_etResult</i>	ET_Result	Enumeration with the result.

Output	Data type	Description
<i>q_xError</i>	BOOL	Output is set to TRUE if an error is detected during the conversion.
<i>q_etResult</i>	ET_Result	Output is set to <i>InvalidResult</i> if an error is detected during the conversion.

Return Value

Data type	Description
STRING(80)	The <i>ET_Result</i> value converted to text.

FC_EtSercosPhaseToString - General Information

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Converts an enumeration element of type *ET_SercosPhase* to a variable of type STRING.

Functional Description

Using the function *FC_EtSercosPhaseToString*, you can convert an enumeration element of type *ET_SercosPhase* to a variable of type STRING.

Interface

Input	Data type	Description
<i>i_etSercosPhase</i>	<i>ET_SercosPhase</i>	Enumeration with the Sercos communication phase to be converted.

Output	Data type	Description
<i>q_xError</i>	BOOL	Output is set to TRUE if an error is detected during the conversion.
<i>q_etResult</i>	<i>ET_Result</i>	Output is set to <i>ET_Result.InvalidSercosPhase</i> if an error is detected during the conversion.

Return Value

Data type	Description
STRING[80]	The <i>ET_SercosPhase</i> value converted to text.

FC_EtSercosStateToString - General Information

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Converts an enumeration element of type *ET_SercosState* to a variable of type STRING.

Functional Description

Using the function *FC_EtSercosStateToString*, you can convert an enumeration element of type *ET_SercosState* to a variable of type STRING.

Interface

Input	Data type	Description
<i>i_etSercosState</i>	<i>ET_SercosState</i>	Enumeration with the Sercos state to be converted.

Output	Data type	Description
<i>q_xError</i>	BOOL	Output is set to TRUE if an error is detected during the conversion.
<i>q_etResult</i>	<i>ET_Result</i>	Output is set to <i>ET_Result.InvalidSercosState</i> if an error is detected during the conversion.

Return Value

Data type	Description
STRING[80]	The <i>ET_SercosState</i> value converted to text.

FC_SlaveGetCommunicationState - General Information

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the communication state of a Sercos device (slave).

Functional Description

Using the function *FC_SlaveGetCommunicationState*, you get the communication state of a Sercos device, based on the current Sercos communication phase and the state of the device.

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Sercos device the communication state is requested for.

Output	Data type	Description
<i>q_etSlaveCommunication-State</i>	<i>ET_SlaveCommunication-State</i>	Communication state of the Sercos device.

Return Value

Data type	Description
STRING[80]	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

Internal Functions

FC_DriveGetError

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns TRUE if a drive error has been detected.

Functional Description

This function returns TRUE if a drive error has been detected (connection control error or the status word of the drive indicates a detected error). If no error is detected, the function returns FALSE. The function can be used in Sercos communication phase 4.

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Topological address of the device for which the function is executed.

Output	Data type	Description
<i>q_xError</i>	BOOL	TRUE if a connection control error is detected or the status word of the drive indicates a detected error.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_GetIdleTimeOnSercosInLastCycle

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the Sercos idle time for the last cycle in microseconds

Functional Description

Returns the idle time in microseconds of the last completed Sercos cycle. For example, if the Sercos cycle time is 2 ms and the motion task finished 0.5 ms into

the cycle, then this function returns 1500 (2000 microseconds - 500 microseconds = 1500 microseconds).

Return Value

Data type	Description
LREAL	Idle time in microseconds

FC_GetMotionCycleTaskLoadOfLastCycle

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the task load of the motion cycle for the last cycle as a percentage.

Functional Description

Returns the percentage of the Sercos cycle time that is taken up by processing time of the motion task. For example, if the Sercos cycle time is 2 ms and the motion task took 0.5 ms to execute, then this function returns 25 ($0.5 \text{ ms} / 2 \text{ ms} = 0.25 = 25 \%$).

Return Value

Data type	Description
LREAL	Task load of motion cycle as a percentage.

FC_GetPhysicalConnectionState

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the physical connection state of a device

Functional Description

This function provides the physical connection state of a Sercos device (slave). The state indicates whether or not the firmware was able match a configured device to a device connected to the Sercos bus.

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device for which the physical connection state is requested.

Output	Data type	Description
<i>q_etPhysicalConnectionState</i>	<i>ET_PhysicalConnectionState</i>	Physical connection state of the device.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_GetScaledFeedbackAcceleration

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the scaled acceleration of a drive.

Functional Description

The function returns the acceleration of the drive scaled in the user units. The value is not modified by a *setPos*.

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device whose scaled acceleration is requested.

Output	Data type	Description
<i>q_lrAcceleration</i>	LREAL	Scaled acceleration of the device.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_GetScaledFeedbackVelocity

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the scaled velocity of a drive.

Functional Description

The function returns the velocity of the drive scaled in user units. The value is not modified by a *setPos*

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device whose scaled velocity is requested.

Output	Data type	Description
<i>q_lrVelocity</i>	LREAL	Scaled velocity of the device.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_ReadPositionFeedbackValue

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the last position feedback value from a device.

Functional Description

The function returns the last position reported by a device (taken from the corresponding IDN).

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device whose position is requested.

Output	Data type	Description
<i>q_diPosition</i>	DINT	Position of the device.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_ReadScaledPositionFeedback

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the feedback position of a device, scaled in user units.

Functional Description

The function returns the feedback position of a drive scaled in user units. If the axis is a modulo axis, the scaled position changes if the reference position reaches its modulo overflow. The feedback position is affected by homing and position setting (*home*, *setPos*).

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device whose scaled position is requested.

Output	Data type	Description
<i>q_lrPosition</i>	LREAL	Scaled position of the device.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_ResetDiagnostic

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Resets the detected device error or the detected connection control error.

Functional Description

Calling *FC_ResetDiagnostic* re-synchronizes connection control if a connection control had been detected. If a device error had been detected, calling *FC_ResetDiagnostic* starts a procedure command S-0-99.

Interface

Input	Data type	Description
<i>i_stSlave</i>	<i>ST_Slave</i>	Device handle of the device to be reset with <i>FC_ResetDiagnostic</i> .

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_SercosGetConfiguration

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the Sercos configuration in form of a structure.

Functional Description

Returns the following information as a structure:

- *uiNumberOfEntries*: Total number of devices (slaves)
- *uiNumberOfPhysicalDevices*: Number of physical devices (slaves)
- *uiPhaseRunUpCount*: 0 (reserved)
- *iCurrentPhase*: Sercos communication phase
- *astDevices*: ARRAY [0..254] OF *ST_SercosConfigurationDevice*: Nameplate data and topology address

Interface

Input / Output	Data type	Description
<i>iq_stSercosConfiguration</i>	<i>ST_SercosConfiguration</i>	Structure to return Sercos configuration data.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_SercosGetCycleCount

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the number of calculation cycles since the first time the device (its board) was started.

Functional Description

Each calculation cycle increases a counter by 1 (starting with 0). As this cycle is linked to the Sercos cycle, it is possible to get the number of Sercos cycles since the first time the device was started.

Interface

Output	Data type	Description
<i>q_diSercosCycleCount</i>	DINT	Number of Sercos cycles since the first time the device was started

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

FC_SercosGetSlaveCount

Overview

Type:	Function
Available as of:	V2.0.108.9437

Task

Returns the number of the physical Sercos devices (slaves) on the Sercos bus.

Functional Description

During phase-up, the Sercos devices (slaves) connected to the Sercos bus are counted. This function returns the number of connected physical Sercos devices (slaves).

Interface

Output	Data type	Description
<i>q_dSercosSlaveCount</i>	DINT	Number of physical Sercos devices (slaves) connected to the Sercos bus.

Return Value

Data type	Description
DINT	Number that indicates the result of the function call (see <i>DeviceApiResult</i>).

Structures

ST_SercosConfiguration - General Information

Overview

Type:	Data structure
Available as of:	V2.0.108.9437
Inherits from:	-

Description

Structure for basic Sercos configuration information.

Structure Elements

Variable	Data type	Description
<i>uiNumberOfEntries</i>	UINT	Total number of devices (slaves)
<i>uiNumberOfPhysicalDe-</i> <i>vices</i>	UINT	Number of physical devices (slaves)
<i>iCurrentPhase</i>	DINT	Sercos communication phase
<i>astDevices</i>	ARRAY [0..254] OF <i>ST_</i> <i>SercosConfigurationDevice</i>	Array with nameplate data and topology address of each device

ST_SercosConfigurationDevice - General Information

Overview

Type:	Data structure
Available as of:	V2.0.108.9437
Inherits from:	-

Description

Structure for Sercos configuration data of an individual device.

Structure Elements

Variable	Data type	Description
<i>stTypePlate</i>	<i>ST_TypePlate</i>	Nameplate data of the device
<i>uiTopologyAddress</i>	UINT	Topological address of the device

ST_SercosTime - General Information

Overview

Type:	Data structure
Available as of:	V2.0.108.9437
Inherits from:	-

Description

Structure for seconds and nanoseconds for a specific Sercos time.

Structure Elements

Variable	Data type	Description
<i>udiSeconds</i>	UDINT	Seconds of Sercos time
<i>udiNanoseconds</i>	UDINT	Nanoseconds of Sercos time

ST_Slave - General Information

Overview

Type:	Data structure
Available as of:	V2.0.108.9437
Inherits from:	-

Description

Structure that contains a handle for a Sercos device (slave).

Structure Elements

Variable	Data type	Description
<i>hSlave</i>	UDINT	Handle for a Sercos device <i>RTS_IEC_HANDLE</i>

ST_TypePlate - General Information

Overview

Type:	Data structure
Available as of:	V2.0.108.9437
Inherits from:	-

Description

Structure for nameplate data of a Sercos device.

Structure Elements

Variable	Data type	Description
<i>sFirmwareVersion</i>	STRING[80]	Firmware version of the device
<i>sHardwareVersion</i>	STRING[80]	Hardware version of the device
<i>sDeviceSerialNumber</i>	STRING[80]	Serial number of the device
<i>sDeviceName</i>	STRING[80]	Name of the device
<i>sOrderNumber</i>	STRING[80]	Order number of the device

Variable	Data type	Description
<i>sFirmwareLoaderRevision</i>	STRING[80]	Firmware loader revision of the device
<i>wVendorCode</i>	WORD	Vendor code of the device
<i>sVendorDeviceId</i>	STRING[80]	Vendor device ID of the device
<i>uiSercosAddress</i>	UINT	Sercos address of the device
<i>sApplicationType</i>	STRING[80]	Application type of the device

Global Elements

Global Constants List (GCL)

Overview

Type:	Global constants
Available as of:	V2.0.108.9437

Description

The global constants list contains the global constants of the SercosMaster library.

Global Constants

Variable	Data type	Value	Description
<i>Gc_uiMaxNumberOfConfigurableIDNsFor-CyclicCommunication</i>	UINT	64	Maximum number of IDNs that can be mapped cyclically.
<i>Gc_uiMaxNumberOfConfiguredPhysical-Devices</i>	UINT	254	Maximum number of physical devices that can be configured.
<i>Gc_uiMaxPayloadSizePerConnection</i>	UINT	1492	Maximum payload size for a connection.

Index

E

ET_IdentificationMode	13
ET_IpConfigMode	13
ET_OperationModeStatus	13
ET_PhysicalConnectionState	14
ET_Result	14
ET_SercosPhase	19
ET_SercosState	20
ET_ServiceChannelAccessingMode	20
ET_SlaveCommunicationState	21
ET_WorkingMode	21

F

FB_ReadIDN	23
FB_WriteIDN	24
FC_DriveGetError	30
FC_EtResultToString	27
FC_EtSercosPhaseToString	27
FC_EtSercosStateToString	28
FC_GetIdleTimeOnSercosInLastCycle	30
FC_GetMotionCycleTaskLoadOfLastCycle	31
FC_GetPhysicalConnectionState	31
FC_GetScaledFeedbackAcceleration	32
FC_GetScaledFeedbackVelocity	33
FC_ReadPositionFeedbackValue	33
FC_ReadScaledPositionFeedback	34
FC_ResetDiagnostic	34
FC_SercosGetConfiguration	35
FC_SercosGetCycleCount	36
FC_SercosGetSlaveCount	36
FC_SlaveGetCommunicationState	29

G

GCL (Global Constants List)	
SercosMaster	41

S

SercosMaster	
ET_IdentificationMode	13
ET_IpConfigMode	13
ET_OperationModeStatus	13
ET_PhysicalConnectionState	14
ET_Result	14
ET_SercosPhase	19
ET_SercosState	20
ET_ServiceChannelAccessingMode	20
ET_SlaveCommunicationState	21
ET_WorkingMode	21
FB_ReadIDN	23
FB_WriteIDN	24
FC_EtResultToString	27
FC_EtSercosPhaseToString	27
FC_EtSercosStateToString	28
FC_SlaveGetCommunicationState	29
GCL (Global Constants List)	41
ST_SercosConfiguration	38
ST_SercosConfigurationDevice	38
ST_SercosTime	38
ST_Slave	39
ST_TypePlate	39

ST_SercosConfiguration	38
ST_SercosConfigurationDevice	38
ST_SercosTime	38
ST_Slave	39
ST_TypePlate	39

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2021 – Schneider Electric. All rights reserved.

EIO0000004624.00

Modicon M262

Synchronized Motion Control

Library Guide

EIO0000003871.03
12/2021

Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2021 Schneider Electric. All Rights Reserved.

Table of Contents

Safety Information.....	5
Before You Begin.....	5
Start-up and Test.....	6
Operation and Adjustments	7
About the Book.....	8
General Description of Motion Control Libraries	11
General Description	11
Library and Related Function Blocks	11
Device Integration of Sercos Devices	12
Motion Control	14
PLCopen State Diagram.....	17
Exception Handling When Function Block Signals Error	17
Real-Time, Synchronous External Task	18
Library-Specific Data Types.....	21
Common Inputs and Outputs	26
Behavior of Function Blocks with the Input <i>Execute</i>	26
Enumerations	28
<i>ET_Result</i> - General Information.....	28
Function Blocks - Single Axis.....	34
<i>MC_AbortTrigger</i>	34
<i>MC_CustomJob</i>	36
<i>MC_Halt</i>	39
<i>MC_Home</i>	42
<i>MC_MoveAbsolute</i>	44
<i>MC_MoveAdditive</i>	47
<i>MC_MoveRelative</i>	50
<i>MC_MoveSuperImposed</i>	53
<i>MC_MoveVelocity</i>	56
<i>MC_Power</i>	59
<i>MC_Reset</i>	61
<i>MC_SetPosition</i>	63
<i>MC_Stop</i>	65
<i>MC_TorqueControl</i>	67
<i>MC_TouchProbe</i>	70
Function Blocks - Multi Axis.....	72
<i>MC_CamIn</i>	72
<i>MC_GearIn</i>	79
<i>MC_PhasingAbsolute</i>	81
Migration Information SoftMotion to PLCopen	83
Migration Information SoftMotion to PLCopen	83
Glossary	87
Index	93

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

Before You Begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠ WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

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

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

Start-up and Test

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check are made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

Operation and Adjustments

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

Document Scope

The function blocks of the library described in the present library guide are used under the EcoStruxure Machine Expert software environment to control LXM32S drives with M262.

The function blocks contained and enabled by the library are compliant with the IEC 61131-3 standard.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.2.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Related Documents

Title of documentation	Reference number
EcoStruxure Machine Expert - Programming Guide	EIO0000002854 (eng) EIO0000002855 (fre) EIO0000002856 (ger) EIO0000002858 (spa) EIO0000002857 (ita) EIO0000002859 (chi)
EcoStruxure Machine Expert - Functions and Libraries User Guide	EIO0000002829 (eng) EIO0000002830 (fre) EIO0000002831 (ger) EIO0000002833 (spa) EIO0000002832 (ita) EIO0000002834 (chi)
Modicon M262 Logic/Motion Controller - Programming Guide	EIO0000003651 (eng) EIO0000003652 (fra) EIO0000003653 (ger) EIO0000003654 (spa) EIO0000003655 (ita) EIO0000003656 (chi) EIO0000003657 (por) EIO0000003658 (tur)

Title of documentation	Reference number
Modicon M262 Logic/Motion Controller - Hardware Guide	EIO0000003659 (eng)
	EIO0000003660 (fre)
	EIO0000003661 (ger)
	EIO0000003662 (spa)
	EIO0000003663 (ita)
	EIO0000003664 (chi)
	EIO0000003665 (por)
	EIO0000003666 (tur)
LXM32S Servo Drive - User Guide	0198441114060 (eng)
	0198441114061 (fre)
	0198441114059 (ger)
	0198441114063 (spa)
	0198441114062 (ita)
	0198441114064 (chi)

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

General Description of Motion Control Libraries

General Description

Library and Related Function Blocks

Library Overview

Library:

- Library name: **PLCopen MC part 1**
Namespace: **PLCO**
- Library name: **MotionInterface**
Namespace: **MOIN**

The library PLCopen MC part 1 largely complies with the specifications of PLCopen Motion Control Part 1, Version 2.0. Deviations from or additions to the specifications are indicated in the present document.

Supported Controllers

The library supports the following devices:

- M262M05
- M262M15
- M262M25
- M262M35
- Connected Sercos devices (refer to the user guide of your device)

Function Blocks

Category	Function block	Brief Description
Single axis	<i>MC_AbortTrigger</i>	This function block terminates position capture.
	<i>MC_CustomJob</i>	This function block allows you to control an axis by a custom algorithm that calculates cyclic set position, velocity, and acceleration of the axis.
	<i>MC_Halt</i>	This function block stops the ongoing movement. The function block can be aborted by other function blocks. See <i>MC_Stop</i> for a stop that cannot be aborted.
	<i>MC_Home</i>	This function block homes the drive with the homing-specific settings of the drive.
	<i>MC_MoveAbsolute</i>	This function block performs a movement to a specified absolute target position.
	<i>MC_MoveAdditive</i>	This function block performs a movement with a specified distance with reference to the previous target position.
	<i>MC_MoveRelative</i>	This function block performs a movement with a specified distance with reference to the actual position.
	<i>MC_MoveSuperImposed</i>	This function block performs a superimposed movement with a specified position offset with reference to the position of an ongoing movement.
	<i>MC_MoveVelocity</i>	This function block performs a movement with a specified target velocity.
	<i>MC_Power</i>	This function block enables or disables the power stage of the drive.
	<i>MC_Reset</i>	This function block acknowledges detected axis-related errors and drive-related errors.
	<i>MC_SetPosition</i>	This function block sets a position value at the actual position of the motor to define the zero point.

Category	Function block	Brief Description
	<i>MC_Stop</i>	This function block stops the ongoing movement. No other movements can be started as long as this function block is active. See <i>MC_Halt</i> for a stop that can be aborted.
	<i>MC_TorqueControl</i>	This function block allows you to operate a drive in the operating mode Cyclic Synchronous Torque (CST).
	<i>MC_TouchProbe</i>	This function block configures and starts position capture.
Multi axis	<i>MC_CamIn</i>	This function block activates master-slave coupling with the profile for an electronic cam specified in a cam table.
	<i>MC_GearIn</i>	This function block activates coupling of a master axis and a slave axis with a given gear factor between the position or velocity of the master axis and the slave axis, depending on the operating mode.
	<i>MC_PhasingAbsolute</i>	This function block creates a position offset between the position of a master axis and the position of this master axis as it is seen by the slave axis.

Device Integration of Sercos Devices

Overview

With the Modicon M262 Motion Controller in EcoStruxure Machine Expert, functionalities of devices are grouped and standardized across different devices. For example, Sercos slave devices have an Identification function which is represented by a function block and which implements the same interface and has the same parameters and properties for any device object.

In addition, some of these grouped functions can be disabled. For example, if not needed by the application, the touch-probe capture functionality of a Lexium 32S can be unchecked. This removes the functionality from the controller application and facilitates programming (unused functionality is not even shown by Intellisense). In addition, it slightly increases performance during build (less code to compile) and run-time (smaller application).

Accessing Device Objects in the Controller Application

The most efficient way to access device objects is through Interfaces. The POU types of the device object (with the prefix *FB_DI_*) should not be used directly by the application.

This has no impact when you access the properties of a device. For example, calling *DRV_X.Identification.ConfiguredSercosAddress* has the expected results.

However, if you want to build a reusable module (which takes a device object as an input), this new device integration concept provides more flexibility, but also requires structural decisions and adherence to a number of rules.

Do not declare any *VAR* or *VAR_INPUT*, or *VAR_IN_OUT* of any concrete device object type (the one with the prefix *FB_DI_*). The compiler automatically keeps you from accidentally making assignments by value of these types.

The following list provides a number of general rules and considerations regarding device objects:

- Do not use concrete device types.

In your controller application, you can unselect unused device functionalities. This decision is made on a per-device-instance basis, which means if there are two Lexium 32S devices in your application project, they may offer different functionalities and thus they would not be represented by the same POU type.

When declaring a variable that holds a reference to a device object, create a reusable module that can work with any device object, or with any drive object, etc. Therefore, use a type that can represent any device or any drive, regardless of the optional functionalities it has.

- Reference devices by using *IF_DeviceAccess*.

Reference devices by declaring a *VAR* or *VAR_IN* of type *IF_DeviceAccess*. Any device object can be assigned to this variable (indeed any device object: it can hold a drive device object, a Safety Logic Controller (SLC) device object, or even the Sercos master device object). The assignment is automatically an assignment by reference because *IDeviceObject* is an interface.

- Use the Codesys operator *__QUERYINTERFACE* to obtain information on device functionality because *IF_DeviceAccess* cannot provide information on device functions.

IF_DeviceAccess does not show any of the functions of the device. *IF_DeviceAccess* is essentially an empty interface. Because *IF_DeviceAccess* can represent any device, the compiler is not aware of the functionalities of this device object. You may want to write a reusable module that can operate on any Sercos slave device (and therefore only uses the identification of the device object), or you may want to write a reusable module that can work on any drive (and uses the axis of the device object), or you may want to create a piece of code that does different things on different devices, depending on which device is given (for example your module may provide additional information: the safety-related project information for an SLC device object and the Copla firmware version an LXM32S drive device object). To allow for these degrees of freedom, you need to explicitly test whether a function is available before accessing it. This test is done by calling *__QUERYINTERFACE* to test whether a certain specific functional interface is available. For example, if a device object implements *IF_Trigger1Access*, the device object provides the trigger capture 1 which can be accessed via the property *triggerCap1* (which is a property of *IF_Trigger1Access*).

- Use logical functions if you do not use the device

Example: You want to create a reusable module for a touch-probe capture input: You could declare a variable of type *IF_DeviceAccess* and then test it (via *__QUERYINTERFACE*) for *IF_Trigger1Access* only to access *MC_TriggerRef*.

An LXM32S provides three triggers with three separate interfaces (*IF_Trigger1Access*, *IF_Trigger2Access*, *IF_Trigger3Access*). You may want to specify in the application which of these three capture inputs is to be used by the module you are creating. You could hard-code that your module always uses trigger 1, but this is not flexible. It would also be possible to add a separate input that specifies which of the triggers to be used.

However, the most efficient and simple solution is to give your module an input of type *MC_TriggerRef* (which already refers to one specific trigger input on one specific device instance). When you call the module from your application, you assign *DRV_X.triggerCap1* to this input. This way, *IF_DeviceAccess*, or *__QUERYINTERFACE* do not have to be used at all.

Working Modes of Device Objects

Slave device objects can have the following “working modes”:

- **Activated**
- **Simulated**

The working mode **Activated** is intended for regular machine operation when the device is connected. By default, slaves added to the Sercos master tree are activated.

The working mode **Simulated** is intended for use during development, for example, for testing an application before the drive is connected to the motion controller. The system treats the simulated device like a device that is physically present and connected. However, it neither searches for a simulated device nor does it perform any communication with it.

The working mode **Simulated** is not intended to be used as a virtual axis. To set up a “pure” virtual axis (for example, for a virtual shaft as an abstract representation of the machine velocity, or as an intermediate axis for complex

master/slave reference value chains), use *FB_ControlledAxis* (refer to the MotionInterface library guide for details).

While the axis of a device in working mode **Simulated** behaves like a virtual axis, the simulated device also causes an unnecessary performance impact. In addition, it is counted towards the maximum number of Sercos devices, which is not the case if you declare a “pure” virtual axis by using *FB_ControlledAxis*.

Motion Control

Task Concept

The controller runs the user application in which motion control function blocks are called in a task separate from the real-time motion task in which motion profiles are calculated and Sercos communication takes place. These two tasks can have different cycle times. The cycle time of the user application task is typically 10 ms, but shorter cycle times down to 1 ms are possible. The real-time task operates with cycle times of 1 ms, 2 ms, or 4 ms, depending on the Sercos cycle time and the machine configuration (controller type, number of axes, etc.).

This separation provides a number of benefits in terms of increased performance. The user application can be run as slowly as required, for example, to allow for file system operations or network communication. Event handling, decision making, and command processing of the application (typically less time-critical) can be performed in this potentially slower task. However, modifications to the motion profile can be executed at maximum speed by executing and buffering the function blocks ahead of time.

This proven approach is also used by Schneider Electric PacDrive systems.

Programmers familiar with systems such as SoftMotion (which executes the user application in the real-time process) and standard PLCopen programming need to reconsider a number of common practices to take this approach to its full potential. A typical approach of PLCopen programming is to wait for a function block to “finish” (for example, outputs *Done*, *inVelocity*, *EndOfProfile*, *InGear*) and use the corresponding signals to execute the next function block.

To achieve the expected movement without sudden jumps (especially with velocity and/or acceleration values not equal to zero at the end of one motion job), function blocks must be executed ahead of time with *MC_Buffer_Mode* set to *Buffered*. This way, the motion profile of a function block becomes active in the same real-time cycle in which a preceding function block finishes.

If you trigger the start (*Execute*) of the subsequent function block via, for example, the *EndOfProfile* signal of the preceding function block instead of buffering it, one or several cycles of delay result in which no function block is active and the axis remains at a standstill until the next application task cycle starts the subsequent function block.

Axis_Ref

As defined by PLCopen, an *Axis_Ref* is used as an input to motion control function blocks to specify the axis to be used as the master axis or slave axis for that function block.

The Modicon M262 Motion Controller provides three types of implementations for the *Axis_Ref*:

- Drive axis

A drive axis provided by, for example, the LXM32S drive or by the generic Sercos Drive device object. It can be accessed via, for example, *DRV_X.Axis*. A drive axis can be used as a master axis or as a slave axis for any function block with the corresponding input. The position, velocity and/or acceleration values of a drive axis are usually target values. The only exception is in situations in which the Modicon M262 Motion Controller does not control the movement of the axis (for example, during homing, during a stop performed autonomously by the drive, or when the drive power stage is disabled). In such conditions, the position of the axis is the feedback position and the

velocity/acceleration are calculated based on the feedback position. A slave axis can still follow a master axis even when the master axis is not currently controlled by the Modicon M262 Motion Controller.

- Virtual axis

A virtual axis represents, for example, a virtual machine shaft, or any other intermediate axis for complex relationships between master axes and slave axes. A virtual axis can be created by declaring a **VAR** of type *FB_ControlledAxis* anywhere in the application. A virtual axis can be used as a master axis or as a slave axis for any function block with the corresponding input.

The Modicon M262 Motion Controller does not limit the number of virtual axes. However, take into account that a virtual axis has a performance impact on the Modicon M262 Motion Controller that is comparable to that of a drive axis.

- Encoder input axis

An encoder input axis is provided by the on-board encoder input of the Modicon M262 Motion Controller. An encoder input axis can be used as a master axis only, not as a slave axis. The feedback values provided by the encoder are used as master axis position.

When a master axis becomes unavailable (for example, encoder is disconnected or inoperative), any slave axis following this master transitions to the PLCopen operating state ErrorStop. If you want to have slaves stop in a synchronous way in such a situation, implement an intermediate virtual axis between the axes.

If the master axis becomes unavailable, the intermediate virtual axis transitions to the PLCopen operating state ErrorStop. The axes following the intermediate axis remain in the PLCopen operating state SynchronizedMotion and continue to follow the intermediate virtual axis during its deceleration ramp.

Axis Configuration

The Modicon M262 Motion Controller supports both axis types defined by PLCopen:

- Modulo axis
- Linear/finite axis (with or without movement limits)

An axis is configured by calling the corresponding method of *Axis_Ref*.

The modulo of an axis defined as modulo axis is observed regardless of the job or axis state (for example, even while the axis is disabled).

There is not necessarily a relation between the axis modulo and the cam application period (neither in X nor in Y direction), even if the cam is started in absolute mode. Refer to the description of the [function block](#), [page 72](#) for details.

The reference position and the actual position are on the same side of the modulo jump. Therefore, for example, in the case of an axis moving forward with modulo 360, the actual position can have a value of -2 (instead of 358) when the reference position is 1.

Absolute Position, Homing, and Absolute Movements

After a power cycle, a drive axis or an encoder axis usually restores its absolute position from the stored encoder position. This is done by multiplying the encoder position with the scaling of the drive and relating the value to the modulo position, if a modulo axis is used. However, the controller cannot verify that this restored absolute position is correct. The position may not be correct if, for example, the mechanical position has been modified (for example, motor, encoder, and/or gearbox have been replaced, or manual movements have been made during power off). In addition, the movement range of the axis must be mechanically restricted to be within less than one encoder overflow. Also, the gear ratio of a mechanical gearbox and/or the scaling factor for the drive must be selected in such a way that the encoder position of the encoder period can be resolved to an integer multiple of the application period.

To help avoid conditions with incorrect absolute positions, the Modicon M262 Motion Controller restores the absolute position, but does not automatically consider the axis or axes to be homed. An axis is only considered to be homed after the function block *MC_Home* or the function block *MC_SetPosition* (with *Relative* = FALSE) have been executed successfully or the flag *isHomed* has been set to TRUE by the controller application.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the axis has been homed properly after each power cycle and after each manual intervention before performing any other movements.
- Verify that the axis has been homed properly before performing any type of absolute movements.
- Do not set the flag *isHomed* to TRUE in the controller application unless you have positively ensured and confirmed by appropriate means that the restored absolute position is correct.
- Fully read and understand all pertinent documentation of all software and equipment used in your application/process.

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

Movement Range and Position Calculation With Floating-Point Numbers

The Modicon M262 Motion Controller uses floating-point numbers for absolute axis positions. It is inherent to floating-point numbers that the resolution (in absolute terms) decreases the further away the numbers are from zero. The position of a drive, on the other hand, is represented by integer numbers (encoder increments) so that the resolution is always the same, regardless of how far away the number is from zero. Over time, this leads to reduced control accuracy and ultimately to a stop of the axis as a result of a detected error even if the floating point resolution is still sufficient.

⚠ WARNING

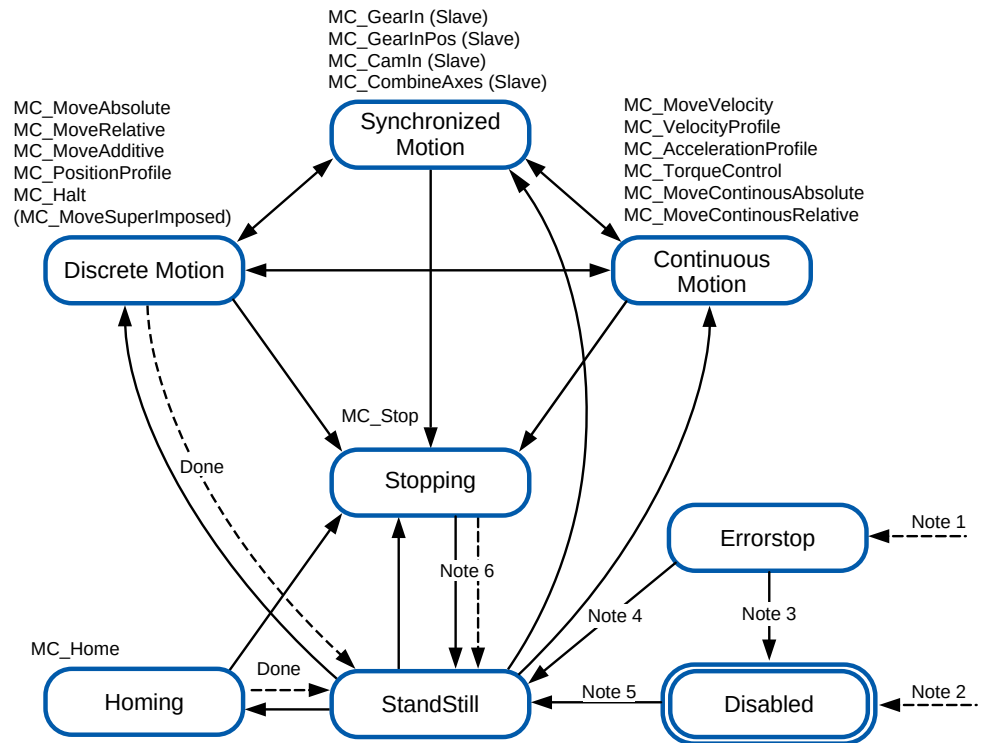
UNINTENDED EQUIPMENT OPERATION

- Use the axis type modulo for an axis that moves in only one direction and continuously increments its position, such as a conveyor axis.
- Use the axis type linear/finite only for axes with physically limited movement ranges, for example, by means of limit switches.

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

PLCopen State Diagram

The following diagram provides a general overview of the PLCOpen state machine. Some of the function blocks shown are not implemented by the library PLCopen Motion Control Part 1, Version 2.0. At any given point in time, the axis is in exactly one state. If a function block is executed or an error is detected, this may cause a state transition. The axis state is available via the property *etAxisState* of each *Axis_Ref*.



Note 1 An error has been detected (transition from any state).

Note 2 The input *Enable* of the function block *MC_Power* is set to FALSE and no error has been detected (transition from any state).

Note 3 *MC_Reset* and *MC_Power.Status* = FALSE.

Note 4 *MC_Reset* and *MC_Power.Status* = TRUE and *MC_Power.Enable* = TRUE.

Note 5 *MC_Power.Enable* = TRUE and *MC_Power.Status* = TRUE.

Note 6 *MC_Stop.Done* = TRUE and *MC_Stop.Execute* = FALSE.

Exception Handling When Function Block Signals Error

Introduction

If a function block cannot start as intended, for example, because it is not properly parameterized or its execution is not permitted in the current state, the function block signals a detected error (output *Error* = TRUE). This does not typically imply that the affected axis transitions to the ErrorStop state, nor that it stops automatically.

Instead, the axis typically stays in whichever state it was before and continues to execute its current job as if the command with the detected error had not been issued. The controller application must respond to this detected error in an appropriate way.

Depending on the purpose of the function block that cannot be started (for example, starting synchronized movements with another axis, or moving to a different target position or with a different target velocity), the system behavior may entail severe consequences for the process. You must therefore implement appropriate application-specific and function block-specific error responses such

as an asynchronous stop (by calling *MC_Stop* on the affected axis), or a synchronous stop (by calling *MC_Stop* on the master axis), or any other response that is appropriate for the specific situation.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Implement appropriate error responses to all potential error conditions.
- Verify the correct operation and effectiveness of all error responses by performing comprehensive tests, including commissioning tests, for all operating states, and all potential error situations.

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

Operating State Transitions

The following cases cause a function block to signal a detected error (output *Error* = TRUE), without impacting the state or job of the affected axis:

- Invalid values for acceleration, deceleration, jerk
- Invalid axis, master, or slave
- Interruption of running function block not permitted (for example, while *MC_Stop* running)
- Execution attempt in invalid Sercos phase
- Axis, slave, or master not homed (for a function block which requires absolute positions)

If *MC_Power* is disabled while a motion function block is active, the axis goes to *Disabled* and the active motion function block signals *Aborted*.

There are a few exceptional cases in which a function block signals a detected error (output *Error* = TRUE) and the affected axis transitions to ErrorStop and performs an ErrorStop movement at the same time. This typically occurs when the situation that causes the error to occur and to be detected occurs while the job is already active. These situations are:

- Interruption of communication with master axis (*MC_CamIn*, *MC_GearIn*, *MC_CustomJob*)
- Homing not completed successfully (*MC_Home*)
- If a drive reports a detected error while a motion function block is active, *MC_Power* signals a detected error, the axis transitions to ErrorStop, and the active motion function block signals a detected error.
- Blended movement which would result in a position overshoot because blending is started while already in deceleration phase with unfavorable acceleration ramp
- Invalid LREAL value or state generated by the user code running in a custom job (*MC_CustomJob*)
- In the case of a Sercos phase-down while a motion function block is active, *MC_Power* signals a detected error, the axis transitions to ErrorStop, and the active motion function block signals a detected error.

Real-Time, Synchronous External Task

Overview

The real-time, synchronous external task is an external task that is synchronous with the Sercos RTP (Real-Time Process). The real-time, synchronous external task is triggered in the RTP and executed after the RTP finishes.

A typical usecase for the real-time task is position capture in the millisecond range without the use of touch probes.

Description

The controller-internal real-time tasks which handles Sercos communication and calculations for the axes triggers the *AFTER_RTP* event in each cycle. This event can be used to trigger a motion controller code task of the type External triggered as an external event.

With this parameterization, the real-time, synchronous external task is triggered as soon as possible after the real-time task is finished.

The priorities of the motion controller code tasks are respected. Therefore, even if this event-triggered task is already triggered by RTP, it is delayed if another task with a higher priority is triggered as well.

The task can be interrupted by the RTP.

Configuration

Procedure:

Step	Action
1	In the Device Tree , display the Task Configuration window.
2	From the listbox Type , select the item External .
3	From the listbox External Event , select the item AFTER_RTP .
4	Set a task priority and enable a Watchdog as required by your application.

NOTE: Refer to the M262 Programming Guide (see Modicon M262 Logic/Motion Controller, Programming Guide) for additional information on the task configuration screen (see Modicon M262 Logic/Motion Controller, Programming Guide) and on task priorities (see Modicon M262 Logic/Motion Controller, Programming Guide).

Cycle Time Considerations

The cycle time of the RTP only applies to the Sercos communication phase 4.

Cycle time of real-time, synchronous external task in Sercos communication phases:

Sercos communication phase	Cycle time
NRT	4 ms
0	2 ms
1	1 ms
2	1 ms
3	Sercos cycle time
4	Sercos cycle time

If the execution time of the RTP, the real-time, synchronous external task, and other tasks with higher priority exceeds the Sercos cycle time, the real-time, synchronous external task is interrupted by the RTP. In such a case, the real-time, synchronous external task is no longer synchronous with the RTP.

To keep the real-time, synchronous external task synchronous with the RTP, ensure that the execution time of the real-time, synchronous external task plus the execution time of the RTP does not exceed the configured Sercos cycle time.

Example code:

```
hTask : RTS_IEC_HANDLE;
pstTaskInfo : POINTER TO CmpIecTask.Task_Info2;
udiResult : RTS_IEC_RESULT;
dwCycleTimeInUs : DWORD;
bLossOfSynchronicity : BOOL;
```

```
lrRtpCycleTimeInUs : LREAL;
lrPercentageOfRtpCycle : LREAL;

hTask := CmpIecTask.IecTaskGetCurrent(ADR(udiResult));
IF udiResult = 0 THEN
    pstTaskInfo := CmpIecTask.IecTaskGetInfo3(hTask, ADR
(udiResult));
    IF udiResult = 0 THEN
        dwCycleTimeInUs := pstTaskInfo^.dwCycleTime;
    END_IF
END_IF

lrPercentageOfRtpCycle := S3M.FC_
GetMotionCycleTaskLoadOfLastCycle();
lrRtpCycleTimeInUs := lrPercentageOfRtpCycle * (UDINT_TO_
LREAL(SercosMaster.SercosCycletimeConfig.Cycletime) / 1E5);

IF (udiResult = 0) AND ( (lrRtpCycleTimeInUs + DWORD_TO_
LREAL(dwCycleTimeInUs)) > UDINT_TO_LREAL(SercosMaster.
SercosCycletimeConfig.Cycletime / 1000) ) THEN
    // loss of synchronicity
    bLossOfSynchronicity := TRUE;
    // TODO stop machine
END_IF
```


Library-Specific Data Types

Data Type Axis_Ref

The data type Axis_Ref is an alias of the interface *IF_Axis* of the MotionInterface library. Refer to the MotionInterface library guide for details.

Data Type MC_Buffer_Mode

The data type defines the method for the start of a new/buffered movement with regard to the ongoing movement.

Name	Value	Description
<i>Aborting</i>	0	The ongoing movement is aborted and the new movement is executed immediately in the next possible real-time cycle.
<i>Buffered</i>	1	The new/buffered movement is executed as soon as the ongoing movement has reached its steady-state, which corresponds to the function block output <i>Done</i> , <i>InVelocity</i> , <i>InSync</i> , or <i>EndOfProfile</i> , depending on the ongoing movement. The buffered job becomes active immediately in the real-time-cycle when the previous job reaches its steady-state. It does not wait for the outputs to become TRUE in the next application task cycle afterwards.
<i>BlendingLow</i> ⁽¹⁾	2	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the lower of the two velocity values of the ongoing movement and the new/buffered movement.
<i>BlendingPrevious</i> ⁽¹⁾	3	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the velocity value of the ongoing movement.
<i>BlendingNext</i> ⁽¹⁾	4	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the velocity value of the new/buffered movement.
<i>BlendingHigh</i> ⁽¹⁾	5	The new/buffered movement is executed as soon as the ongoing movement has finished, but without standstill in between. The transition is made with the higher of the two velocity values of the ongoing movement and the new/buffered movement.
<i>StartAtMasterposition</i> ⁽¹⁾	6	Provided for buffering a cam to be started at a specified master position via <i>MC_CamIn</i> , page 72.
⁽¹⁾ Only for function blocks <i>MC_MoveVelocity</i> , <i>MC_MoveAbsolute</i> , <i>MC_MoveAdditive</i> , and <i>MC_MoveRelative</i>		

The following table provides details on how buffering (*MC_BufferMode.Buffered*) works for different function blocks:

Function block	Movement of the function block can be specified as a buffered movement	Movement of the function block can be followed by a buffered movement	Condition for buffered command to become active
<i>MC_Power</i>	No	No ⁽¹⁾	-
<i>MC_MoveVelocity</i>	Yes	Yes	<i>InVelocity</i>
<i>MC_MoveAbsolute</i>	Yes	Yes	<i>Done</i>
<i>MC_MoveAdditive</i>	Yes	Yes	<i>Done</i>
<i>MC_MoveRelative</i>	Yes	Yes	<i>Done</i>
<i>MC_Home</i>	No	Yes	<i>Done</i>
<i>MC_Stop</i>	No	Yes	<i>Done</i> and input <i>Execute</i> set to FALSE
<i>MC_Halt</i>	Yes	Yes	<i>Done</i>

Function block	Movement of the function block can be specified as a buffered movement	Movement of the function block can be followed by a buffered movement	Condition for buffered command to become active
<i>MC_CamIn</i>	Yes	Yes ⁽¹⁾	<i>EndOfProfile</i> regardless of whether input <i>Periodic</i> is set to TRUE or to FALSE
<i>MC_GearIn</i>	Yes	Yes	<i>InGear</i>
<i>MC_PhasingAbsolute</i>	Yes	No	-
<i>MC_MoveSuperImposed</i>	No	No	-
<i>MC_CustomJob</i>	Yes	Yes	<i>InSteadyState</i>
⁽¹⁾ Differs from specifications as per PLCopen Motion Control Part 1, Version 2.0.			

The controller executes the user application task (in which motion control function blocks are called) in a task separate from the real-time motion process (in which cyclic reference and target values are calculated, for example). To start a function block in the same Sercos cycle in which the active function block reaches its steady state (*Done*, *inVelocity*, *EndOfProfile*, *InGear*), the function block needs to be buffered ahead of time. If you trigger the start (*Execute*) of the subsequent function block via, for example, the *EndOfProfile* signal of the preceding function block instead of buffering it, one or several cycles of delay result in which no function block is active and the axis remains at a standstill. Refer to *Task Concept*, page 14 for details.

Data Type MC_CAM_ID

The data type *MC_CAM_ID* is an alias of *ST_MultiCam* of the *CommonMotionTypes* library. Refer to the *CommonMotionTypes* library guide for details.

Data Type MC_Direction

The data type defines the direction of movement.

For *MC_MoveVelocity* and *MC_MoveRelative*, the direction inverts the direction of movement (by inverting the sign of the velocity or distance). This is independent of the axis type (modulo or linear).

For *MC_MoveAbsolute* and the ramp-in functionality of *MC_CamIn*, the direction input specifies the direction in which the absolute target position is approached. For these function blocks, the direction is only considered for modulo axes. It is ignored for linear axes (as absolute positions on linear axes are approached in the only mathematically possible direction).

Name	Value	Description
<i>PositiveDirection</i>	0	Positive direction of movement
<i>NegativeDirection</i>	1	Negative direction of movement
<i>ShortestWay</i> ⁽¹⁾	2	The direction of movement depends on whether the positive direction of movement or the negative direction of movement has the shortest distance to the target position.
⁽¹⁾ Only for function blocks <i>MC_MoveAbsolute</i> and <i>MC_CamIn</i> .		

Data Type MC_Interpolation_Mode

The data type *MC_Interpolation_Mode* is an alias of the enumeration *etInterpolationMode* of the structure *ST_Interpolation_Parameter* of the *MotionInterface* library. Refer to the *MotionInterface* library guide for details.

Data Type MC_Interpolation_Parameter

The data type `MC_Interpolation_Parameter` is an alias of the structure `ST_Interpolation_Parameter` of the MotionInterface library. Refer to the MotionInterface library guide for details.

Data Type MC_Master_Start_Mode

Name	Value	Description
<i>Absolute</i>	0	The cam starts at the X coordinate equal to the absolute master axis position at the time of start.
<i>Relative</i>	1	The cam starts at the X coordinate equal to the X of the first cam point, which is assumed to correlate with the position of the master axis at the time of start.

NOTE: In PLCopen Motion Control Part 1, Version 2.0, there is a data type `MC_Start_Mode`, as well as two Boolean flags `MasterAbsolute` and `SlaveAbsolute`. To improve clarity, this library instead implements two data types `MC_Master_Start_Mode` (a combination of `MC_Start_Mode` and `MasterAbsolute`) and `MC_Slave_Start_Mode` (a combination of `MC_Start_Mode` and `SlaveAbsolute`). The latter also contains the mode `RampIn`.

Data Type MC_OperationMode

The data type `MC_OperationMode` specifies the operating mode for the function block `MC_MoveVelocity` via the input `OperationMode`.

Name	Value	Description
<i>Position</i>	0	Velocity control with active position control loop in drive (Cyclic Synchronous Position).
<i>Velocity</i>	1	Cyclic Synchronous Velocity, pure velocity control.

The value `Position` performs a movement with the velocity set at the input `Velocity` of the function block `MC_MoveVelocity`. In this operating mode, the position control loop of the drive remains active (Cyclic Synchronous Position). This is the default operating mode for the function block `MC_MoveVelocity`.

The value `Velocity` activates the operating mode Cyclic Synchronous Velocity. In this operating mode, the position control loop of the drive is not active (pure velocity control).

The operating mode Cyclic Synchronous Velocity is started when the value at the input `OperationMode` of the function block `MC_MoveVelocity` is `Velocity` and the value at the input `Execute` changes from `FALSE` to `TRUE`.

The checkbox **VelocityOperationMode** on the tab Feature Configuration must be activated to enable the operating mode Cyclic Synchronous Velocity.

Executing `MC_MoveVelocity` with the operating mode Cyclic Synchronous Velocity for an axis whose drive does not support this operating mode or for which it has not been enabled results in a detected error by `MC_MoveVelocity` without affecting the axis behavior.

If the active operating mode is Cyclic Synchronous Velocity and the function blocks `MC_Stop` or `MC_Halt` are executed or if an error is detected with a resulting transition to the operating state `ErrorStop`, the operating mode Cyclic Synchronous Velocity remains active.

If an attempt is made to start a motion function block (for example, `MC_MoveAbsolute`) while `MC_MoveVelocity` is being executed in the operating mode Cyclic Synchronous Velocity, that motion function block is not executed and its output `Error` is set to `TRUE`. `MC_MoveVelocity` continues to run in the operating mode Cyclic Synchronous Velocity.

To change from the operating mode *Cyclic Synchronous Velocity* to a different operating mode, stop the axis with the function block *MC_Stop* or *MC_Halt*. Executing any motion function block other than *MC_MoveVelocity* then changes the operating mode from *Cyclic Synchronous Velocity* to the operating mode used by that function block. For example, if you want to change from the operating mode *Cyclic Synchronous Velocity* to the operating mode *Cyclic Synchronous Position* without starting a movement, you can execute the function block *MC_MoveRelative* with a distance of 0. If you want to change from the operating mode *Cyclic Synchronous Velocity* without executing a motion function block, disable and re-enable the power stage of the drive via the function block *MC_Power*.

If the requested operating mode is not confirmed by the drive within 30 Sercos cycles, an error is detected (output *Error* of the requesting function block is set to TRUE).

Data Type *MC_Slave_Start_Mode*

Name	Value	Description
<i>Absolute</i>	0	For starting the cam, the position of the slave axis is directly set to the first calculated Y value. The slave axis position is calculated on the basis of the cam definition and "master as seen by slave". As opposed to the slave start modes <i>Relative</i> and <i>RampIn</i> , there are no offsets and superimposed movements. The reference velocity and the acceleration are calculated on the basis of the cam definition. If there is a position difference between the position of the slave and its calculated start position (Y value) for the cam, and if this start position cannot be reached within one task scan, an error is detected. However, if this start position can be reached in spite of the position difference, this movement may be made in the form of a sudden position jump.
<i>Relative</i>	1	The cam starts at the Y coordinate defined by $f(X \text{ start})$, where $f()$ is the cam function and X start is determined by the master start mode (<i>MC_Master_Start_Mode</i>). This Y is correlated with the current slave axis position.
<i>RampIn</i>	2	It is assumed that the absolute slave axis position is supposed to be equal to the cam Y coordinate for the cam to be in sync. At the beginning, the cam starts like slave relative, meaning $f(X \text{ start})$ is correlated with the absolute slave axis position when the cam starts. Then a ramp-in movement is performed which offsets the slave axis to align the coordinate system of the axis position with the coordinate system of Y.

WARNING

UNINTENDED EQUIPMENT OPERATION

Verify the physical position of the slave axis at the start of the cam and verify that it matches the position in the cam definition.

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

NOTE: In PLCopen Motion Control Part 1, Version 2.0, there is a data type *MC_Start_Mode*, as well as two Boolean flags *MasterAbsolute* and *SlaveAbsolute*. To improve clarity, this library instead implements two data types *MC_Master_Start_Mode* (a combination of *MC_Start_Mode* and *MasterAbsolute*) and *MC_Slave_Start_Mode* (a combination of *MC_Start_Mode* and *SlaveAbsolute*). The latter also contains the mode *RampIn*.

Data Type MC_Trigger_Ref

MC_Trigger_Ref is an alias of DAL_IF_Trigger. This is an input type for *MC_TouchProbe* and *MC_AbortTrigger* in order to connect the function blocks to the corresponding *TouchProbe* inputs, similar to *Axis_Ref*.

The interface MC_Trigger_Ref/DAL_IF_Trigger provides a property *CaptureEdge* of type UINT which can have a value from 0 to 2.

Name	Value	Description
<i>FallingEdge</i>	0	Falling edge
<i>RisingEdge</i>	1	Rising edge
<i>BothEdges</i>	2	Both rising edge and falling edge

VAR

```
fb_MC_TriggerRef: PLCO.MC_Trigger_Ref;
```

```
fb_MC_Touchprobe : PLCO.MC_TouchProbe;
```

```
END_VAR
```

```
fb_MC_TriggerRef := DRV_Lexium32S.triggerCap1;
```

```
fb_MC_TriggerRef.CaptureEdge := MOIN.ET_CaptureEdge.
```

```
RisingEdge;
```

```
fb_MC_Touchprobe(Axis := DRV_Lexium32S.Axis, ifTrigger :=
```

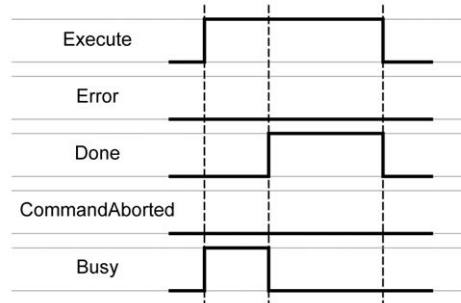
```
fb_MC_TriggerRef);
```

Common Inputs and Outputs

Behavior of Function Blocks with the Input *Execute*

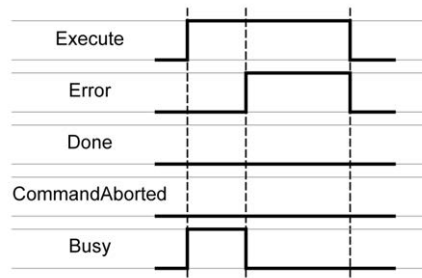
Example 1

Execution terminated without an error detected.



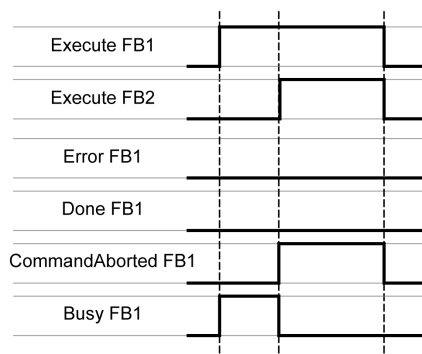
Example 2

Execution terminated with an error detected.



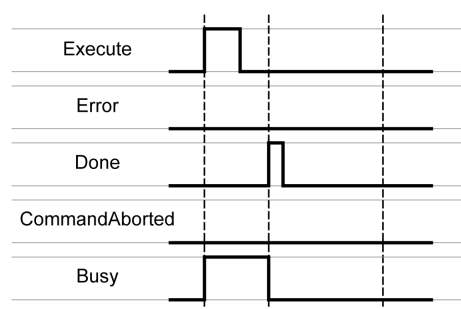
Example 3

Execution aborted because another motion function block has been started.



Example 4

If the input *Execute* is set to FALSE during a cycle, the function block execution is not terminated; the output *Done* is set to TRUE only for one cycle.



Enumerations

ET_Result - General Information

Description

This enumeration is used to return identifiers of detected errors for functions and function blocks.

Enumeration Elements

Name	Value (UDINT)	Description
<i>Ok</i>	0	No error detected
<i>UnexpectedReturnValue</i>	1	Indeterminable return value from system. Contact your Schneider Electric representative.
<i>AxisInvalid</i>	2	No axis is specified for the input <i>Axis</i> or specified axis does not support the required function. For <i>MC_Touchprobe</i> and <i>MC_AbortTrigger</i> : Specified axis does not support capture. Connect the axis for which the function block is to be executed to the input <i>Axis</i> . For <i>MC_Touchprobe</i> and <i>MC_AbortTrigger</i> : Use an axis that supports capture.
<i>IfMotionCommandNotSupported</i>	3	The connected axis does not support all of the required functions. Verify that the connected axis implements the <i>IF_MotionCommand</i> interface of the MotionInterface library.
<i>NoBusCommunication</i>	4	The fieldbus is not in the state Operational (Sercos: phase 4). Verify that the fieldbus is in the state Operational (Sercos: phase 4) at the rising edge of the input <i>Execute</i> and during the execution.
<i>PositionOutsideModulo</i>	5	The target position is outside of modulo range of the axis. Set the target position to a value within the modulo range (from 0 to modulo value of the axis).
<i>VelocityOutOfRange</i>	6	The value at the input <i>Velocity</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Velocity</i> .
<i>AccelerationOutOfRange</i>	7	The value at the input <i>Acceleration</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Acceleration</i> .
<i>DecelerationOutOfRange</i>	8	The value at the input <i>Deceleration</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>Deceleration</i> .
<i>JerkOutOfRange</i>	9	The value at the input <i>Jerk</i> is less than zero. Use a positive value or zero at the input <i>Jerk</i> .
<i>BufferModelInvalid</i>	10	A value different from <i>MC_Buffer_Mode.Aborting</i> or <i>MC_Buffer_Mode.Buffered</i> was supplied at the input <i>BufferMode</i> . Supply the value of <i>MC_Buffer_Mode.Aborting</i> or <i>MC_Buffer_Mode.Buffered</i> as <i>BufferModelInput</i> (if no value has previously been supplied, <i>MC_Buffer_Mode.Aborting</i> is used).

Name	Value (UDINT)	Description
<i>AxisIsDisabled</i>	11	Function block cannot be executed because the axis is the operating state Disabled. Verify that the axis is not in the operating state Disabled when attempting to start a new function block.
<i>AxisIsStopping</i>	12	Function block cannot be executed because an <i>MC_Stop</i> function block is active and the axis is in the operating state Stopping. Verify that the axis is not in the operating state Stopping when attempting to start a new function block.
<i>AxisNotHomed</i>	13	The axis is not homed (flag <i>xHomed</i> of axis is FALSE). Home the axis to obtain a valid zero point to start a movement relative to the zero point.
<i>AxisInErrorStop</i>	14	Function block cannot be executed because an axis error has been detected and the axis is in operating state ErrorStop. Verify that the axis is not in the operating state ErrorStop when attempting to start a new function block.
<i>BufferSaturated</i>	15	The maximum number of function blocks that can be buffered for the axis has been reached. Buffer only one function block for a given axis at any point in time.
<i>BufferNotSupported</i>	16	Buffering of this function block or combination of function blocks (for example, blending) is not permissible.
<i>FBBusyBufferModeNotPossible</i>	17	An attempt was made to trigger a new function block while a function block was waiting for a buffered function block to become active. Wait for the output <i>Active</i> of the function block to be set to TRUE before starting a new function block.
<i>MasterInvalid</i>	19	The object at the input <i>Master</i> is invalid. Provide a valid reference to the axis for which the function block is to be executed (object from Devices tree such as an axis or an encoder).
<i>DirectionInvalid</i>	20	Only use the following elements of <i>MC_Direction</i> : - For <i>MC_MoveAbsolute</i>: - PositiveDirection - NegativeDirection - ShortestWay - For <i>MC_MoveVelocity</i>: - PositiveDirection - NegativeDirection
<i>NotAbleToResetAxis</i>	21	The axis could not be reset. Read out the diagnostic code for the axis, remove the detected error and trigger the function block again.
<i>InvalidRatioNumerator</i>	22	The value at the in input <i>RatioNumerator</i> is zero. Use a value other than zero for the numerator.
<i>InvalidRatioDenominator</i>	23	The value at the in input <i>RatioDenominator</i> is zero. Use a value other than zero for the denominator.
<i>OutOfMemory</i>	24	Insufficient memory for movement command. Reduce the memory required by your application.
<i>InvalidMasterAddress</i>	25	The specified master is not valid.
<i>DriveInError</i>	35	The connected drive has detected an error. Homing cannot be started. Use the function block <i>MC_Reset</i> to reset the detected error.

Name	Value (UDINT)	Description
<i>HomingIsAlreadyActive</i>	36	The axis is being homed. Verify that the axis is in operating mode "Standstill" before executing this function block.
<i>AxisNotInStandstill</i>	37	The axis was not in operating state Standstill when homing was attempted to start. Verify that the axis is in operating mode Standstill before executing this function block.
<i>JobStartedWhileAxisIsHoming</i>	38	The command cannot be executed while the axis is in the operating state Homing.
<i>AxisResetInExecutingState</i>	39	Drive has detected an error while axis was executing.
<i>InvalidCamTableID</i>	40	The definition of the electronic cam is invalid. Verify that a correct cam table is provided for <i>MC_CamIn</i> via the input <i>CamTableID</i> .
<i>MasterIsNotModulo</i>	41	The specified master has to be defined as a modulo axis.
<i>LastMovementIsInvalid</i>	42	The ongoing job has caused an invalid movement.
<i>InvalidLambda</i>	43	One of the points of the electronic cam has an invalid Lambda value. Lambda is the value of the next cam segment preceding the inflection point. Permissible values for Lambda: $0 < \text{Lambda} < 1$.
<i>InvalidC</i>	44	One of the points of the electronic cam has an invalid C value. C is the value of the next curved segment of the electronic cam. Permissible values for C: $0 < C \leq 1$.
<i>InvalidM</i>	45	One of the points of the electronic cam has an invalid M value. M is the slope of the electronic cam at the position for which M is defined.
<i>InvalidK</i>	46	One of the points of the electronic cam has an invalid K value. K is the curvature of the electronic cam at the position for which K is defined. The value must be 0 for a simple sine (<i>ET_CamType</i> = <i>SimplSin</i>) and for a general fifth degree polynomial (<i>ET_CamType</i> = <i>Poly5Com</i>).
<i>InvalidCaptureSource</i>	47	The specified capture source does not exist. Verify that the capture source is supported by the device.
<i>DeviceAccessFailed</i>	48	Error detected writing/reading via the service channel in Sercos phase 4. Reduce the frequency of access to the service channel with <i>FB_WriteIDN</i> and/or <i>FB_ReadIDN</i> .
<i>CaptureSourceAlreadyInUse</i>	49	The same capture source is used for two function blocks <i>MC_TouchProbe</i> . Only use a single <i>MC_TouchProbe</i> with a given capture source at a time.
<i>InvalidConfiguration</i>	50	The configuration for <i>MC_TouchProbe</i> is invalid. Verify the configuration for <i>MC_Touchprobe</i> .
<i>NoCamInJobOnSlaveAxis</i>	56	<i>MC_CamIn</i> is not active for the slave axis specified. <i>MC_Phasing</i> can only be executed if <i>MC_CamIn</i> is active for the specified axis
<i>MasterAxisNotHomed</i>	57	The master axis has not been homed. Running <i>MC_CamIn</i> with <i>mcAbsolute</i> for <i>MC_Master_Start_Mode</i> requires a homed master axis.
<i>RealTimeConfigurationOfParameterFailed</i>	58	The IDNs could not be mapped in the real-time channel. Verify that cyclic data can be used and that it is possible to map the IDNs for this device.

Name	Value (UDINT)	Description
<i>DrivePowerLoss</i>	59	Power outage at connected drive.
<i>NotSupportedWithFeedbackAxis</i>	60	Command not permitted in conjunction with an axis of type feedback, such as an encoder axis. Provide a correct axis type at the input <i>Axis</i> .
<i>NoEncoderSupplyDetected</i>	61	No encoder power supply Ensure correct encoder power supply.
<i>InvalidDigitalInputConfiguration</i>	62	The configuration of the digital input of the controller for the encoder is invalid. Verify correct configuration of the digital input for the encoder.
<i>InvalidDeviceHandle</i>	63	No device with the specified handle.
<i>VelocityDifferenceOutOfRange</i>	64	The value at the input <i>VelocityDiff</i> is less than or equal to zero. Supply a positive value (greater than zero) at the input <i>VelocityDiff</i> .
<i>StartAdditiveJobDuringSuperimpose</i>	65	Additive jobs cannot start while the axis performs a superimposed movement.
<i>ActiveJobNotAllowedToBeFollowedByBlending</i>	66	The active job cannot be followed by a blended movement.
<i>InvalidCustomJobStateTransition</i>	67	Custom job has sent a job state which does not match the previously sent job state.
<i>PreactiveJobNotAllowedToBeFollowedByBlending</i>	68	The job becoming active cannot be followed by a blended movement.
<i>HomingFailed</i>	69	Error detected during homing.
<i>MasterAxisIsCurrentlyHoming</i>	70	This command cannot be executed when the master axis is being homed.
<i>AxisIsUsedAsMasterForSynchronousMotion</i>	71	This command cannot be executed when the master axis is being homed.
<i>ASlaveChannelOfMultiAxisGroupWasUnableToStart</i>	72	The axis of one of the slave channels could not start the job for the slave channel.
<i>InvalidCaptureEdge</i>	73	The specified capture edge is not valid.
<i>EmergencyStopRequiredByPlcApplication</i>	74	An emergency stop was triggered by the logic controller application.
<i>InvalidNumberOfInterpolatedCamPoints</i>	75	The value at the input <i>InterpolationParameter. udiNumCamPoints</i> is invalid The minimum number of cam points is 3, the maximum number of cam points is 10000.
<i>MaxMasterPositionIsLowerThanMinMasterPosition</i>	76	The interpolation mode <i>YArrayLinear</i> has been selected and <i>IrMinMasterPosition</i> is greater than or equal to <i>IrMaxMasterPosition</i> .
<i>InvalidInterpolationMode</i>	77	The value at the input <i>InterpolationParameter. etInterpolationMode</i> is invalid Use an element of the enumeration <i>MC_Interpolation_Mode</i> .
<i>XValuesNotStrictlyMonotonic</i>	78	X values do not increase strictly monotonically throughout the cam profile. Define a cam profile with X values that increase strictly monotonically.
<i>BlendingOvershootsFirstJob</i>	79	Blending movement would move beyond the target position of movement.
<i>MasterSlaveCascadeFormsLoop</i>	80	Master/slave function blocks create a loop (the master itself as slave, or a following slave with the initial master as slave). Resolve the loop in the chain.
<i>OperationModeInvalid</i>	81	Invalid operating mode selected for <i>MC_MoveVelocity</i> . Select operating mode "Position" or operating mode "Velocity".

Name	Value (UDINT)	Description
<i>OperationModeChangeNotAllowedForAxisNotInStandstill</i>	82	An attempt was made to change the operating mode for an axis that is not in the state Standstill. Ensure that the axis is in the state Standstill before changing the operating mode.
<i>OperationModeIDNsNotMapped</i>	83	A job with operating mode "Velocity" was started for a drive for which no velocity IDNs are mapped. Verify that the operating mode "Velocity" is activated in the drive feature configuration (checkbox VelocityOperationMode).
<i>OperationModeChangeNotAllowedForMasterAxisNotInStandstill</i>	84	An attempt was made to change the operating mode to "Velocity" for an axis that is used as master for a synchronous movement and that is not in the state Standstill. Ensure that the master axis is in the state Standstill before changing the operating mode.
<i>ModuloAxisNotSupportedWithOperationModeVelocity</i>	85	Reserved.
<i>NotPossibleToStartMoveSuperImposedOnCsvOperationModeAxis</i>	86	A function block <i>MC_MoveSuperimposed</i> is running or was started for an axis which is in the operating mode Cyclic Synchronous Velocity or will transition to this operating mode. Verify that function blocks <i>MC_MoveSuperimposed</i> are not used while the axis is in the operating mode Cyclic Synchronous Velocity.
<i>NotSupportedWithLimitedAxis</i>	87	An attempt was made to use a function block <i>MC_TorqueControl</i> for an axis of the type linear axis with limited movement range. Verify that no function block <i>MC_TorqueControl</i> is used for an axis of the type linear axis with limited movement range.
<i>AbortingTorqueControlNotPossibleWithThisJob</i>	88	An attempt was made to abort a running function block <i>MC_TorqueControl</i> with a further motion function block. Only <i>MC_TorqueControl</i> , <i>MC_Stop</i> , and <i>MC_Power</i> can be used to abort a running function block <i>MC_TorqueControl</i> .
<i>NotPossibleToStartMoveSuperImposedOnCstOperationModeAxis</i>	89	A function block <i>MC_MoveSuperimposed</i> is running or was started for an axis which is in the operating mode Cyclic Synchronous Torque or will transition to this operating mode. Verify that function blocks <i>MC_MoveSuperimposed</i> are not used while the axis is in the operating mode Cyclic Synchronous Torque.
<i>TorqueInValuesOutOfRange</i>	90	The value at the input <i>Torque</i> of a function block <i>MC_TorqueControl</i> is not within the permissible range. The permissible value range is between $-30 \cdot \text{continuous stall torque (M\_M\_0\_)}$ to $+30 \cdot \text{continuous stall torque (M\_M\_0\_)}$ of the connected motor.
<i>StartAtMasterPositionDoesNotInterruptACam</i>	91	An attempt was made to start a function block <i>MC_CamIn</i> with the buffer mode <i>StartAtMasterPosition</i> , but no other cam is active for the axis. The buffer mode <i>StartAtMasterPosition</i> requires another cam to be active for the axis.
<i>MasterStartPositionIsNotInsidePreviousCamRange</i>	92	An attempt was made to start a function block <i>MC_CamIn</i> with the buffer mode <i>StartAtMasterPosition</i> , but the master start position is outside of the master position range as seen by the slave range of the running cam. The master start position has to be greater than or equal to the X value of the leftmost cam point, and less than or equal to the X value of the rightmost cam point of the cam currently running for the axis.
<i>MasterChangeNotAllowedWithStartAtMasterPosition</i>	93	An attempt was made to start a function block <i>MC_CamIn</i> with the buffer mode <i>StartAtMasterPosition</i> , but the master of the cam currently running cam for the axis is different from the master of the new cam. The buffer mode <i>StartAtMasterPosition</i> is only possible if both cams have the same master.

Name	Value (UDINT)	Description
<i>NegativeTorqueRampValueNotAllowed</i>	94	The value at the input <i>TorqueRamp</i> of a function block <i>MC_TorqueControl</i> is less than zero. Provide a positive value if you want to use a torque ramp. If the input is set to 0, the target torque specified via the input <i>Torque</i> is generated immediately without a torque ramp.
<i>TimeoutWhileEnablingAxis</i>	95	After triggering of a function block <i>MC_Power</i>, the axis does not transition to the operating state <i>Standstill</i> in time. Verify correct mains supply of the drive and the signal state of the safety-related function Safe Torque Off (STO()).
<i>TorqueJobNotAllowedWithSimulatedDrive</i>	96	An attempt was made to start a function block <i>MC_TorqueControl</i> for an axis whose working mode is set to simulated. <i>MC_TorqueControl</i> requires the working mode real.
<i>ConflictingIdnMapping</i>	97	At least one IDN to write a parameter in the slave was mapped manually, which needs to be mapped by the system as well. Write parameters can be mapped only once. Either remove the manual mapping, or deactivate the corresponding system feature.
<i>WrongOperationModeOnDrive</i>	98	The drive has not confirmed the requested transition to operating mode Cyclic Synchronous Position (CSP), Cyclic Synchronous Velocity (CSV), or Cyclic Synchronous Torque (CST), Contact your Schneider Electric representative.

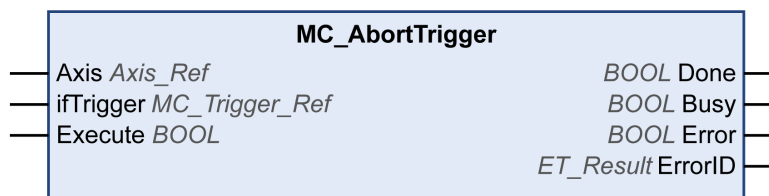
Function Blocks - Single Axis

MC_AbortTrigger

Functional Description

This function block terminates position capture.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>ifTrigger</i>	MC_Trigger_Ref	Edge to trigger position capture. Refer to <i>MC_Trigger_Ref</i> for a description.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. A rising edge at the input <i>Execute</i> is ignored while the function block is being executed.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.

Output	Data type	Description
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

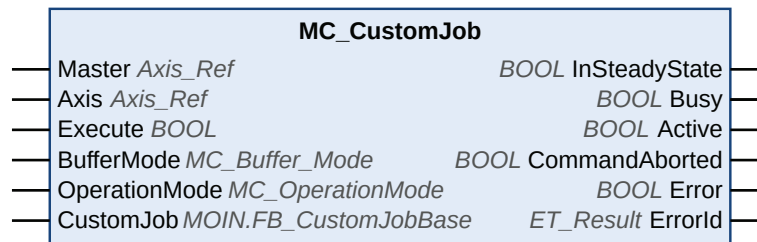
MC_CustomJob

Functional Description

This function block allows you to control an axis by a custom algorithm that calculates cyclic set position, velocity, and acceleration of the axis.

The function block created by you in order to program a motion profile has to extend *FB_CustomJobBase* of the MotionInterface library. Then the function block is provided at the input *CustomJob*.

Graphical Representation



Inputs

Input	Data type	Description
<i>Master</i>	Axis_Ref	Reference to the axis for which the function block is to be executed. Can be left unassigned if the custom job (provided at input <i>CustomJob</i>) does not use a master axis. If an axis is assigned, the callback to the user function block to define the motion profile gets the movement values of the master axis. Otherwise, the movement values of the master axis are given as zero.
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>BufferMode</i>	MC_Buffer_Mode, page 21	The target values (position, velocity, acceleration) of the axis are overwritten by the new values in the motion task cycle when the function block gets active on the axis. Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> See <i>MC_Buffer_Mode</i> for a description of the values.

Input	Data type	Description
<i>OperationMode</i>	MC_OperationMode, page 23	Operating mode for function block Default value: Position
<i>CustomJob</i>	MOIN.FB_ CustomJobBase	An instance of a user-created function block type which must be derived from <i>FB_CustomJobBase</i> . The function block instance can be parameterized with additional parameters (for example,. target position, velocity, acceleration, jerk, etc.) according to the requirements of the algorithm used by the custom job. Override the following methods: • <i>CalculateMovement</i> • <i>Prepare</i> • <i>ResetJob</i> Do not override the other methods of this function block.

Outputs

Output	Data type	Description
<i>InSteadyState</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE, as reported by the custom job • FALSE: Steady state has not yet been reached or an error has been detected. • TRUE: Steady-state reached. This way, the custom job signals that a buffered job can become active.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE, as reported by the custom job • FALSE: Function block is not being executed. • TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. • FALSE: The function block does not control the movement of the axis. • TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. • FALSE: Execution has not been aborted. • TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. • FALSE: Function block is being executed, no error has been detected during execution. • TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

If you use the function block *MC_SetPosition* with a function block *MC_CustomJob*, this can result in position jumps if you do not consider the offset position in your position calculation.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use the function block *MC_SetPosition* with a function block *MC_CustomJob* without adjusting the offset position

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

To avoid any potential position jumps, base the calculation of the axis position for the next cycle on the last physical position (as per *Axis.IrPosition*) of the axis, or otherwise verify that the offset position is correctly considered in your position calculation.

If you use the function block *MC_CustomJob* with a modulo axis, the position generated via the method *CalculateMovement* is modulo-corrected if a modulo overflow occurs. This correction is based on saving the modulo offset in *MC_CustomJob*. This implies that if the calculation is based on the last reference position (as per *Axis.IrPosition*), the position for the next cycle drifts by the magnitude of the modulo jump.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Verify that all effects of modulo jumps are correctly considered in your position calculation if you use the function block *MC_CustomJob* with a modulo axis.

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

Notes

If an axis is provided for the input *Master*, the new target values or reference values for the master axis for the running real-time cycle are calculated before *MC_CustomJob* is triggered. Therefore, the custom job implementation gets up-to-date (newly calculated from the real-time cycle) values from the master axis when it is called to calculate its values for the slave axis.

If the operating mode is set to Velocity via the input *OperationMode* and if the drive is not able to operate in the operating mode Cyclic Synchronous Velocity, the function block *MC_CustomJob* detects an error. The axis is not affected.

MC_Halt

Functional Description

This function block stops the ongoing movement. The function block can be aborted by other function blocks. See *MC_Stop* for a stop that cannot be aborted.

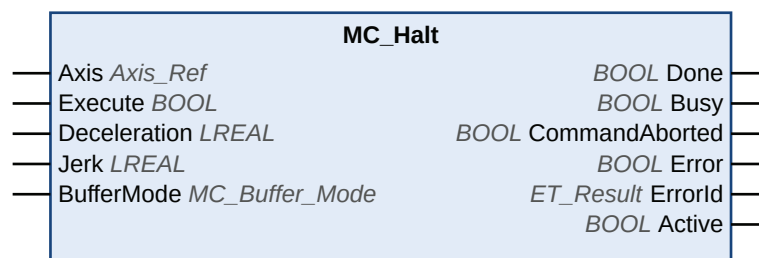
When this function block is started, the execution of any other function block is aborted.

The execution of the function block *MC_Halt* can be aborted by another function block in non-buffered mode.

If the function block *MC_Halt* is triggered, the axis transitions to the PLCopen operating state Discrete Motion and remains in this operating state until the motor has reached a standstill or another function block is started. Once the motor has reached standstill, the output *Done* is set and the axis transitions to the operating state StandStill.

As opposed to the function block *MC_Stop* which is primarily intended for Emergency Stop functions, the function block *MC_Halt* is intended for regular operation of the axis.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.

Input	Data type	Description
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> See <i>MC_Buffer_Mode</i> for a description of the values.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.

Additional Information

PLCopen State Diagram, page 17

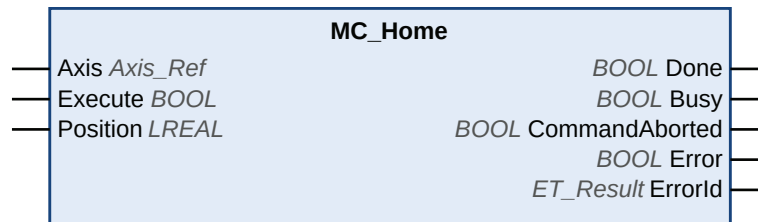
MC_Home

Functional Description

This function block homes the drive with the homing-specific settings of the drive.

Refer to the user guide of the drive, page 8 for the Homing-specific parameter settings.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. A rising edge at the input <i>Execute</i> is ignored while the function block is being executed.
<i>Position</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Position at reference point in user-defined units. After a successful reference movement, this position is automatically set at the reference point.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.

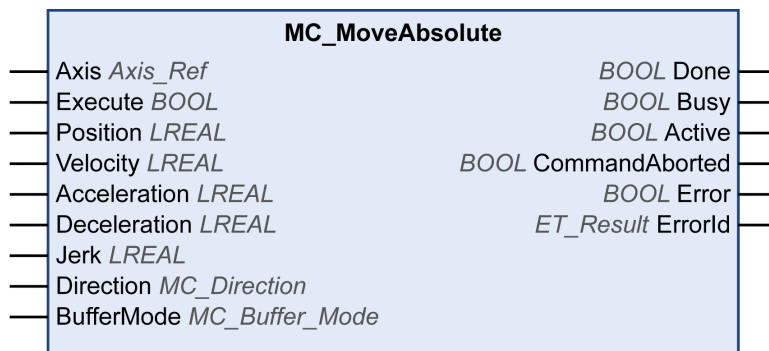
Output	Data type	Description
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

MC_MoveAbsolute

Functional Description

This function block performs a movement to a specified absolute target position.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Position</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0. Target position absolute in user-defined units.
<i>Velocity</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Target velocity in user-defined units.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units.
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0

Input	Data type	Description
<i>Direction</i>	MC_Direction, page 22	Default value: <i>PositiveDirection</i> Direction of movement. Possible values: - Value <i>PositiveDirection</i> - Value <i>NegativeDirection</i> - Value <i>ShortestWay</i> (only for modulo axes, ignored for linear axes) See <i>MC_Direction</i> for a description of the values.
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> - Value <i>BlendingLow</i> - Value <i>BlendingPrevious</i> - Value <i>BlendingNext</i> - Value <i>BlendingHigh</i> See <i>MC_Buffer_Mode</i> for a description of the values.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Notes

Absolute positioning requires a valid zero point. This means the axis has to be homed (flag *xHomed* must be TRUE).

Additional Information

PLCopen State Diagram, page 17

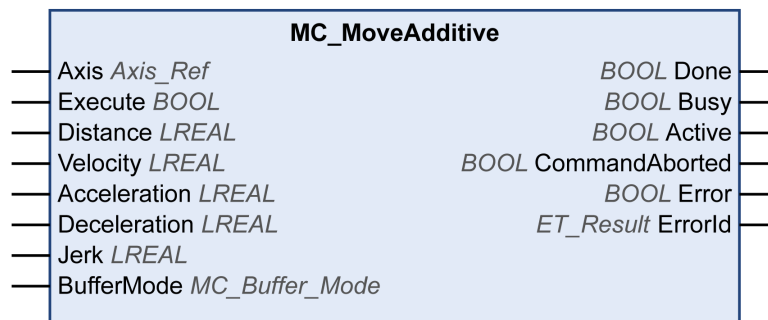
MC_MoveAdditive

Functional Description

This function block performs a movement with a specified distance with reference to the previous target position.

The function block replaces another positioning function block currently active on the axis. It performs a movement to a new target position which is calculated based on the target position of the previously active positioning function block plus the value of the *Distance* input of *MC_MoveAdditive*. If no function block is active, a new movement is started equal to the functionality of *MC_MoveRelative*.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Distance</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0. Target position relative to the previous target position in user-defined units.
<i>Velocity</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Target velocity in user-defined units.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units.
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.

Input	Data type	Description
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> - Value <i>BlendingLow</i> - Value <i>BlendingPrevious</i> - Value <i>BlendingNext</i> - Value <i>BlendingHigh</i> See <i>MC_Buffer_Mode</i> for a description of the values.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Additional Information

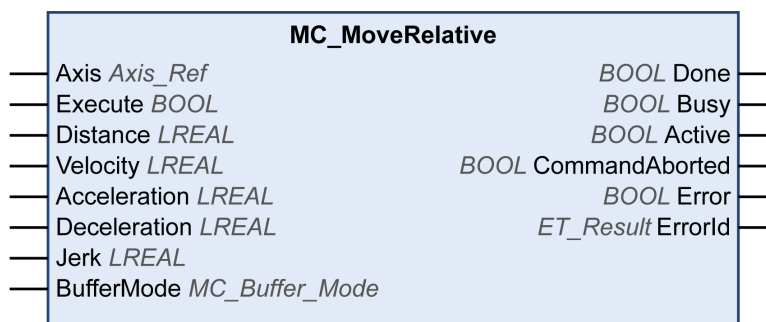
PLCopen State Diagram, page 17

MC_MoveRelative

Functional Description

This function block performs a movement with a specified distance with reference to the actual position.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Distance</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0. Target position relative with reference to actual position.
<i>Velocity</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Target velocity in user-defined units. Negative values for the target velocity invert the direction of the movement.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units.
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.

Input	Data type	Description
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> - Value <i>BlendingLow</i> - Value <i>BlendingPrevious</i> - Value <i>BlendingNext</i> - Value <i>BlendingHigh</i> See <i>MC_Buffer_Mode</i> for a description of the values.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Additional Information

PLCopen State Diagram, page 17

MC_MoveSuperImposed

Functional Description

This function block performs a superimposed movement with a specified position offset with reference to the position of an ongoing movement.

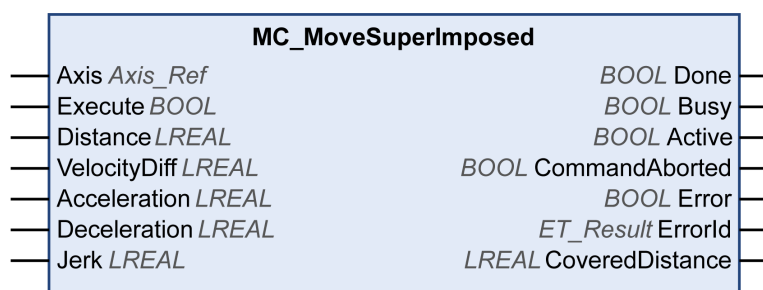
The function block can be used to add an offset movement based on the measurements of an encoder or other sensor, for example, to compensate for size differences of irregularly shaped objects on a belt.

If a new function block *MC_MoveSuperImposed* is started while another function block *MC_MoveSuperImposed* is still running, the running function block is aborted and the new one started. The underlying motion command is not aborted.

If the underlying function block is aborted by another function block, for example, *MC_Stop*, the superimposed movement is aborted as well.

The output *CoveredDistance* indicates the distance moved.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Distance</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Additional distance to be superimposed in user-defined units.
<i>VelocityDiff</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Value of the velocity difference of the additional movement in user-defined units.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units.

Input	Data type	Description
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed. NOTE: The output <i>Busy</i> remains TRUE even when the target velocity has been reached or <i>Execute</i> becomes FALSE. The output <i>Busy</i> is set to FALSE as soon as another function block such as <i>MC_Stop</i> is executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.
<i>CoveredDistance</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Indicates the distance moved in user-defined units.

Notes

Setting the input *Distance* to 0 halts the superimposed movements without halting the underlying movement (acts like the function block *MC_HaltSuperimposed* which is not separately implemented in the library).

Starting a function block *MC_MoveAdditive* while a function block *MC_MoveSuperImposed* is running results in a detected error.

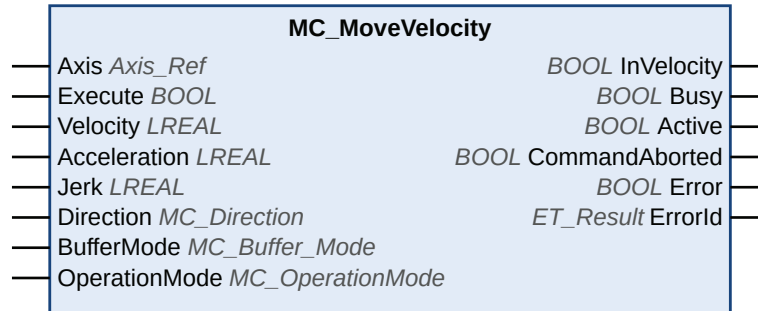
The implementation of the function block *MC_MoveSuperimposed* complies with the specifications of PLCopen Motion Control Part 1, Version 2.0. It differs from the SoftMotion SM3_Basic library (refer to *Specific Information on Individual Function Blocks*, page 83).

MC_MoveVelocity

Functional Description

This function block performs a movement with a specified target velocity.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Velocity</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Target velocity in user-defined units. Negative values for the target velocity invert the direction of the movement.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units. The value at this input is used to reach the specified target velocity (acceleration or deceleration).
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0
<i>Direction</i>	MC_Direction, page 22	Default value: <i>PositiveDirection</i> Direction of movement. Possible values: - Value <i>PositiveDirection</i> - Value <i>NegativeDirection</i> See <i>MC_Direction</i>, page 22 for a description of the values.

Input	Data type	Description
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> - Value <i>BlendingLow</i> - Value <i>BlendingPrevious</i> - Value <i>BlendingNext</i> - Value <i>BlendingHigh</i> See <i>MC_Buffer_Mode</i>, page 21 for a description of the values.
<i>Operation-Mode</i>	MC_OperationMode, page 23	Default value: <i>Position</i> Type of operation for this function block. Possible values: - Value <i>Position</i> - Value <i>Velocity</i> See <i>MC_OperationMode</i>, page 23 for a description of the values.

Outputs

Output	Data type	Description
<i>InVelocity</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Target value not reached. - TRUE: Target value reached.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed. NOTE: The output <i>Busy</i> remains TRUE even when the target velocity has been reached or <i>Execute</i> becomes FALSE. The output <i>Busy</i> is set to FALSE as soon as another function block such as <i>MC_Stop</i> is executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.

Output	Data type	Description
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Notes

The output *Busy* remains TRUE even if the target velocity has been reached or the input *Execute* is set to FALSE. The output *Busy* is set to FALSE as soon as another function block such as *MC_Stop* is executed.

If you use *MC_MoveVelocity* to move an axis continuously in the same direction and if the input *OperationMode* is set to Position, define this axis as modulo axis. Refer to Movement Range and Position Calculation With Floating-Point Numbers, page 16 for additional information.

The function block can be used with two different operating modes. See data type *MC_OperationMode*, page 23 for details.

Additional Information

PLCopen State Diagram, page 17

MC_Power

Functional Description

This function block enables or disables the power stage of the drive.

TRUE at the input *Enable* enables the power stage. Once the power stage is enabled, the output *Status* is set.

FALSE at the input *Enable* disables the power stage. Once the power stage is disabled, the output *Status* is reset.

If errors are detected during execution, the output *Error* is set.

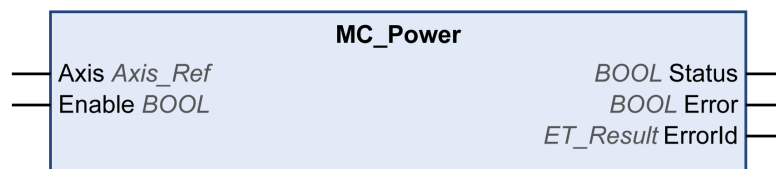
Whenever the function block is called, the input *Enable* is compared to the output *Status*. If these values are different, a new command is executed, either to enable the power stage (*Enable* = TRUE and *Status* = FALSE) or to disable the power stage (*Enable* = FALSE and *Status* = TRUE). The function must be called as long as the requested state of the power stage is reached or until an error is detected. If a function block error (for example, time-out) is detected, the *Error* output is set. The output is reset with the next call of the function block if the cause of the detected error has been removed and acknowledged with *MC_Reset*.

If the power stage is not enabled within a timeout of 3000 ms, an error is detected. In such a case, remove the cause of the error and trigger *MC_Power*. You can use the functions *FC_SetPowerEnableTimeout* and *FC_GetPowerEnableTimeout* of the SercosMaster library to modify the default timeout value of 3000 ms and to read the timeout value.

Call this function block cyclically, for example, in order to detect errors of the axis.

Use only a single instance of this function block per axis.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Enable</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. The input <i>Enable</i> starts or terminates execution of a function block. - FALSE: Execution of the function block is terminated. The outputs <i>Valid</i>, <i>Busy</i>, and <i>Error</i> are set to FALSE. - TRUE: The function block is being executed. The function block continues executing as long as the input <i>Enable</i> is set to TRUE.

Outputs

Output	Data type	Description
<i>Status</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Power stage is disabled. - TRUE: Power stage is enabled.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Additional Information

PLCopen State Diagram, page 17

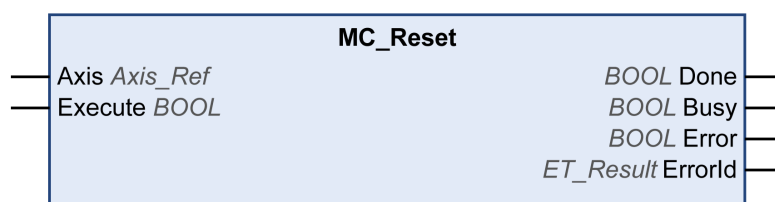
MC_Reset

Functional Description

This function block acknowledges detected axis-related errors and drive-related errors.

The error memory is cleared so that it is available for new error messages. If the power stage is disabled by the error response of the drive, the power stage can be enabled again if the cause of the detected error has been rectified when the error message is acknowledged.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. A rising edge at the input <i>Execute</i> is ignored while the function block is being executed.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.

Output	Data type	Description
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

MC_SetPosition

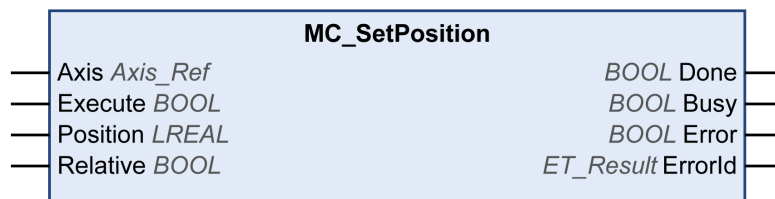
Functional Description

This function block sets a position value at the actual position of the motor to define the zero point.

The position value set with this function block determines the zero point.

The function block can be called at any time.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. A rising edge at the input <i>Execute</i> is ignored while the function block is being executed.
<i>Position</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Position in user-defined units. Value for position setting.
<i>Relative</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The actual position is set to the value of the input <i>Position</i>. - TRUE: The value of the <i>Position</i> is added to the actual position. If the absolute position is set, the flag <i>xHomed</i> of the axis is set to TRUE as well.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

MC_Stop

Functional Description

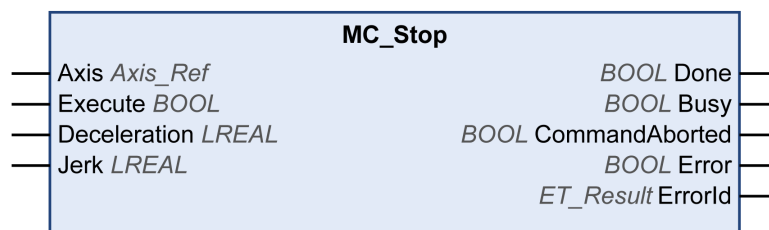
This function block stops the ongoing movement. No other movements can be started as long as this function block is active. See *MC_Halt* for a stop that can be aborted.

The function block *MC_Stop* triggers a stop of the drive. With the exception of the operating mode Cyclic Synchronous Torque (*MC_TorqueControl*), the stop is performed with the values of the inputs *Deceleration* and *Jerk*. No parameters of the drive are used. If the function block is used to abort a function block *MC_TorqueControl*, page 67, the values of the inputs *Deceleration* and *Jerk* are ignored and the stop is performed with the maximum current specified via the corresponding drive parameter.

When this function block is executed, the axis transitions to the PLCopen operating state Stopping and remains in this operating state as long as the input *Execute* is *TRUE*. As long as the axis is in this operating state, no other function block can be executed.

After successful completion of the function block, the axis transitions to the operating state StandStill. After a stop in the operating mode Cyclic Synchronous Torque, the operating mode is set to Position (refer to data type *MC_OperationMode*, page 23 for details).

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration in user-defined units.
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Notes

As long as the input *Execute* is TRUE, no other function block except for *MC_Power* can be started.

If an attempt is made to start a second function block *MC_Stop* while another function block *MC_Stop* is running, the output *Error* of the second *MC_Stop* is set to *TRUE* and the axis continues to decelerate with the settings of the first *MC_Stop*.

Additional Information

PLCopen State Diagram, page 17

MC_TorqueControl

Functional Description

This function block allows you to operate a drive in the operating mode Cyclic Synchronous Torque (CST).

In the operating mode Cyclic Synchronous Torque, movements are made with a specified target torque. The target torque in Nm is provided via the input *Torque*. The permissible torque range at this input is -30 times the continuous stall torque (*M_M_0_*) to +30 times the continuous stall torque of the motor connected to the drive. Negative values start a movement in negative direction of movement.

The continuous stall torque is a motor-specific value. During phasing-up (transition to Communication Phase 2), the system determines the continuous stall torque value via the parameter *P-3013-0-22*. When the function block is started (value at the input *Execute* is set to TRUE), the system verifies that the torque value at the input *Torque* is valid.

The input *TorqueRamp* allows you to specify a torque ramp in Nm/s. If the value at the input *TorqueRamp* is 0, the torque specified via the input *Torque* is generated immediately without a torque ramp.

The output *InTorque* is set to TRUE once the specified target torque is reached.

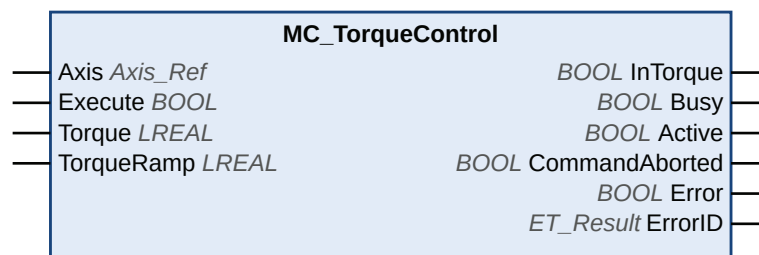
The function block can be started when the axis is in the operating state StandStill.

The function block can be aborted in three ways:

- By another function block *MC_TorqueControl*
- By disabling the power stage of the drive via the function block *MC_Power*, page 59
- Via the function block *MC_Stop*, page 65

If the requested operating mode is not confirmed by the drive within 30 Sercos cycles, an error is detected (output *Error* of the requesting function block is set to TRUE).

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.

Input	Data type	Description
<i>Torque</i>	LREAL	Target torque for the operating mode Cyclic Synchronous Torque in Nm Value range: Any positive LREAL value Value range: -30 * continuous stall torque (M_M_0_) to +30 * continuous stall torque (M_M_0_) of the connected motor Negative values trigger a movement in negative direction, positive values trigger a movement in positive direction of movement. Default value: 0
<i>TorqueRamp</i>	LREAL	Torque ramp for the operating mode Cyclic Synchronous Torque in Nm/s. If the input is set to 0, the target torque specified via the input <i>Torque</i> is generated immediately without a torque ramp. Value range: Any positive LREAL value Default value: 0

Outputs

Output	Data type	Description
<i>InTorque</i>	BOOL	This output indicates whether the specified target torque has been reached. Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

In the operating mode Cyclic Synchronous Torque, the drive can be in the PLCopen operating state Standstill. In this operating state, the target torque is 0 Nm. When the torque is 0 Nm, movements are possible, for example due to external forces. There is no monitoring for physical standstill of the motor.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- In your risk assessment, take into account all consequences that can arise when the motor torque is 0 Nm.
- Implement all measures required to ensure that a motor torque of 0 Nm at standstill does not result in hazardous movements as identified in your risk assessment (for example, install mechanical brakes).

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

Notes

The checkbox **TorqueOperationMode** on the tab Feature Configuration needs to be checked to enable the operating mode Cyclic Synchronous Torque.

In the case of LMX28S drives, you can use either the operating mode Cyclic Synchronous Torque or the operating mode Cyclic Synchronous Velocity (the operating modes are not available at the same time). Check only one of the two checkboxes.

You can use the function *FC_GetTorqueInNm* of the SercosMaster library to read the torque value.

MC_TouchProbe

Functional Description

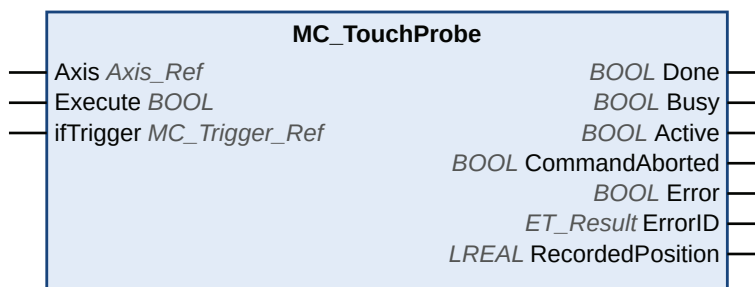
This function block configures and starts position capture.

The function block returns the axis position when a trigger event occurs. The trigger parameters of the drive are provided by the device implementation.

Executing the function block *MC_AbortTrigger* while *MC_TouchProbe* is busy aborts the function for the referenced trigger input.

A new rising edge at the input *Execute* overwrites and restarts the active trigger function.

Graphical Representation



Inputs

Input	Data type	Description
<i>Axis</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>ifTrigger</i>	MC_Trigger_Ref	Edge to trigger position capture. Refer to <i>MC_Trigger_Ref</i> for a description.

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been finished, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.
<i>RecordedPosition</i>	DINT	Returns the captured position value in user-defined units when a trigger event occurs. Value range: -2147483648...2147483647 Default value: 0

Notes

Use the function block *MC_AbortTrigger* to abort the execution of the function block *MC_TouchProbe*.

Function Blocks - Multi Axis

MC_CamIn

Functional Description

This function block activates master-slave coupling with the profile for an electronic cam specified in a cam table.

The library supports the following cam types (motion laws) via the CommonMotionTypes library (refer to *ST_MultiCam* and *ET_CamType* in the CommonMotionTypes library guide for details):

- Straight line
- Simple sine
- General fifth degree polynomial
- Standard fifth degree polynomial

ST_MultiCam is the same data structure used by PacDrive3 and therefore can be created with the same cam editor.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Verify the physical position of the slave axis at the start of the cam and verify that it matches the position in the cam definition.

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

Ramp-In Mechanism

The function block provides a ramp-in mechanism. The ramp-in mechanism is activated by setting the input *SlaveStartMode* to *RampIn* and configured via the inputs *VelocityOffsetRampIn*, *AccelerationOffsetRampIn*, *DecelerationOffsetRampIn*, and *JerkOffsetRampIn*. The ramp-in direction of a modulo axis can be set via the input *RampInDirection*.

Interpolated Cam

The function block lets you implement interpolated cams. Two types of interpolated cams are available:

- Linear interpolation
- Interpolation with Poly5 cam law

The cam is interpolated from an array of cam points. To use an interpolated cam, create an array in your application with a minimum of 3 points and a maximum of 10000 points.

Linear interpolation:

The array describes the function of the cam ($Y = f(X)$). The values you specify for the array are the Y coordinates of the cam points. These Y values are equidistantly distributed along the X axis (which means that the X coordinates are determined by the function block). The array values are assigned in ascending order to the individual points from left to right, starting with the lowest array index as the lowest X value.

Interpolation with Poly5 cam law:

The array describes the function of the cam in terms of the master position (X), the slave position (Y), the velocity at the cam point (V, corresponds to the slope), and

the acceleration at the cam point (A, corresponds to the curvature). Use strictly monotonically increasing values for X.

To start an interpolated cam, set the input *InterpolationPoints* to the address of the array where the cam points are stored. If the input *InterpolationPoints* is not equal to zero on a positive edge of the input *Execute*, the function block *MC_CamIn* starts an interpolated cam as parameterized via the input *InterpolationParameter*. Data provided via the input *CamTableID* is ignored. If the input *InterpolationPoints* is equal to zero on a positive edge of the input *Execute*, the function block starts the cam and ignores the data provided via the input *InterpolationParameter*.

The data type *MC_Interpolation_Parameter* is used to parameterize the interpolated cam. It is an alias of the structure *ST_Interpolation_Parameter* of the MotionInterface library. Parameterization:

- *udiNumCamPoints*
Number of array entries filled with cam points. If the array is larger than the amount of filled cam points, additional array elements are ignored.
- *IrMinMasterPosition* and *IrMaxMasterPosition*
For an array for linear interpolation, the position range of the master is set via *IrMinMasterPosition* and *IrMaxMasterPosition*. The cam point in the lowest array index corresponds to *IrMinMasterPosition*. The cam point in the array index set via *udiNumCamPoints* corresponds to *IrMaxMasterPosition*. The other cam points are evenly distributed between these master positions. *IrMinMasterPosition* and *IrMaxMasterPosition* are ignored for Poly5 interpolation.
- *etlInterpolationMode*
This enumeration specifies the interpolation type.
 - *YArrayLinear* (cam is a straight line between each of the cam points)
 - *XYVAArrayPoly5* (polynomial of fifth degree)

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the number of interpolation points you specify for the input *InterpolationPoints* is the same value you specify for *udiNumCamPoints* of the structure *ST_InterpolationParameter* used for the input *InterpolationParameter* if you use an interpolated cam.
- Verify that the values for X of the structure *ST_InterpolationPointXYVA* increase strictly monotonically if you use an interpolated cam with cam law Poly5.
- Verify that the data in the array of cam points is not modified while the cam is buffered or being executed.
- Verify that no online modifications are triggered while the cam is being executed.

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

Refer to the MotionInterface library guide for details on parameterizing an interpolated cam via *ST_InterpolationParameter*.

Starting a Cam at a Specific Master Position

The input *MasterStartPosition* is provided to start a cam at the specified master position. This input is ignored unless the buffer mode is set to *StartAtMasterPosition* via *MC_BufferMode*, page 21.

If *MC_CamIn* with the buffer mode *StartAtMasterPosition* is started, the slave axis need to be performing an other cam. If this is not the case, *MC_CamIn* detects an error without interfering with the movement of the slave axis. The value provided via the input *MasterStartPosition* has to be inside the range of *MasterAsSeenBySlave* defined by the currently running cam. If this is not the

case, *MC_CamIn* detects an error without interfering with the movement of the slave axis.

If there is another job already buffered behind the currently running cam when *MC_CamIn* with the buffer mode *StartAtMasterPosition* is started (input *Execute* set to TRUE), the buffered job is set to *CommandAborted* as if the function block with the buffer mode *StartAtMasterPosition* had interrupted the running cam with the buffer mode *Aborting*.

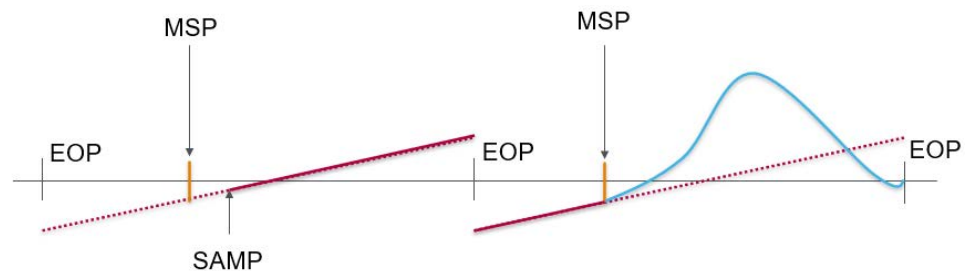
Behavior in conjunction with output *EndOfProfile*:

- If the running cam finishes its last segment (output *EndOfProfile* is set to TRUE), before the position *MasterStartPosition* of a function block with the buffer mode *StartAtMasterPosition* is reached, the running cam behaves as if no other command is started.
- If the position *MasterStartPosition* of a function block with the buffer mode *StartAtMasterPosition* is reached before the running cam finishes its last segment, the running cam behaves as if it was aborted by the buffered function block (*CommandAborted* is set to TRUE, *EndOfProfile* remains FALSE).

Behavior of *StartAtMasterPosition* with running periodic cam:

- If a function block *MC_CamIn* with the buffer mode *StartAtMasterPosition* is triggered during the execution of a running periodic cam, and if the running cam reaches its *EndOfProfile* before the position *MasterStartPosition* is reached, the running cam “turns around” and sets its output *EndOfProfile* to TRUE for one cycle.
- In the next cycle of the running periodic cam, the position *MasterStartPosition* is reached before the running periodic cam reaches its *EndOfProfile*. This is when the new cam with the buffer mode *StartAtMasterPosition* is started.

The following figure illustrates this behavior:



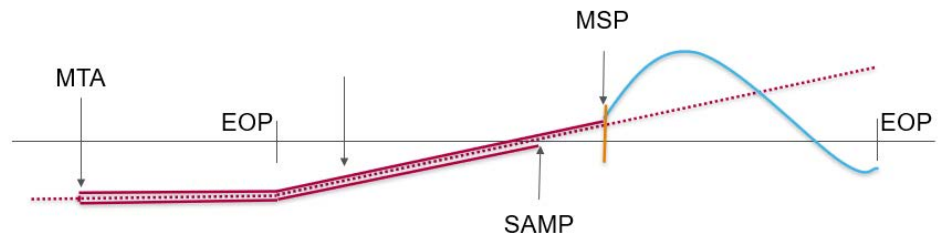
Legend:

- MSP = Position *MasterStartPosition*
- SAMP = *MC_CamIn* with buffer mode *StartAtMasterPosition* is triggered
- EOP = Position at *EndOfProfile*

Behavior of *StartAtMasterPosition* with running onetime cam:

- If a function block *MC_CamIn* with the buffer mode *StartAtMasterPosition* is triggered during the execution of a running onetime cam, and if the running cam reaches its *EndOfProfile* before the position *MasterStartPosition* is reached, the running cam sets its output *EndOfProfile* to TRUE and remains at the position as if no other cam is triggered.
- When the master “turns around” and the position *MasterStartPosition* is reached, the function block *MC_CamIn* with the buffer mode *StartAtMasterPosition* is started. *CommandAborted* is set to TRUE, *EndOfProfile* remains FALSE.

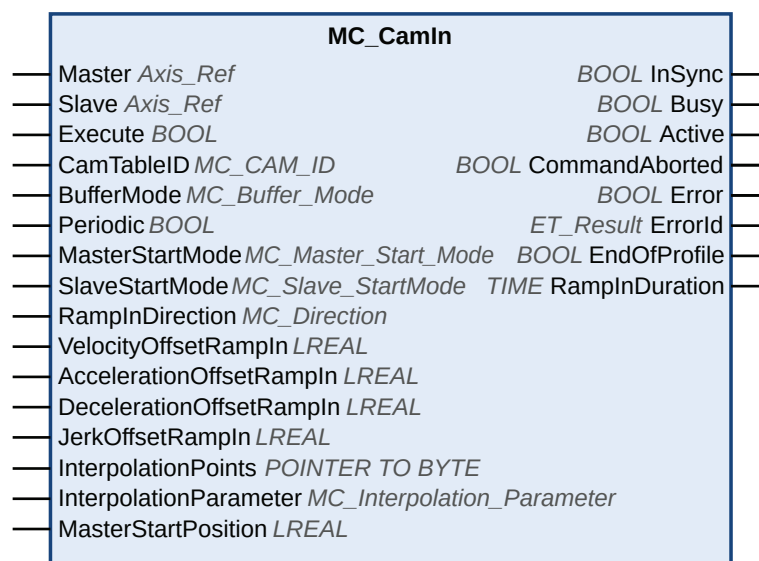
The following figure illustrates this behavior:



Legend:

- MSP = Position *MasterStartPosition*
- SAMP = *MC_CamIn* with buffer mode *StartAtMasterPosition* is triggered
- EOP = Position at *EndOfProfile*
- MTA = Master “turns around”

Graphical Representation



Inputs

Input	Data type	Description
<i>Master</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Slave</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>CamTableID</i>	MC_CAM_ID	Identifier of the cam table to be used. The data type MC_CAM_ID is an alias of <i>ST_MultiCam</i> of the CommonMotionTypes library. Refer to the CommonMotionTypes library guide for details.

Input	Data type	Description
<i>BufferMode</i>	MC_Buffer_Mode, page 21	Default value: <i>Aborting</i> Buffer mode. Possible values: - Value <i>Aborting</i> - Value <i>Buffered</i> See <i>MC_Buffer_Mode</i> for a description of the values.
<i>Periodic</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. TRUE starts periodic mode for <i>MC_CamIn</i>. This mode repeats the execution of the cam on a continuous basis. FALSE starts the cam in single-shot mode. The slave axis position of the closest edge (first or last cam point) is frozen if it is outside the defined range, that is, the slave axis is at a standstill (but still in the state <i>SynchronizedMotion</i>) if the cam is outside the defined range. NOTE: Regardless of whether a cam is started in periodic or single-shot mode, it signals <i>EndOfProfile</i>, and a buffered motion job (if such a job exists) becomes active when <i>EndOfProfile</i> is reached (even if the cam is defined as periodic).
<i>MasterStartMode</i>	MC_Master_Start_Mode, page 23	Default value: <i>Absolute</i> Possible values: - Value <i>Absolute</i> - Value <i>Relative</i> See <i>MC_Master_Start_Mode</i> for a description of the values.
<i>SlaveStartMode</i>	MC_Slave_Start_Mode, page 23	Default value: <i>Relative</i> Possible values: - Value <i>Relative</i> - Value <i>RampIn</i> - Value <i>Absolute</i> See <i>MC_Slave_Start_Mode</i> for a description of the values.
<i>RampInDirection</i>	MC_Direction, page 22	Direction of ramp-in for coupling if the slave axis is a modulo axis. The direction is the direction to the absolute target of the ramp-in mechanism (where <i>MC_CamIn</i> is considered to be <i>InSync</i>) from the position of the slave axis, not the Y period of the cam profile. If the slave axis is not a modulo axis, the values for this input have no effect. Default value: <i>PositiveDirection</i> Possible values: - Value <i>PositiveDirection</i>: - Value <i>NegativeDirection</i> - Value <i>ShortestWay</i> See <i>MC_Direction</i> for a description of the values.
<i>VelocityOffsetRampIn</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Velocity offset for ramp-in mechanism in user-defined units.

Input	Data type	Description
<i>AccelerationOffsetRampIn</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Acceleration offset for ramp-in mechanism in user-defined units.
<i>DecelerationOffsetRampIn</i>	LREAL	Value range: Any positive LREAL value Default value: 0 Deceleration offset for ramp-in mechanism in user-defined units.
<i>JerkOffsetRampIn</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0
<i>InterpolationPoints</i>	POINTER TO BYTE	Memory address of an array with a length from 3 up to 10,000. Array type depends on value of <i>etInterpolationMode</i> for the input <i>InterpolationParameter</i> , either <i>ARRAY OF LREAL</i> or <i>ARRAY OF ST_InterpolationPointXYVA</i> . Value range: 0 and 3 ... 10000 Default value: 0 NOTE: The value must be the same as for <i>udiNumCamPoints</i> of <i>ST_InterpolationParameter</i> used by the input <i>InterpolationParameter</i> . Refer to the MotionInterface library guide for details.
<i>InterpolationParameter</i>	MC_Interpolation_Parameter	Uses <i>MC_InterpolationParameter</i> for parameterization of an interpolated cam. Refer to <i>MC_InterpolationParameter</i> for details.
<i>MasterStartPosition</i>	LREAL	Value range: -2147483648...2147483647 Default value: 0 Position of master (as seen by slave) of a previous cam when a new cam becomes active. This input is ignored unless <i>StartAtMasterPosition</i> is used for <i>MC_BufferMode</i> , page 21.

Outputs

Output	Data type	Description
<i>InSync</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: If the axes are not coupled and the cam is not processed. - TRUE: If the axes are coupled and the cam is processed.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.

Output	Data type	Description
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.
<i>EndOfProfile</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The last segment of the cam has not been completed. - TRUE: After the last segment of the cam has been completed.
<i>RampInDuration</i>	TIME	Indicates the time remaining until the ramp-in procedure is completed and the output <i>InSync</i> is set to TRUE.

Notes

As opposed to the specifications of PLCopen Motion Control Part 1, Version 2.0, the library does not provide a separate function block *MC_CamTableSelect*. The cam table is specified as an input of *MC_CamIn*.

The library does not provide a separate function block *MC_CamOut*. A running function block can be replaced by any other function block.

This function block provides high flexibility for both absolute and relative movements. For example, there is not necessarily a relation between the modulo of a master (or slave) axis and the application period of a cam in X (or Y) direction. Therefore, offset corrections can be applied on the fly by slightly adjusting the application period of the cam profile in X or Y direction. This would not be possible with the axis modulo which cannot be modified while the axis executes a function block.

MC_GearIn

Functional Description

This function block activates coupling of a master axis and a slave axis with a given gear factor between the position or velocity of the master axis and the slave axis, depending on the operating mode.

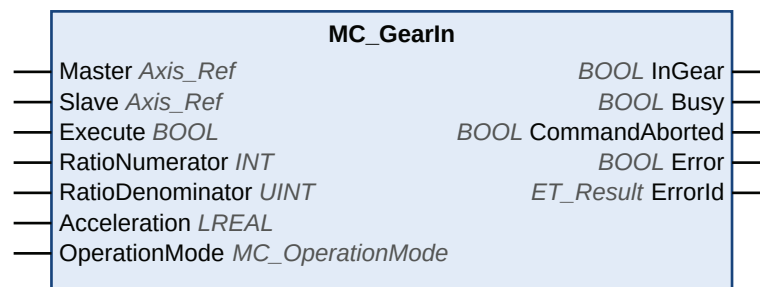
The slave axis synchronously follows the movement of the master axis (position or velocity synchronicity).

The inputs *RatioNumerator* and *RatioDenominator* let you set a user-specific gear ratio for the movement of the slave axis.

When the output *InGear* is set to TRUE, the operating mode set via the input *OperationMode* determines the type of coupling:

- In the operating mode *Cyclic Synchronous Position*, the coupling is performed based on position values. For example, with a gear ratio of 1:2, the slave moves half of the distance of the master.
- In the operating mode *Cyclic Synchronous Velocity*, the coupling is performed based on velocity values. For example, with a gear ratio of 1:2, the slave moves at half of the velocity of the master.

Graphical Representation



Inputs

Input	Data type	Description
<i>Master</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Slave</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>RatioNumerator</i>	INT	Value range: -2147483648...2147483647 Default value: 1 Numerator of gear ratio. NOTE: The value 0 is invalid.
<i>RatioDenominator</i>	UINT	Value range: 1...2147489647 Default value: 1 Denominator of gear ratio.

Input	Data type	Description
<i>Acceleration</i>	LREAL	The value at this input is used to reach the specified target velocity (acceleration or deceleration). Value range: Any positive LREAL value Default value: 0 Acceleration in user-defined units.
<i>OperationMode</i>	MC_ OperationMode, page 23	Operating mode for function block Default value: Position

Outputs

Output	Data type	Description
<i>InGear</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The adjusted gear ratio is not reached. - TRUE: When the adjusted gear ratio is reached.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.

Notes

The input *Acceleration* needs to be set to a value greater than 0 before the function block is executed.

The gear ratio can be modified during a movement. However, the new values are taken into account only with the next rising edge of the input *Execute*.

The slave axis uses the values for *Acceleration* and *Jerk* only during the first acceleration phase. The slave axis then follows the master axis.

If the operating mode is set to Velocity via the input *OperationMode* and if the drive is not able to operate in the operating mode Cyclic Synchronous Velocity, the function block *MC_CamIn* detects an error. The axis is not affected.

The library does not provide a separate function block *MC_GearOut*. A running function block can be replaced by any other function block.

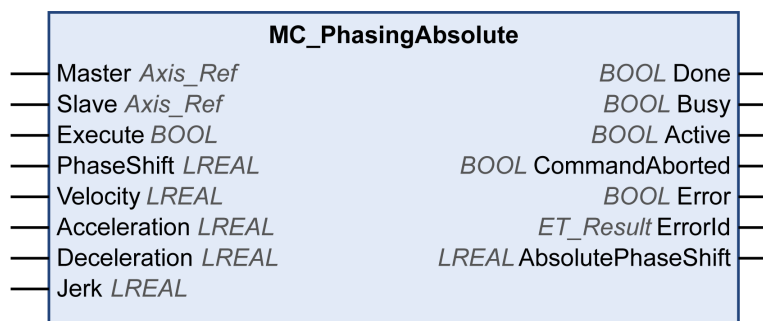
MC_PhasingAbsolute

Functional Description

This function block creates a position offset between the position of a master axis and the position of this master axis as it is seen by the slave axis.

The function block *MC_PhasingAbsolute* requires a function block *MC_CamIn* to be active for the specified slave axis. The master axis must be identical to the master axis of the active function block *MC_CamIn*.

Graphical Representation



Inputs

Input	Data type	Description
<i>Master</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Slave</i>	Axis_Ref	Reference to the axis for which the function block is to be executed.
<i>Execute</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. A rising edge of the input <i>Execute</i> starts the function block. The function block continues execution and the output <i>Busy</i> is set to TRUE. This function block can be restarted while it is executed. The target values are overwritten by the new values at the point in time the rising edge occurs.
<i>PhaseShift</i>	LREAL	Value range: 1...2147483647 Default value: 1 Phase shift in user-defined units.
<i>Velocity</i>	LREAL	Value range: 1...2147489647 Default value: 1 Velocity in user-defined units.
<i>Acceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 1 Acceleration in user-defined units.

Input	Data type	Description
<i>Deceleration</i>	LREAL	Value range: Any positive LREAL value Default value: 1 Deceleration in user-defined units.
<i>Jerk</i>	LREAL	Value range: Any positive LREAL value and zero - Positive values: Jerk limit (in units/s³) (maximum jerk with which the acceleration is modified). - Zero: Jerk limit disabled. The acceleration instantaneously jumps from zero to maximum acceleration (infinite jerk). Default value: 0

Outputs

Output	Data type	Description
<i>Done</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been started, or an error has been detected. - TRUE: Execution terminated without an error detected.
<i>Busy</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is not being executed. - TRUE: Function block is being executed.
<i>Active</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: The function block does not control the movement of the axis. - TRUE: The function block controls the movement of the axis.
<i>CommandAborted</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Execution has not been aborted. - TRUE: Execution has been aborted by another function block.
<i>Error</i>	BOOL	Value range: FALSE, TRUE. Default value: FALSE. - FALSE: Function block is being executed, no error has been detected during execution. - TRUE: An error has been detected in the execution of the function block.
<i>ErrorID</i>	ET_Result, page 28	This enumeration provides diagnostics information.
<i>AbsolutePhaseShift</i>	LREAL	Provides the current phase shift.

Migration Information SoftMotion to PLCopen

Migration Information SoftMotion to PLCopen

Overview

The following sections provide information intended to facilitate migrating from SoftMotion libraries (also based on PLCopen) to the libraries PLCopen MC part 1 and MotionInterface.

Note 1: Buffer Modes

The Modicon M262 Motion Controller supports the buffer modes buffered and blending in addition to aborting. Refer to the description of the `data type`, page 21 for details.

Note 2: Task Concept

The controller runs the user application in which motion control function blocks are called in a task separate from the real-time motion task in which motion profiles are calculated and Sercos communication takes place. If you want to start a function block in the same Sercos cycle in which the preceding function block reaches its steady state, you need to buffer it ahead of time. Refer to the chapter Task Concept, page 14 for details.

Note 3: Unavailable POU

POUs from SoftMotion which are not defined by PLCopen Motion Control Part 1, Version 2.0 are not available for M262 (that is, POUs with the prefixes `SMC_` or `SMC3_` or most function blocks which do not have the prefix `MC_`). Typically, the motion functionality originally implemented by such POUs can be implemented with alternative means in the new libraries.

Note 4: Reading Device and Axis Parameters

For reading of device and axis parameters, the M262 relies less on function blocks and more on parameters, properties, and methods of device objects and of the `Axis_Ref.`, page 14

Note 5: Homing and Absolute Movements

As defined by PLCopen Motion Control Part 1, Version 2.0, absolute movements are only possible with a homed axis. This includes cams for which the slave axis start mode *Absolute* is only permitted if the slave axis is homed and the master axis start mode *Absolute* is only permitted if the master axis is homed. An axis is considered to be homed after `MC_Home` or an absolute `MC_SetPosition` have been executed for the axis. Alternatively, the controller application can set the flag *isHomed* to TRUE via the code (verify that the absolute position of the axis is correct by appropriate means). Refer to Absolute Position, Homing, and Absolute Movements, page 15 for details.

Note 6: Specific Information on Individual Function Blocks

The following table provides information on SoftMotion SM3_Basic function blocks for master and slave axes and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>MC_CamIn</i>	<i>MC_CamIn</i>	Master start mode and slave start mode are specified separately. The cam table is specified directly as an input (<i>MC_CamTableSelect</i> is not required). Also refer to Note 2, page 83 and to the description of the function block, page 72 for details.
<i>MC_CamOut</i>	-	This function block is not required to terminate a running <i>MC_CamIn</i> . It is sufficient to execute the new function block for the axis while an <i>MC_CamIn</i> is running. Continuing a movement with the current velocity without an active function block is not available.
<i>MC_CamTableSelect</i>	-	Not required. Cam table and master axis/slave axis start modes are directly supplied as inputs for <i>MC_CamIn</i> .
<i>MC_GearIn</i>	<i>MC_GearIn</i>	-
<i>MC_GearInPos</i>	-	Use <i>MC_CamIn</i> with a straight line instead.
<i>MC_GearOut</i>	-	This function block is not required to terminate a running <i>MC_GearIn</i> . It is sufficient to execute the new function block for the axis while an <i>MC_GearIn</i> is running.
<i>MC_Phasing</i>	<i>MC_PhasingAbsolute</i>	The implementation for M262 complies with PLCopen Motion Control Part 1, Version 2.0 and thus differs from the implementation in SM3_Basic. No extra axis is required for phasing, but phasing is a motion component of the slave axis.

The following table provides information on SoftMotion SM3_Basic single axis function blocks and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>MC_AccelerationProfile</i>	-	Not available
<i>MC_Halt</i>	<i>MC_Halt</i>	-
<i>MC_Home</i>	<i>MC_Home</i>	-
<i>MC_MoveAbsolute</i>	<i>MC_MoveAbsolute</i>	-
<i>MC_MoveAdditive</i>	<i>MC_MoveAdditive</i>	-
<i>MC_MoveRelative</i>	<i>MC_MoveRelative</i>	-
<i>MC_MoveSuperImposed</i>	<i>MC_MoveSuperImposed</i>	-
<i>MC_MoveVelocity</i>	<i>MC_MoveVelocity</i>	-
<i>MC_PositionProfile</i>	-	Not available
<i>MC_Power</i>	<i>MC_Power</i>	-
<i>MC_ReadActualPosition</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_ReadAxisError</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.

SoftMotion LMC0x8	M262 equivalent	Remark
<i>MC_ReadBoolParameter</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_ReadParameter</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_ReadStatus</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_Reset</i>	<i>MC_Reset</i>	-
<i>MC_Stop</i>	<i>MC_Stop</i>	-
<i>MC_VelocityProfile</i>	-	Not available.
<i>MC_WriteBoolParameter</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_WriteParameter</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.

The following table provides information on SoftMotion SM3_Basic single axis function blocks and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>MC_AbortTrigger</i>	<i>MC_AbortTrigger</i>	-
<i>MC_DigitalCamSwitch</i>	-	Not available
<i>MC_ReadActualTorque</i>	-	Not available
<i>MC_ReadActualVelocity</i>	-	Refer to Note 4: Reading Device and Axis Parameters, page 83.
<i>MC_SetPosition</i>	<i>MC_SetPosition</i>	-
<i>MC_TouchProbe</i>	<i>MC_TouchProbe</i>	-
<i>SMC_MoveContinuousAbsolute</i>	Refer to comments	Blend an <i>MC_MoveVelocity</i> after an <i>MC_MoveAbsolute</i> .
<i>SMC_MoveContinuousRelative</i>	Refer to comments	Blend an <i>MC_MoveVelocity</i> after an <i>MC_MoveAbsolute</i> .

The following table provides information on SoftMotion SM3_Basic single axis function blocks and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>MC_Jog</i>	-	Not available. Refer to the sample code in PLCopen MC part 3 for information on how to implement the functionality.

The following table provides information on miscellaneous feature of SoftMotion SM3_Basic and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>AXIS_REF_SM3</i>	<i>Axis_Ref</i>	Internal structure is different, refer to <i>Axis_Ref</i> , page 14 for details
<i>AXIS_REF_VIRTUAL_SM3</i>	<i>FB_ControlledAxis</i>	Internal structure is different, refer to <i>Axis_Ref</i> , page 14 for details

The following table provides information on miscellaneous feature of SoftMotion SM3_CNC and their equivalents in M262:

SoftMotion LMC0x8	M262 equivalent	Remark
<i>SMC_ControlAxisByPos</i>	<i>MC_CustomJob</i>	Handling is different due to task concept. Refer to description of function block, page 36 for details.
<i>SMC_ControlAxisByPosVel</i>		
<i>SMC_ControlAxisByVel</i>		

Glossary

A

%:

According to the IEC standard, % is a prefix that identifies internal memory addresses in the logic controller to store the value of program variables, constants, I/O, and so on.

application:

A program including configuration data, symbols, and documentation.

ARRAY:

The systematic arrangement of data objects of a single type in the form of a table defined in logic controller memory. The syntax is as follows: ARRAY

[<dimension>] OF <Type>

Example 1: ARRAY [1..2] OF BOOL is a 1-dimensional table with 2 elements of type BOOL.

Example 2: ARRAY [1..10, 1..20] OF INT is a 2-dimensional table with 10 x 20 elements of type INT.

B

BOOL:

(*boolean*) A basic data type in computing. A BOOL variable can have one of these values: 0 (FALSE), 1 (TRUE). A bit that is extracted from a word is of type BOOL; for example, %MW10.4 is a fifth bit of memory word number 10.

Boot application:

(*boot application*) The binary file that contains the application. Usually, it is stored in the controller and allows the controller to boot on the application that the user has generated.

BOOTP:

(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address (and possibly other data) from a server. The client identifies itself to the server using the client MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its pre-configured IP address. BOOTP was originally used as a method that enabled diskless hosts to be remotely booted over a network. The BOOTP process assigns an infinite lease of an IP address. The BOOTP service utilizes UDP ports 67 and 68.

byte:

A type that is encoded in an 8-bit format, ranging from 00 hex to FF hex.

C

CAN:

(*controller area network*) A protocol (ISO 11898) for serial bus networks, designed for the interconnection of smart devices (from multiple manufacturers) in smart systems and for real-time industrial applications. Originally developed for use in automobiles, CAN is now used in a variety of industrial automation control environments.

CFC:

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of

graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

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.

control network:

A network containing logic controllers, SCADA systems, PCs, HMI, switches, ...

Two kinds of topologies are supported:

- flat: all modules and devices in this network belong to same subnet.
- 2 levels: the network is split into an operation network and an inter-controller network.

These two networks can be physically independent, but are generally linked by a routing device.

CRC:

(cyclical redundancy check) A method used to determine the validity of a communication transmission. The transmission contains a bit field that constitutes a checksum. The message is used to calculate the checksum by the transmitter according to the content of the message. Receiving nodes, then recalculate the field in the same manner. Any discrepancy in the value of the 2 CRC calculations indicates that the transmitted message and the received message are different.

D

DHCP:

(dynamic host configuration protocol) An advanced extension of BOOTP. DHCP is more advanced, but both DHCP and BOOTP are common. (DHCP can handle BOOTP client requests.)

DWORD:

(double word) Encoded in 32-bit format.

E

EtherNet/IP:

(Ethernet industrial protocol) An open communications protocol for manufacturing automation solutions in industrial systems. EtherNet/IP is in a family of networks that implement the common industrial protocol at its upper layers. The supporting organization (ODVA) specifies EtherNet/IP to accomplish global adaptability and media independence.

Ethernet:

A physical and data link layer technology for LANs, also known as IEEE 802.3.

F

FB:

(function block) A convenient programming mechanism that consolidates a group of programming instructions to perform a specific and normalized action, such as speed control, interval control, or counting. A function block may comprise configuration data, a set of internal or external operating parameters and usually 1 or more data inputs and outputs.

firmware:

Represents the BIOS, data parameters, and programming instructions that constitute the operating system on a controller. The firmware is stored in non-volatile memory within the controller.

function block diagram:

One of the 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks where each network contains a graphical structure of boxes and connection lines representing either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

function block:

A programming unit that has 1 or more inputs and returns 1 or more outputs. FBs are called through an instance (function block copy with dedicated name and variables) and each instance has a persistent state (outputs and internal variables) from 1 call to the other.

Examples: timers, counters

function:

A programming unit that has 1 input and returns 1 immediate result. However, unlike FBs, it is directly called with its name (as opposed to through an instance), has no persistent state from one call to the next and can be used as an operand in other programming expressions.

Examples: boolean (AND) operators, calculations, conversions (BYTE_TO_INT)

G

GVL:

(*global variable list*) Manages global variables within an EcoStruxure Machine Expert project.

H

hex:

(*hexadecimal*)

I

I/O:

(*input/output*)

ID:

(*identifier/identification*)

IEC 61131-3:

Part 3 of a 3-part IEC standard for industrial automation equipment. IEC 61131-3 is concerned with controller programming languages and defines 2 graphical and 2 textual programming language standards. The graphical programming languages are ladder diagram and function block diagram. The textual programming languages include structured text and instruction list.

IEC:

(*international electrotechnical commission*) A non-profit and non-governmental international standards organization that prepares and publishes international standards for electrical, electronic, and related technologies.

IEEE 802.3:

A collection of IEEE standards defining the physical layer, and the media access control sublayer of the data link layer, of wired Ethernet.

IL:

(*instruction list*) A program written in the 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 (refer to IEC 61131-3).

INT:

(*integer*) A whole number encoded in 16 bits.

IP:

(*Internet protocol*) Part of the TCP/IP protocol family that tracks the Internet addresses of devices, routes outgoing messages, and recognizes incoming messages.

L

LD:

(*ladder diagram*) 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 (refer to IEC 61131-3).

legacy projects:

Application projects that were created with SoMachine, SoMachine Motion, or a previous version of EcoStruxure Machine Expert.

LWORD:

(*long word*) A data type encoded in a 64-bit format.

M

MAC address:

(*media access control address*) A unique 48-bit number associated with a specific piece of hardware. The MAC address is programmed into each network card or device when it is manufactured.

MAST:

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

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

Modbus:

The protocol that allows communications between many devices connected to the same network.

%MW:

According to the IEC standard, %MW represents a memory word register (for example, a language object of type memory word).

N

network:

A system of interconnected devices that share a common data path and protocol for communications.

NVM:

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

P

PLC:

(*programmable logic controller*) An industrial computer used to automate manufacturing, industrial, and other electromechanical processes. PLCs are different from common computers in that they are designed to have multiple input and output arrays and adhere to more robust specifications for shock, vibration, temperature, and electrical interference among other things.

POU:

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

program:

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

protocol:

A convention or standard definition that controls or enables the connection, communication, and data transfer between 2 computing system and devices.

R

run:

A command that causes the controller to scan the application program, read the physical inputs, and write to the physical outputs according to solution of the logic of the program.

S

STOP:

A command that causes the controller to stop running an application program.

string:

A variable that is a series of ASCII characters.

ST:

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

system variable:

A variable that provides controller data and diagnostic information and allows sending commands to the controller.

T

task:

A group of sections and subroutines, executed cyclically or periodically for the MAST task or periodically for the FAST task.

A task possesses a level of priority and is linked to inputs and outputs of the controller. These I/O are refreshed in relation to the task.

A controller can have several tasks.

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**UDINT:**

(*unsigned double integer*) Encoded in 32 bits.

UINT:

(*unsigned integer*) Encoded in 16 bits.

unlocated variable:

A variable that does not have an address (refer to *located variable*).

V**variable:**

A memory unit that is addressed and modified by a program.

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.

WORD:

A type encoded in a 16-bit format.

Index

A

axis configuration	15
Axis_Ref	14

C

common inputs and outputs	
behavior of function blocks with the input	
<i>Execute</i>	26

E

ET_Result	28

F

FB_ControlledAxis	14
finite axis	15

G

general description	
libraries and related function blocks	11
PLCopen state diagram	17

I

initialization	59

L

linear axis	15

M

<i>MC_AbortTrigger</i>	34
<i>MC_CamIn</i>	72
<i>MC_CustomJob</i>	36
<i>MC_GearIn</i>	79
<i>MC_Halt</i>	39
<i>MC_Home</i>	42
<i>MC_MoveAbsolute</i>	44
<i>MC_MoveAdditive</i>	47
<i>MC_MoveRelative</i>	50
<i>MC_MoveSuperImposed</i>	53
<i>MC_MoveVelocity</i>	56
<i>MC_PhasingAbsolute</i>	81
<i>MC_Power</i>	59
<i>MC_Reset</i>	61
<i>MC_SetPosition</i>	63
<i>MC_Stop</i>	65
<i>MC_TorqueControl</i>	67
<i>MC_TouchProbe</i>	70
modulo axis	15

T

task concept	14

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2021 – Schneider Electric. All rights reserved.

EIO0000003871.03

Modicon M262

Logic/Motion Controller

Hardware Guide

EIO0000003659.08

06/2022



Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

As part of a group of responsible, inclusive companies, we are updating our communications that contain non-inclusive terminology. Until we complete this process, however, our content may still contain standardized industry terms that may be deemed inappropriate by our customers.

© 2022 – Schneider Electric. All rights reserved.

Table of Contents

Safety Information.....	5
Qualification of Personnel	5
Intended Use.....	6
About the Book.....	7
M262 General Overview	12
M262 General Overview.....	12
M262 Logic/Motion Controller Description.....	12
Maximum Hardware Configuration	15
TM3 Expansion Modules	17
TM3 Bus Couplers	26
TM5 Fieldbus Interfaces	26
TM5 CANopen Fieldbus Interfaces.....	27
TM7 CANopen Fieldbus Interfaces.....	27
TMS Expansion Modules.....	28
Accessories.....	28
M262 Features	30
Real Time Clock (RTC).....	30
Input Management.....	30
Output Management	32
Run/Stop.....	32
SD Card	33
Alarm Relay.....	37
M262 Installation	39
M262 Logic/Motion Controller General Rules for Implementing	39
Environmental Characteristics	39
Certifications and Standards.....	41
M262 Logic/Motion Controller Installation	42
Installation and Maintenance Requirements.....	42
M262 Logic/Motion Controller Mounting Positions and Clearances.....	43
Top Hat Section Rail (DIN rail).....	46
Installing and Removing the Controller with Expansions.....	49
Mounting a M262 Logic/Motion Controller on a Panel Surface	51
M262 Electrical Requirements	52
Wiring Best Practices	52
DC Power Supply Characteristics and Wiring	56
Grounding the M262 Logic/Motion Controller System	58
Alarm Relay Wiring	64
Modicon M262 Logic/Motion Controller.....	66
TM262L01MESE8T Presentation	66
TM262L10MESE8T Presentation	70
TM262L20MESE8T Presentation	75
TM262M05MESS8T Presentation.....	80
TM262M15MESS8T Presentation.....	85
TM262M25MESS8T Presentation.....	90
TM262M35MESS8T Presentation.....	95
Embedded I/O Channels	101

Digital Inputs	101
Digital Outputs.....	104
Encoder Interface.....	109
Encoder Interface	109
Integrated Communication Ports.....	114
Ethernet 1 Port	114
Ethernet 2 Ports	116
USB Mini-B Programming Port	118
Serial Line.....	119
Connecting the M262 Logic/Motion Controller to a PC	122
Connecting the Controller to a PC	122
Glossary	125
Index	131

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a "Danger" or "Warning" safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

Qualification of Personnel

Only appropriately trained persons who are familiar with and understand the contents of this manual and all other pertinent product documentation are authorized to work on and with this product.

The qualified person must be able to detect possible hazards that may arise from parameterization, modifying parameter values and generally from mechanical, electrical, or electronic equipment. The qualified person must be familiar with the standards, provisions, and regulations for the prevention of industrial accidents, which they must observe when designing and implementing the system.

Intended Use

The products described or affected by this document are, together with software, accessories and options, programmable logic controllers (referred to herein as “controllers”), intended for industrial use according to the instructions, directions, examples and safety information contained in the present document and other supporting documentation.

The product may only be used in compliance with all applicable safety regulations and directives, the specified requirements and the technical data.

Prior to using the product, you must perform a risk assessment in view of the planned application. Based on the results, the appropriate safety-related measures must be implemented.

Since the product is used as a component in an overall machine or process, you must ensure the safety of persons by means of the design of this overall system.

Operate the product only with the specified cables and accessories. Use only genuine accessories and spare parts.

Any use other than the use explicitly permitted is prohibited and can result in unanticipated hazards.

About the Book

Document Scope

Use this document to:

- Familiarize yourself with the features of the M262 Logic/Motion Controller.
- Install and operate your M262 Logic/Motion Controller.
- Interface the M262 Logic/Motion Controller with I/O expansion modules and other devices.
- Connect the M262 Logic/Motion Controller to a programming device equipped with EcoStruxure Machine Expert software.

NOTE: Read and understand this document and all related documents, page 7 before installing, operating, or maintaining your controller.

Validity Note

This document has been updated for the release of EcoStruxure™ Machine Expert V2.0.3.

The characteristics that are described in the present document, as well as those described in the documents included in the Related Documents section below, can be found online. To access the information online, go to the Schneider Electric home page www.se.com/ww/en/download/.

The characteristics that are described in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

For product compliance and environmental information (RoHS, REACH, PEP, EOL, etc.), go to www.se.com/ww/en/work/support/green-premium/.

Related Documents

Title of Documentation	Reference Number
Modicon M262 Logic/Motion Controller - Programming Guide	EIO0000003651 (ENG)
	EIO0000003652 (FRA)
	EIO0000003653 (GER)
	EIO0000003654 (SPA)
	EIO0000003655 (ITA)
	EIO0000003656 (CHS)
	EIO0000003657 (POR)
Modicon TM3 Digital I/O Modules - Hardware Guide	EIO0000003658 (TUR)
	EIO0000003125 (ENG)
	EIO0000003126 (FRE)
	EIO0000003127 (GER)
	EIO0000003128 (SPA)
	EIO0000003129 (ITA)
	EIO0000003130 (CHS)
	EIO0000003424 (TUR)
	EIO0000003425 (POR)

Title of Documentation	Reference Number
Modicon TM3 Analog I/O Modules - Hardware Guide	EIO0000003131 (ENG) EIO0000003132 (FRE) EIO0000003133 (GER) EIO0000003134 (SPA) EIO0000003135 (ITA) EIO0000003136(CHS) EIO0000003426 (POR) EIO0000003427 (TUR)
Modicon TM3 Expert I/O Modules - Hardware Guide	EIO0000003137 (ENG) EIO0000003138 (FRE) EIO0000003139 (GER) EIO0000003140 (SPA) EIO0000003141 (ITA) EIO0000003142 (CHS) EIO0000003428 (POR) EIO0000003429 (TUR)
Modicon TM3 Safety Modules - Hardware Guide	EIO0000003353 (ENG) EIO0000003354 (FRE) EIO0000003355 (GER) EIO0000003356 (SPA) EIO0000003357 (ITA) EIO0000003358 (CHS) EIO0000003359 (POR) EIO0000003360 (TUR)
Modicon TM3 Transmitter and Receiver Modules - Hardware Guide	EIO0000003143 (ENG) EIO0000003144 (FRE) EIO0000003145 (GER) EIO0000003146 (SPA) EIO0000003147 (ITA) EIO0000003148 (CHS) EIO0000003430 (POR) EIO0000003431 (TUR)
Modicon TM3 Bus Coupler Module - Hardware Guide	EIO0000003635 (ENG) EIO0000003636 (FRE) EIO0000003637 (GER) EIO0000003638 (SPA) EIO0000003639 (ITA) EIO0000003640 (CHS) EIO0000003641 (POR) EIO0000003642 (TUR)

Title of Documentation	Reference Number
Modicon TM5 Fieldbus Interface - Hardware Guide	EIO0000003715 (ENG) EIO0000003716 (FRE) EIO0000003717 (GER) EIO0000003718 (SPA) EIO0000003719 (ITA) EIO0000003720 (CHS)
Modicon TMS Expansion Modules - Hardware Guide	EIO0000003699 (ENG) EIO0000003700 (FRA) EIO0000003701 (GER) EIO0000003702 (SPA) EIO0000003703 (ITA) EIO0000003704 (CHS) EIO0000003705 (POR) EIO0000003706 (TUR)
EcoStruxure Machine Expert Industrial Ethernet Overview	EIO0000003053 (ENG) EIO0000003054 (FRE) EIO0000003055 (GER) EIO0000003056 (SPA) EIO0000003057 (ITA) EIO0000003058 (CHS) EIO0000003816 (POR) EIO0000003817 (TUR)
M262 Logic/Motion Controller - Instruction Sheet	HRB59604

You can download these technical publications and other technical information from our website at www.se.com/ww/en/download/.

Product Related Information

⚠️⚠️ DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

⚠ DANGER**POTENTIAL FOR EXPLOSION**

- Only use this equipment in non-hazardous locations, or in locations that comply with Class I, Division 2, Groups A, B, C and D.
- Do not substitute components which would impair compliance to Class I Division 2.
- Do not connect or disconnect equipment unless power has been removed or the location is known to be non-hazardous.

Failure to follow these instructions will result in death or serious injury.

⚠ WARNING**LOSS OF CONTROL**

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ WARNING**UNINTENDED EQUIPMENT OPERATION**

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

M262 General Overview

Overview

This chapter provides general information about the M262 Logic/Motion Controller system architecture and its components.

M262 General Overview

M262 Logic/Motion Controller Description

Overview

The M262 Logic/Motion Controller has various powerful features and can service a wide range of applications.

Software configuration, programming, and commissioning are accomplished with the EcoStruxure Machine Expert software version 1.1 or later, described in detail in the EcoStruxure Machine Expert Programming Guide as well as the present document.

Programming Languages

The M262 Logic/Motion Controller is configured and programmed with the EcoStruxure Machine Expert software, which supports the following IEC 61131-3 programming languages:

- IL: Instruction List
- ST: Structured Text
- FBD: Function Block Diagram
- SFC: Sequential Function Chart
- LD: Ladder Diagram

EcoStruxure Machine Expert software can also be used to program these controllers using CFC (Continuous Function Chart) language.

Power Supply

The power supply of the M262 Logic/Motion Controller is 24 Vdc, page 56.

Real Time Clock

The M262 Logic/Motion Controller includes a Real Time Clock (RTC) system, page 30.

The system time is maintained by capacitors when the power is off. The time is maintained for 1 000 hours when the controller is not supplied.

Run/Stop

The M262 Logic/Motion Controller can be operated externally by the following:

- A hardware Run/Stop switch, page 32.

- A Run/Stop, page 30 operation by a dedicated digital input, defined in the software configuration. For more information, refer to Configuration of Digital Inputs (see Modicon M262 Logic/Motion Controller, Programming Guide).
- An EcoStruxure Machine Expert software command.
- The system variable PLC_W in a Relocation Table.
- The Web server.

Memory

This table describes the different types of memory:

Memory Type	Size	Use
RAM	256 Mbytes, of which 32 Mbytes are available for the application	For the execution of the application and the firmware.
Flash	1 Gbyte	Non-volatile memory dedicated to the retention of the program and data in case of a power interruption.
Non-volatile RAM	512 Kbytes	Non-volatile memory dedicated to the retention of the retain-persistent variables, and the diagnostic files and associated information.

Embedded Inputs/Outputs

The following embedded I/O types are available:

- Fast inputs
- Fast source outputs

Encoder

The following encoder modes are available:

- Incremental mode
- SSI mode

Removable Storage

The M262 Logic/Motion Controllers include an integrated SD card slot, page 33.

The main uses of the SD card are:

- Initializing the controller with a new application
- Updating the controller and expansion module firmware (see Modicon M262 Logic/Motion Controller, Programming Guide)
- Applying post configuration files to the controller (see Modicon M262 Logic/Motion Controller, Programming Guide)
- Storing recipes, files
- Receiving data logging files

Embedded Communication Features

The following types of communication ports are available:

- Ethernet, page 116
- USB Mini-B, page 118

- Serial Line, page 119
- Sercos (Ethernet 1), page 115

Expansion Module and Bus Coupler Compatibility

Refer to the compatibility tables in the EcoStruxure Machine Expert - Compatibility and Migration User Guide.

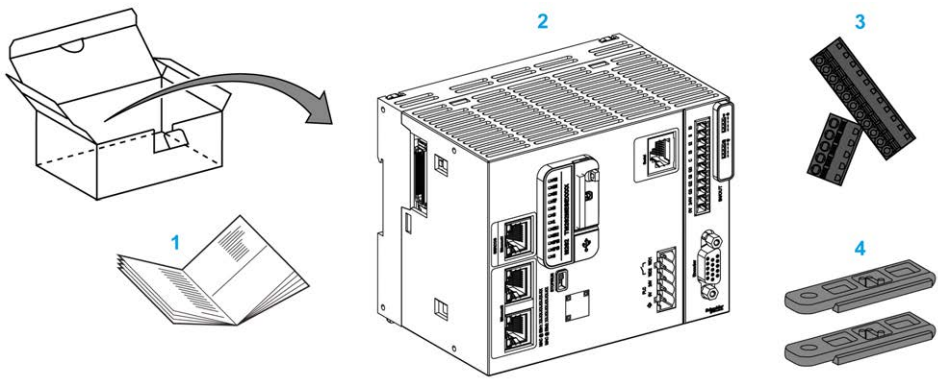
M262 Logic/Motion Controller

Reference	Digital I/O	Power supply	Communication Ports	Terminal Type	Encoder
M262 Logic Controller: TM262L•	4 fast inputs Source outputs 4 fast outputs	24 Vdc	1 serial line port 1 USB programming port 1 Ethernet port 1 dual port Ethernet switch	Removable spring	–
M262 Motion Controller: TM262M•	4 fast inputs Source outputs 4 fast outputs	24 Vdc	1 serial line port 1 USB programming port 1 Ethernet port for fieldbus with Sercos interface 1 dual port Ethernet switch	Removable spring	1 Encoder port

NOTE: You can use the fast inputs/outputs as regular inputs/outputs.

Delivery Content

The following figure presents the content of the delivery for the M262 Logic/Motion Controller:



- 1 M262 Logic/Motion Controller Instruction Sheet
- 2 M262 Logic/Motion Controller
- 3 Removable spring terminal blocks
- 4 Attachment parts

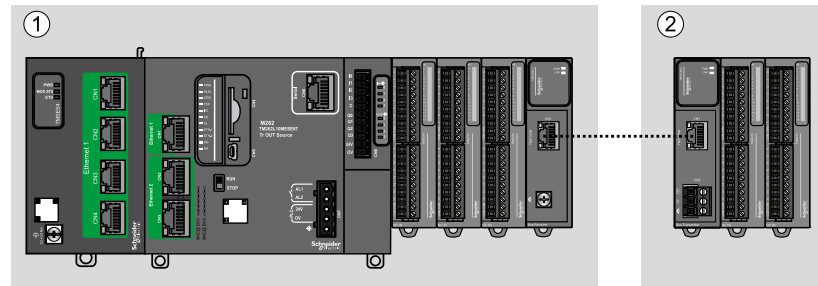
Maximum Hardware Configuration

Introduction

The M262 Logic/Motion Controller is a control system that offers an all-in-one solution for motion applications and a scalable solution for logic applications, with optimized configurations and an open, expandable architecture.

Local and Remote Configuration Principle

The following figure defines the local and remote configurations:



(1) Local configuration

(2) Remote configuration

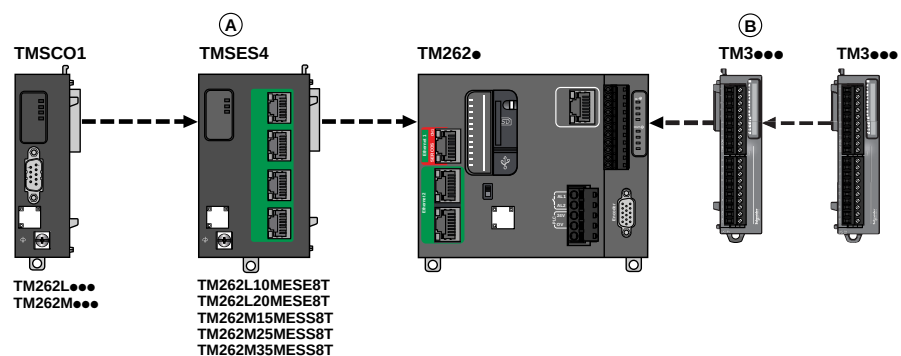
M262 Logic/Motion Controller Local Configuration Architecture

Optimized local configuration and flexibility are provided by the association of:

- M262 Logic/Motion Controller
- TMS expansion modules
- TM3 expansion modules

Application requirements determine the architecture of your M262 Logic/Motion Controller configuration.

The following figure represents the components of a local configuration:



(A) TMS expansion modules.

- 1 TMSCO1 for TM262L01MESE8T and TM262M05MESS8T
- 3 TMSES4 or 2 TMSES4 and 1 TMSCO1 for the other references

TMSCO1 must be the leftmost module connected to the controller.

(B) TM3 expansion modules (7 maximum).

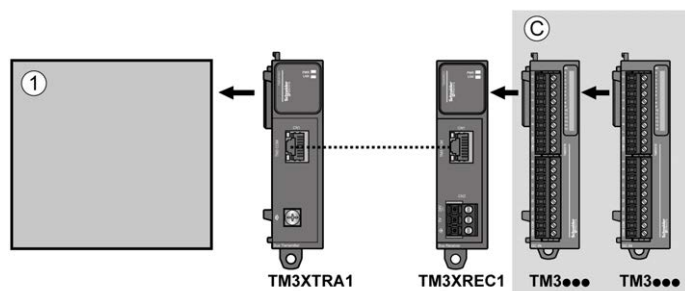
M262 Logic/Motion Controller Remote Configuration Architecture

Optimized remote configuration and flexibility are provided by the association of:

- M262 Logic/Motion Controller
- TMS expansion modules
- TM3 expansion modules
- TM3 transmitter and receiver modules

Application requirements determine the architecture of your M262 Logic/Motion Controller configuration.

The following figure represents the components of a remote configuration:



(1) Logic/motion controller and modules

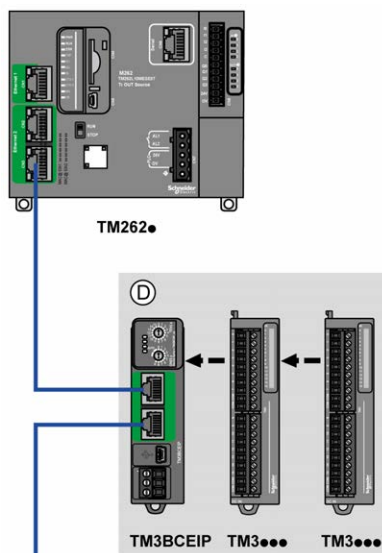
(C) TM3 expansion modules (7 maximum)

M262 Logic/Motion Controller Distributed Configuration Architecture

Optimized remote configuration and flexibility are provided by the association of:

- TM3 bus couplers, page 26
- TM5 fieldbus interface, page 26

This figure shows the components of a distributed architecture:



(D) TM3 distributed modules

Maximum Number of Modules

The following table shows the maximum configuration supported:

References	Maximum	Type of Configuration
TM262L01MESE8T TM262M05MESS8T	7 TM3 expansion modules 1 TMSCO1	Local
TM262L10MESE8T TM262M15MESS8T TM262L20MESE8T TM262M25MESS8T TM262M35MESS8T	7 TM3 expansion modules 3 TMS expansion modules composed of: - up to 3 TMSES4 - up to 1 TMSCO1	Local
TM3XREC1	7 TM3 expansion modules	Remote
TM3BCEIP TM3BCSL TM3BCCO	7 TM3 expansion modules without transmitter and receiver 14 TM3 expansion modules with transmitter and receiver	Distributed
NOTE: TM3 transmitter and receiver modules are not included in a count of the maximum number of expansion modules.		

NOTE: The configuration with its TMS and TM3 expansion modules is validated by EcoStruxure Machine Expert software in the **Configuration** window.

NOTE: In some environments, the maximum configuration populated by high power consumption modules, coupled with the maximum distance allowable between the TM3 transmitter and receiver modules, may present bus communication issues although the EcoStruxure Machine Expert software allowed for the configuration. In such a case you will need to analyze the power consumption of the modules chosen for your configuration, as well as the minimum cable distance required by your application, and possibly seek to optimize your choices.

TM3 Expansion Modules

Introduction

The range of TM3 expansion modules includes:

- Digital modules, classified as follows:
 - Input modules, page 18
 - Output modules, page 19
 - Mixed input/output modules, page 20
- Analog modules, classified as follows:
 - Input modules, page 21
 - Output modules, page 22
 - Mixed input/output modules, page 23
- Expert modules, page 24
- Safety modules, page 25
- Transmitter and Receiver modules, page 26

For more information, refer to the following documents:

- TM3 Digital I/O Modules Hardware Guide
- TM3 Analog I/O Modules Hardware Guide
- TM3 Expert I/O Modules Hardware Guide

- TM3 Safety Modules Hardware Guide
- TM3 Transmitter and Receiver Modules Hardware Guide

TM3 Digital Input Modules

The following table shows the TM3 digital input expansion modules, with corresponding channel type, nominal voltage/current, and terminal type:

Reference	Channels	Channel Type	Voltage Current	Terminal Type / Pitch
TM3DI8A	8	Regular inputs	120 Vac 7.5 mA	Removable screw terminal block / 5.08 mm
TM3DI8	8	Regular inputs	24 Vdc 7 mA	Removable screw terminal block / 5.08 mm
TM3DI8G	8	Regular inputs	24 Vdc 7 mA	Removable spring terminal block / 5.08 mm
TM3DI16	16	Regular inputs	24 Vdc 7 mA	Removable screw terminal blocks / 3.81 mm
TM3DI16G	16	Regular inputs	24 Vdc 7 mA	Removable spring terminal blocks / 3.81 mm
TM3DI16K	16	Regular inputs	24 Vdc 5 mA	HE10 (MIL 20) connector
TM3DI32K	32	Regular inputs	24 Vdc 5 mA	HE10 (MIL 20) connector

TM3 Digital Output Modules

The following table shows the TM3 digital output expansion modules, with corresponding channel type, nominal voltage/current, and terminal type:

Reference	Channels	Channel Type	Voltage Current	Terminal Type / Pitch
TM3DQ8R	8	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	Removable screw terminal block / 5.08 mm
TM3DQ8RG	8	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	Removable spring terminal block / 5.08 mm
TM3DQ8T	8	Regular transistor outputs (source)	24 Vdc 4 A maximum per common line / 0.5 A maximum per output	Removable screw terminal block / 5.08 mm
TM3DQ8TG	8	Regular transistor outputs (source)	24 Vdc 4 A maximum per common line / 0.5 A maximum per output	Removable spring terminal block / 5.08 mm
TM3DQ8U	8	Regular transistor outputs (sink)	24 Vdc 4 A maximum per common line / 0.5 A maximum per output	Removable screw terminal block / 5.08 mm
TM3DQ8UG	8	Regular transistor outputs (sink)	24 Vdc 4 A maximum per common line / 0.5 A maximum per output	Removable spring terminal block / 5.08 mm
TM3DQ16R	16	Relay outputs	24 Vdc / 240 Vac 8 A maximum per common line / 2 A maximum per output	Removable screw terminal blocks / 3.81 mm
TM3DQ16RG	16	Relay outputs	24 Vdc / 240 Vac 8 A maximum per common line / 2 A maximum per output	Removable spring terminal blocks / 3.81 mm
TM3DQ16T	16	Regular transistor outputs (source)	24 Vdc 8 A maximum per common line / 0.5 A maximum per output	Removable screw terminal blocks / 3.81 mm
TM3DQ16TG	16	Regular transistor outputs (source)	24 Vdc 8 A maximum per common line / 0.5 A maximum per output	Removable spring terminal blocks / 3.81 mm
TM3DQ16U	16	Regular transistor outputs (sink)	24 Vdc 8 A maximum per common line / 0.5 A maximum per output	Removable screw terminal blocks / 3.81 mm
TM3DQ16UG	16	Regular transistor outputs (sink)	24 Vdc 8 A maximum per common line / 0.5 A maximum per output	Removable spring terminal blocks / 3.81 mm
TM3DQ16TK	16	Regular transistor outputs (source)	24 Vdc 2 A maximum per common line / 0.1 A maximum per output	HE10 (MIL 20) connector
TM3DQ16UK	16	Regular transistor outputs (sink)	24 Vdc 2 A maximum per common line / 0.1 A maximum per output	HE10 (MIL 20) connector

Reference	Channels	Channel Type	Voltage Current	Terminal Type / Pitch
TM3DQ32TK	32	Regular transistor outputs (source)	24 Vdc 2 A maximum per common line / 0.1 A maximum per output	HE10 (MIL 20) connectors
TM3DQ32UK	32	Regular transistor outputs (sink)	24 Vdc 2 A maximum per common line / 0.1 A maximum per output	HE10 (MIL 20) connectors

TM3 Digital Mixed Input/Output Modules

This following table shows the TM3 mixed I/O modules, with corresponding channel type, nominal voltage/current, and terminal type:

Reference	Channels	Channel Type	Voltage Current	Terminal Type / Pitch
TM3DM8R	4	Regular inputs	24 Vdc 7 mA	Removable screw terminal block / 5.08 mm
	4	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	
TM3DM8RG	4	Regular inputs	24 Vdc 7 mA	Removable spring terminal block / 5.08 mm
	4	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	
TM3DM16R ⁽¹⁾	8	Regular inputs	24 Vdc 5 mA	Removable screw terminal block / 3.81 mm
	8	Relay outputs	24 Vdc / 240 Vac 4 A maximum per common line / 2 A maximum per output	
TM3DM24R	16	Regular inputs	24 Vdc 7 mA	Removable screw terminal block / 3.81 mm
	8	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	
TM3DM24RG	16	Regular inputs	24 Vdc 7 mA	Removable spring terminal block / 3.81 mm
	8	Relay outputs	24 Vdc / 240 Vac 7 A maximum per common line / 2 A maximum per output	
TM3DM32R ⁽¹⁾	16	Regular inputs	24 Vdc 5 mA	Removable screw terminal block / 3.81 mm
	16	Relay outputs	24 Vdc / 240 Vac 4 A maximum per common line / 2 A maximum per output	
(1) This expansion module is available only in selected countries.				

TM3 Analog Input Modules

The following table shows the TM3 analog input expansion modules, with corresponding resolution, channel type, nominal voltage/current, and terminal type:

Reference	Resolution	Channels	Channel Type	Mode	Terminal Type / Pitch
TM3AI2H	16 bit, or 15 bit + sign	2	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable screw terminal block / 5.08 mm
TM3AI2HG	16 bit, or 15 bit + sign	2	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable spring terminal block / 5.08 mm
TM3AI4	12 bit, or 11 bit + sign	4	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable screw terminal block / 3.81 mm
TM3AI4G	12 bit, or 11 bit + sign	4	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable spring terminal blocks / 3.81 mm
TM3AI8	12 bit, or 11 bit + sign	8	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA 0...20 mA extended 4...20 mA extended	Removable screw terminal block / 3.81 mm
TM3AI8G	12 bit, or 11 bit + sign	8	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA 0...20 mA extended 4...20 mA extended	Removable spring terminal blocks / 3.81 mm
TM3TI4	16 bit, or 15 bit + sign	4	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA Thermocouple PT100/1000 NI100/1000	Removable screw terminal block / 3.81 mm

Reference	Resolution	Channels	Channel Type	Mode	Terminal Type / Pitch
TM3TI4G	16 bit, or 15 bit + sign	4	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA Thermocouple PT100/1000 NI100/1000	Removable spring terminal blocks / 3.81 mm
TM3TI4D	16 bit, or 15 bit + sign	4	inputs	Thermocouple	Removable screw terminal block / 3.81 mm
TM3TI4DG	16 bit, or 15 bit + sign	4	inputs	Thermocouple	Removable spring terminal blocks / 3.81 mm
TM3TI8T	16 bit, or 15 bit + sign	8	inputs	Thermocouple NTC/PTC Ohmmeter	Removable screw terminal block / 3.81 mm
TM3TI8TG	16 bit, or 15 bit + sign	8	inputs	Thermocouple NTC/PTC Ohmmeter	Removable spring terminal blocks / 3.81 mm

TM3 Analog Output Modules

The following table shows the TM3 analog output modules, with corresponding resolution, channel type, nominal voltage/current, and terminal type:

Reference	Resolution	Channels	Channel Type	Mode	Terminal Type / Pitch
TM3AQ2	12 bit, or 11 bit + sign	2	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable screw terminal block / 5.08 mm
TM3AQ2G	12 bit, or 11 bit + sign	2	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable spring terminal block / 5.08 mm
TM3AQ4	12 bit, or 11 bit + sign	4	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable screw terminal block / 5.08 mm
TM3AQ4G	12 bit, or 11 bit + sign	4	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	Removable spring terminal block / 5.08 mm

TM3 Analog Mixed Input/Output Modules

This following table shows the TM3 analog mixed I/O modules, with corresponding resolution, channel type, nominal voltage/current, and terminal type:

Reference	Resolution	Channels	Channel Type	Mode	Terminal Type / Pitch
TM3AM6	12 bit, or 11 bit + sign	4	inputs	0...10 Vdc	Removable screw terminal block / 3.81 mm
		2	outputs	-10...+10 Vdc 0...20 mA 4...20 mA	
TM3AM6G	12 bit, or 11 bit + sign	4	inputs	0...10 Vdc	Removable spring terminal block / 3.81 mm
		2	outputs	-10...+10 Vdc 0...20 mA 4...20 mA	
TM3TM3	16 bit, or 15 bit + sign	2	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA Thermocouple PT100/1000 NI100/1000	Removable screw terminal block / 5.08 mm
	12 bit, or 11 bit + sign	1	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	
TM3TM3G	16 bit, or 15 bit + sign	2	inputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA Thermocouple PT100/1000 NI100/1000	Removable spring terminal block / 5.08 mm
	12 bit, or 11 bit + sign	1	outputs	0...10 Vdc -10...+10 Vdc 0...20 mA 4...20 mA	

TM3 Expert Modules

The following table shows the TM3 expert expansion modules, with corresponding terminal types:

Reference	Description	Terminal Type / Pitch
TM3XTYS4	TeSys module	4 front connectors RJ-45 1 removable power supply connector / 5.08 mm
TM3XFHSC202	High Speed Counting (HSC) module with events	Removable screw terminal blocks / 3.81 mm
TM3XFHSC202G	High Speed Counting (HSC) module with events	Removable spring terminal blocks / 3.81 mm
TM3XHSC202	High Speed Counting (HSC) module	Removable screw terminal blocks / 3.81 mm
TM3XHSC202G	High Speed Counting (HSC) module	Removable spring terminal blocks / 3.81 mm

TM3 Safety Modules

This table contains the TM3 safety modules, with the corresponding channel type, nominal voltage/current, and terminal type:

Reference	Function Category	Channels	Channel type	Voltage Current	Terminal type
TM3SAC5R	1 function, up to category 3	1 or 2 ⁽¹⁾	Safety input	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable screw terminal block
		Start ⁽²⁾	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAC5RG	1 function, up to category 3	1 or 2 ⁽¹⁾	Safety input	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable spring terminal block
		Start ⁽²⁾	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAF5R	1 function, up to category 4	2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable screw terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAF5RG	1 function, up to category 4	2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable spring terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAFL5R	2 functions, up to category 3	2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable screw terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAFL5RG	2 functions, up to category 3	2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable spring terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAK6R	3 functions, up to category 4	1 or 2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable screw terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
TM3SAK6RG	3 functions, up to category 4	1 or 2 ⁽¹⁾	Safety inputs	24 Vdc	3.81 mm (0.15 in.) and 5.08 mm (0.20 in.), removable spring terminal block
		Start	Input	100 mA maximum	
		3 in parallel	Relay outputs Normally open	24 Vdc / 230 Vac 6 A maximum per output	
⁽¹⁾ Depending on external wiring					
⁽²⁾ Non-monitored start					

TM3 Transmitter and Receiver Modules

The following table shows the TM3 transmitter and receiver expansion modules:

Reference	Description	Terminal Type / Pitch
TM3XTRA1	Data transmitter module for remote I/O	1 front connector RJ-45 1 screw for functional ground connection
TM3XREC1	Data receiver module for remote I/O	1 front connector RJ-45 Power supply connector / 5.08 mm

TM3 Bus Couplers

Introduction

The TM3 bus coupler is a device designed to manage fieldbus communication when using TM2 and TM3 expansion modules in a distributed architecture.

For more information, refer to the Modicon TM3 Bus Coupler Hardware Guide.

Modicon TM3 Bus Couplers

The following table shows the TM3 bus couplers, with ports and terminal types:

Reference	Port	Communication type	Terminal type
TM3BCEIP	2 isolated switched Ethernet ports	EtherNet/IP Modbus TCP	RJ45
	1 USB port	USB 2.0	USB mini-B
TM3BCSL	2 isolated RS-485 ports (daisy-chained)	Serial Line Modbus	RJ45
	1 USB port	USB 2.0	USB mini-B
TM3BCCO	2 isolated CANopen ports (daisy-chained)	CANopen	RJ45
	1 USB port	USB 2.0	USB mini-B

TM5 Fieldbus Interfaces

Introduction

The TM5 fieldbus interfaces are devices designed to manage EtherNet/IP and Sercos communication when using TM5 System and TM7 expansion modules with a controller in a distributed architecture.

For more information, refer to the Modicon TM5 System Interface – Hardware Guide.

TM5 Fieldbus Interfaces

The following table shows the TM5 fieldbus interfaces with ports and terminal type:

Reference	Port	Communication type	Terminal type
TM5NEIP1	2 Ethernet switched ports	EtherNet/IP	RJ45
TM5NS31	2 Ethernet switched ports	Sercos	RJ45

TM5 CANopen Fieldbus Interfaces

Introduction

The TM5 fieldbus module is a CANopen interface with built-in power distribution and is the first TM5 distributed I/O island.

For more information, refer to the Modicon TM5 CANopen Interface Hardware Guide.

Modicon TM5 CANopen Fieldbus Interfaces

The following table shows the TM5 CANopen fieldbus interfaces:

Reference	Communication type	Terminal type
TM5NCO1	CANopen	1 SUB-D 9, male

TM7 CANopen Fieldbus Interfaces

Introduction

The TM7 fieldbus modules are CANopen interfaces with 24 Vdc digital configurable input or output on 8 or 16 channels.

For more information, refer to the Modicon TM7 CANopen Interface I/O Blocks Hardware Guide.

Modicon TM7 CANopen Fieldbus Interfaces

The following table shows the TM7 CANopen fieldbus interfaces:

Reference	Number of channels	Voltage/Current	Communication type	Terminal type
TM7NCOM08B	8 inputs	24 Vdc / 4 mA	CANopen	M8 Connector
	8 outputs	24 Vdc / 500 mA		
TM7NCOM16A	16 inputs	24 Vdc / 4 mA	CANopen	M8 Connector
	16 outputs	24 Vdc / 500 mA		
TM7NCOM16B	16 inputs	24 Vdc / 4 mA	CANopen	M12 Connector
	16 outputs	24 Vdc / 500 mA		

TMS Expansion Modules

Introduction

TMS expansion modules attach to the left side of the controller to provide additional communication possibilities. The modules are dedicated to Ethernet and CANopen high-speed communication.

For more information, refer to the TMS Expansion Modules Hardware Guide.

TMS Expansion Modules

The following table describes the TMS expansion module features:

Module reference	Type	Terminal type	Compatibility
TMSES4	Ethernet communication	RJ45	TM262L10MESE8T TM262L20MESE8T TM262M15MESS8T TM262M25MESS8T TM262M35MESS8T
TMSCO1	CANopen master module	SUB-D 9 pin male	TM262L• TM262M•

Accessories

Overview

This section describes the accessories and cables.

Accessories

Reference	Description	Use	Quantity
TMASD1	SD Card	Use to update the controller firmware, initialize a controller with a new application or clone a controller, apply post configuration file to the controller, store recipe files, and receive data logging files.	1
TMA262SET8G	Removable 11-pt spring terminal block (pitch 3.81 mm): 3 terminals for 24 Vdc I/O 4 terminals for inputs 4 terminals for outputs	Connects 24 Vdc power supply, and embedded I/Os.	1
	Removable 5-pt spring terminal block (pitch 5.08 mm): 3 terminals for 24 Vdc I/O 2 terminals for relay output	Connects 24 Vdc power supply and relay output.	1
TMA262SET8S	Removable 11-pt screw terminal block (pitch 3.81 mm): 3 terminals for 24 Vdc I/O 4 terminals for inputs 4 terminals for outputs	Connects 24 Vdc power supply and embedded I/Os.	1
	Removable 5-pt screw terminal block (pitch 5.08 mm): 3 terminals for 24 Vdc I/O 2 terminals for relay output	Connects 24 Vdc power supply and relay output.	1

Reference	Description	Use	Quantity
NSYTRAAB35	End brackets	Helps secure the controller or receiver module and their expansion modules on a top hat section rail (DIN rail).	1
TM2XMTGB	Grounding Bar	Connects the cable shield and the module to the functional ground.	1
TM200RSRCMC	Shielding take-up clip	Mounts and connects the ground to the cable shielding.	25 pack
TMAM3	2 attachment parts	Mounts the controller and TMS modules directly to a flat, vertical panel.	1

Cables

Reference	Description	Details	Length
TCSXCNAMUM3P	Terminal port/USB port cordset	From the USB mini-B port on the M262 Logic/Motion Controller to USB port on the PC terminal.	3 m (10 ft)
BMXXCAUSBH018	Terminal port/USB port cordset	From the USB mini-B port on the M262 Logic/Motion Controller to USB port on the PC terminal. NOTE: Grounded and shielded, this USB cable is suitable for long-duration connections.	1.8 m (5.9 ft)
TCSMCN3M4F3C2	RS-232 serial link cordset 1 RJ45 connector and 1 SUB-D 9 connector	For DTE terminal (printer).	3 m (9.84 ft)
490NTW000**	Ethernet shielded cable for DTE connections	Standard cable, equipped with RJ45 connectors at each end for DTE. CE compliant.	2, 5, 12, 40, or 80 m (6.56, 16.4, 39.37, 131.23 or 262.47 ft)
490NTW000**U		Standard cable, equipped with RJ45 connectors at each end for DTE. UL compliant.	2, 5, 12, 40, or 80 m (6.56, 16.4, 39.37, 131.23, or 262.47 ft)
TCSECE3M3M**S4		Cable for harsh environment, equipped with RJ45 connectors at each end. CE compliant.	1, 2, 3, 5, or 10 m (3.28, 6.56, 9.84, 16.4, 32.81 ft)
TCSECU3M3M**S4		Cable for harsh environment, equipped with RJ45 connectors at each end. UL compliant.	1, 2, 3, 5, or 10 m (3.28, 6.56, 9.84, 16.4, 32.81 ft)
VW3E5001R***	Sercos cable	Cable with Tj45 connectors at each end.	0.5, 1, 1.5, 2, 3, 5, 10, 15, 20, 25, 30, 40, or 50 m (1.64, 3.28, 4.92, 6.56, 9.84, 16.4, 32.8, 49.2, 65.6, 82, 98.4, 131.2, or 164 ft)
VW3A8306R**	2 RJ45 connectors	Cable equipped with RJ45 connectors at each end for Modbus serial link.	0.3, 1, or 3 m (0.98, 3.28, or 9.84 ft)

M262 Features

Real Time Clock (RTC)

Overview

The M262 Logic/Motion Controller includes a real-time clock (RTC) to provide system date and time information and to support related functions requiring a real-time clock.

The RTC also provides the system date and time to any TMS expansion modules (see Modicon TMS, Expansion Module, Hardware Guide) installed on the left side of the controller.

Provided the controller has been powered on for at least 2 hours, the system date and time are maintained for 1000 hours at 25 °C (77 °F) even when the controller is powered off.

This table shows how RTC drift is managed:

RTC Characteristics	Description
RTC drift	Less than 15 seconds per month with no user calibration at 25 °C (77 °F)

To set and calibrate the RTC in EcoStruxure Machine Expert, use either:

- The **Services** tab (see M262 Logic/Motion Controller - Programming Guide).
- The `SysTimeRtcSet` function block (see EcoStruxure Machine Expert, Getting & Setting Real Time Clock, SysTimeRtc and SysTimeCore Library Guide).

Input Management

Overview

The M262 Logic/Motion Controller features 4 fast digital inputs.

The following functions are configurable:

- Filters (depends on the function associated with the input).
- All inputs can be used for the Run/Stop function.
- The inputs can be either latched or used for events (rising edge, falling edge, or both) and thus be linked to an external task.

NOTE: All inputs can be used as regular inputs.

Input Management Functions Availability

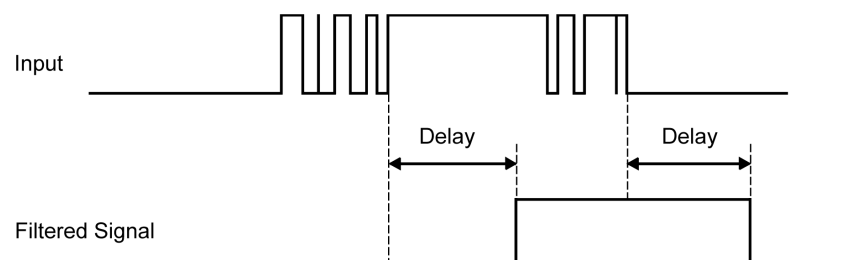
Embedded digital inputs can be configured as functions (Run/Stop, events).

Inputs not configured as functions are used as regular inputs.

Filter Principle

The filter is designed to reduce the bouncing effect at the inputs. Setting the filter value allows the controller to ignore some sudden changes of input levels caused by electrical noise. The filter is only available on the fast inputs.

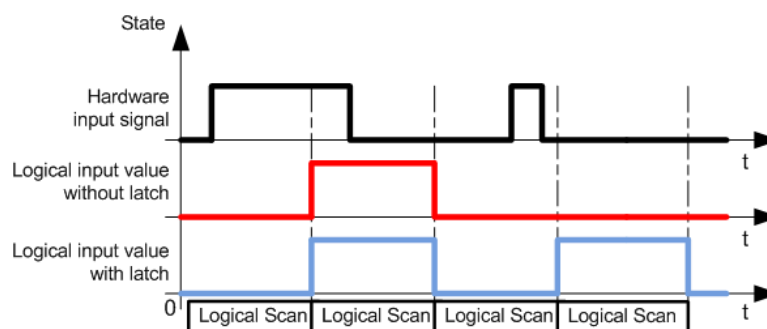
The following timing diagram illustrates the anti-bounce filter effects:



Latching

Latching is a function that can be assigned to the M262 Logic/Motion Controller fast inputs. This function is used to memorize (or latch) any pulse with a duration that is less than the M262 Logic/Motion Controller scan time. When a pulse is shorter than one scan, the controller latches the pulse, which is then updated in the next scan. This latching mechanism only recognizes rising edges. Falling edges cannot be latched. Assigning inputs to be latched is done in the **I/O Configuration** tab in EcoStruxure Machine Expert.

The following timing diagram illustrates the latching effects:



Event

An input configured for Event can be associated with an External Task (see Modicon M262 Logic/Motion Controller, Programming Guide).

Run/Stop

The Run/Stop function is used to start or stop an application program using an input. In addition to the embedded Run/Stop switch, it is allowed to configure one (and only one) input as an additional Run/Stop command.

For more information, refer to [Run/Stop](#), page 32.

⚠ WARNING

UNINTENDED MACHINE OR PROCESS START-UP

- Verify the state of security of your machine or process environment before applying power to the Run/Stop input.
- Use the Run/Stop input to help prevent the unintentional start-up from a remote location.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Use the sensor and actuator power supply only for supplying power to sensors or actuators connected to the module.

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

Output Management

Introduction

The M262 Logic/Motion Controller features fast digital outputs.

Alarm output functions can be configured on the outputs.

NOTE: All outputs can be used as regular outputs.

Fallback Modes (Behavior for Outputs in Stop)

When the controller enters the STOPPED or one of the exception states for any reason, the local (embedded and expansion) outputs are set to **Default Value** defined in the application.

Short-circuit or Over-current on Outputs

In the case of a short-circuit or current overload, all outputs enter into thermal or over-current protection mode (all outputs are set to 0), and are then periodically rearmed (every 10 seconds) to test the connection state. However, you must be aware of the effect of this rearming on the machine or process being controlled.

⚠ WARNING

UNINTENDED MACHINE START-UP

Inhibit the automatic rearming of outputs if this feature is an undesirable behavior for your machine or process.

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

NOTE: The automatic rearming feature can be inhibited. Refer to the Programming Guide of your controller for more information.

Run/Stop

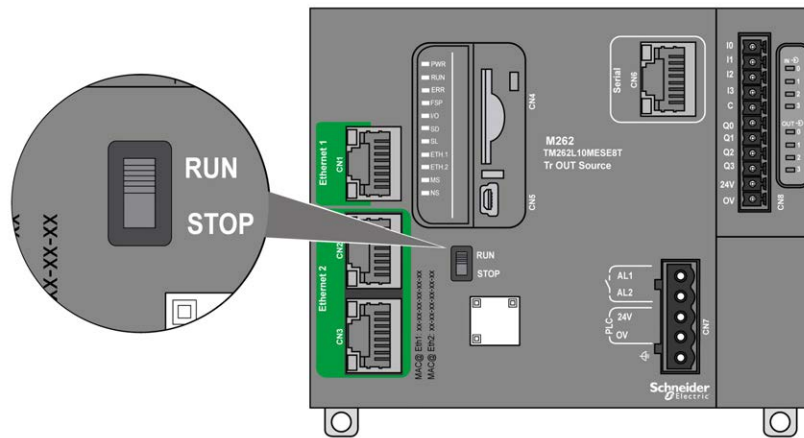
Run/Stop

The M262 Logic/Motion Controller can be operated externally by the following:

- A hardware Run/Stop switch.
- An EcoStruxure Machine Expert software command.
- A Run/Stop operation triggered by an embedded digital input. The digital input is defined in the software configuration. For more information, refer to the M262 Logic/Motion Controller Programming Guide.

- The system variable PLC_W in a Relocation Table (see Modicon M262 Logic/Motion Controller, Programming Guide).
- The Web server (see Modicon M262 Logic/Motion Controller, Programming Guide).

The M262 Logic/Motion Controller has a hardware Run/Stop switch, which puts the controller in the RUN or STOP state.



The interaction of the 2 external operators on the controller state behavior is summarized in the table below:

		Embedded hardware Run/Stop switch		
		Switch on Stop	Stop to Run transition	Switch on Run
Software configurable Run/Stop digital input	None	STOP Ignores external Run/Stop commands.	Commands a transition to RUN state ¹ .	Allows external Run/Stop commands.
	State 0	STOP Ignores external Run/Stop commands.	STOP Ignores external Run/Stop commands.	STOP Ignores external Run/Stop commands.
	Rising edge	STOP Ignores external Run/Stop commands.	Commands a transition to RUN state ¹ .	Commands a transition to RUN state.
	State 1	STOP Ignores external Run/Stop commands.	Commands a transition to RUN state ¹ .	Allows external Run/Stop commands.

¹ For more information, refer to the M262 Logic/Motion Controller Programming Guide.

⚠ WARNING

UNINTENDED MACHINE OR PROCESS START-UP

- Verify the state of security of your machine or process environment before applying power to the Run/Stop input or engaging the Run/Stop switch.
- Use the Run/Stop input to help prevent the unintentional start-up from a remote location, or from accidentally engaging the Run/Stop switch.

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

SD Card

Overview

The main uses of the SD card are:

- Downloading a new application to the controller without using EcoStruxure Machine Expert software.
- Updating the controller firmware
- Cloning the controller application or firmware
- Applying post configuration changes to the controller (for example, changing IP addresses or serial line configuration)
- Applying recipe files
- Retrieving data logging files

The SD card file system is FAT32. SD card files can therefore be used directly on your computer.

When handling the SD card, follow the instructions below to help prevent internal data on the SD card from being corrupted or lost or an SD card malfunction from occurring:

NOTICE

LOSS OF APPLICATION DATA

- Do not store the SD card where there is static electricity or probable electromagnetic fields.
- Do not store the SD card in direct sunlight, near a heater, or other locations where high temperatures can occur.
- Do not bend the SD card.
- Do not drop or strike the SD card against another object.
- Keep the SD card dry.
- Do not touch the SD card connectors.
- Do not disassemble or modify the SD card.
- Use only SD cards formatted using FAT or FAT32.

Failure to follow these instructions can result in equipment damage.

The M262 Logic/Motion Controller does not recognize NTFS formatted SD cards. Format the SD card on your computer using FAT or FAT32.

When using the M262 Logic/Motion Controller and an SD card, observe the following to avoid losing valuable data:

- Accidental data loss can occur at any time. Once data is lost it cannot be recovered.
- If you forcibly extract the SD card, data on the SD card may become corrupted.
- Removing an SD card that is being accessed (**SD** LED flashing yellow) could damage the SD card, or corrupt its data.
- If the SD card is not positioned correctly when inserted into the controller, the data on the card and the controller could become damaged.

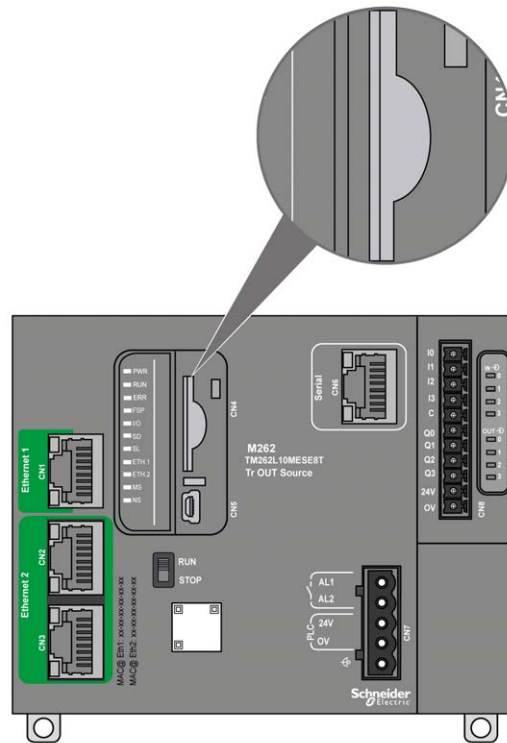
NOTICE

LOSS OF APPLICATION DATA

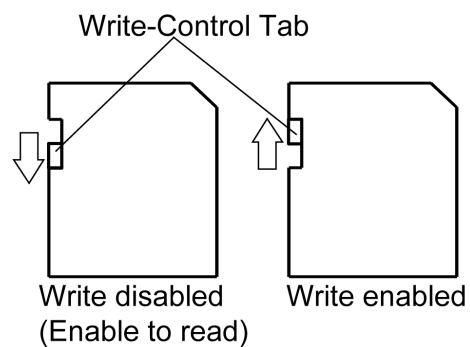
- Backup SD card data regularly.
- Do not remove power or reset the controller, and do not insert or remove the SD card while it is being accessed.

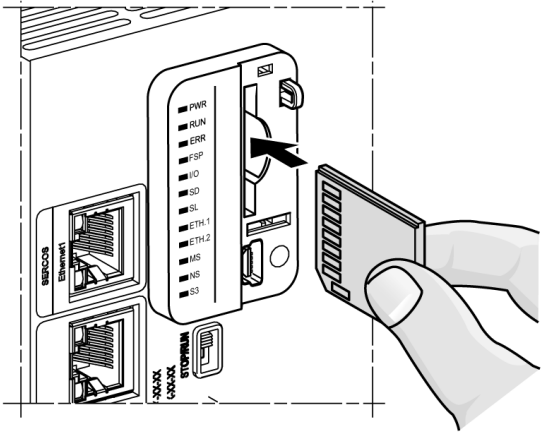
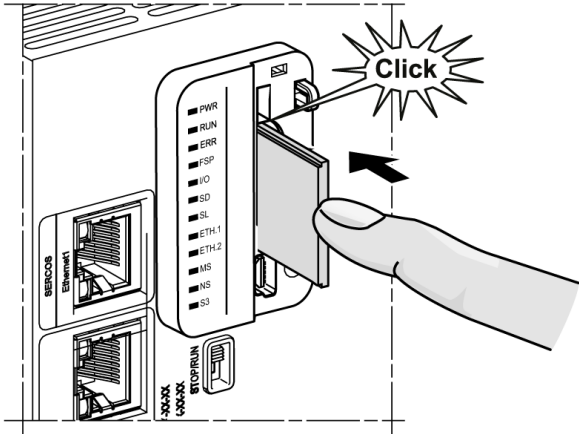
Failure to follow these instructions can result in equipment damage.

The following figure shows the SD card slot:



It is possible to set the Write-Control Tab to prevent write operations to the SD card. Push the tab up, as shown in the example on the right-hand side, to release the lock and enable writing to the SD card. Before using an SD card, read the manufacturer's instructions.



Step	Action
1	Insert the SD card into the SD card slot: 
2	Push until you hear it "click": 

SD Card Slot Characteristics

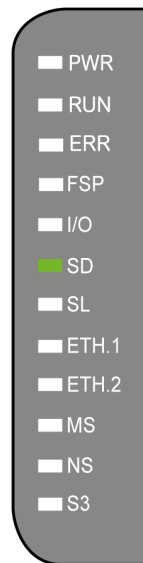
Topic	Characteristics	Description
Supported type	Standard Capacity	SD (SDSC)
	High Capacity	SDHC
Global memory	Size	32 GB max. (SDHC only)

TMA5D1 Characteristics

Characteristics	Description
Card removal durability	Minimum 1000 times
File retention time	10 years at 25 °C (77 °F)
Flash type	SLC NAND
Memory size	256 MB
Ambient operation temperature	–10 ... +85°C (14...185 °F)
Storage temperature	–25 ... +85°C (–13...185 °F)
Relative humidity	95% max. non-condensing
Write/Erase cycles	3,000,000 (approximately)

Status LED

The following figure shows the **SD** status LED:



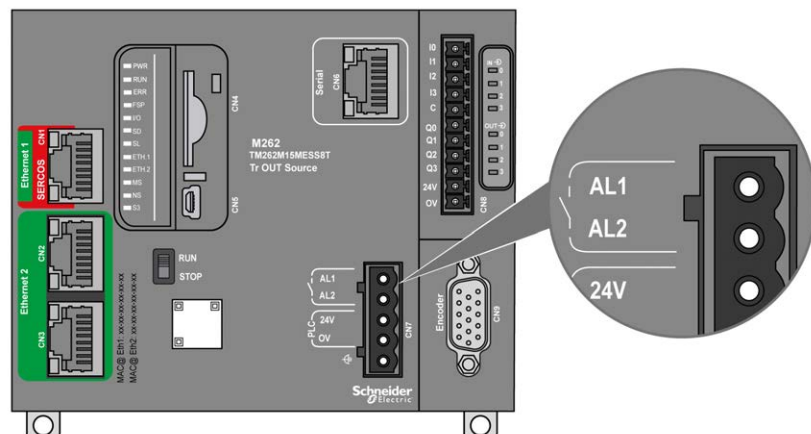
The following table describes the **SD** status LED:

Label	Description	LED	
		State	Description
SD	SD card	Green On	Firmware update completed.
		Green Flashing	Firmware update or script execution in progress.
		Yellow On	Firmware update or script execution is unsuccessful.
		Yellow Flashing	SD card is being accessed (script execution in progress).
		Off	No SD card activity.

Alarm Relay

Introduction

The M262 Logic/Motion Controller has integrated relay connections that can be wired to an external alarm:



For wiring details, refer to Alarm Relay Wiring, page 64.

Characteristics

This table shows the characteristics of the alarm relay:

Characteristic	Value
Wiring type	2 terminals on removable spring terminal block
Output type	Relay
Contact type	Normally Open (NO)
Nominal input voltage	24 Vdc
Maximum input voltage	28.8 Vdc
Input voltage type	PELV
Contact resistance	300 mΩ max.
Minimum switching load	5 V at 100 mA
Maximum current	700 mA
Overload protection	Yes, resettable fuse, max. 3.2 A
Reverse polarity protection	Not necessary

Operation

When the controller is operating normally in the RUN state, the alarm relay is activated and its contact is closed.

The relay contact is opened by one of the following conditions:

- Appearance of a blocking error.
- Disappearance of the mains voltage.

Perform a power cycle of the controller to recover from a hardware watchdog event and reset the relay output contact to the closed state.

When the controller is powered off, the alarm relay is deactivated and its contact is opened.

M262 Installation

Overview

This chapter provides installation safety guidelines, device dimensions, mounting instructions, and environmental specifications.

M262 Logic/Motion Controller General Rules for Implementing

Environmental Characteristics

Enclosure Requirements

M262 Logic/Motion Controller system components are designed as Zone B, Class A industrial equipment according to IEC/CISPR Publication 11. If they are used in environments other than those described in the standard, or in environments that do not meet the specifications in this manual, the ability to meet electromagnetic compatibility requirements in the presence of conducted and/or radiated interference may be reduced.

All M262 Logic/Motion Controller system components meet European Community (CE) requirements for open equipment as defined by IEC/EN 61131-2. You must install them in an enclosure designed for the specific environmental conditions and to minimize the possibility of unintended contact with hazardous voltages. Use metal enclosures to improve the electromagnetic immunity of your M262 Logic/Motion Controller system. Use enclosures with a keyed locking mechanism to minimize unauthorized access.

Environmental Characteristics

All the M262 Logic/Motion Controller module components are electrically isolated between the internal electronic circuit and the input/output channels within the limits set forth and described by these environmental characteristics. For more information on electrical isolation, see the technical specifications of your particular controller found later in the current document. This equipment meets CE requirements as indicated in the table below. This equipment is intended for use in a Pollution Degree 2 industrial environment.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not exceed any of the rated values specified in the environmental and electrical characteristics tables.

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

The following table shows the general environmental characteristics:

Characteristic	Minimum Specification	Tested Range	
Standard compliance	IEC/EN 61131-2 UL/CSA 61010-1, -2-201	–	
Ambient operating temperature	–	Horizontal installation	–20...60 °C (–4...140 °F)
	–	Vertical installation	–20...50 °C (–4...122 °F)
	–	Flat installation	–20...45 °C (–4...113 °F)
Transport/storage temperature	–	–40...85 °C (–40...185 °F)	
Relative humidity	–	Transport and storage	5...95 % (non-condensing)
	–	Operation	5...95 % (non-condensing)
Degree of pollution	IEC/EN 60664-1	2	
Degree of protection	IEC/EN 61131-2	IP20 with protective covers in place	
Corrosion immunity	–	Atmosphere free from corrosive gases	
Operating altitude	–	0...2000 m (0...6560 ft)	
Storage altitude	–	0...3000 m (0...9843 ft)	
Vibration resistance	IEC/EN 61131-2	Panel mounting or mounted on a top hat section rail (DIN rail)	3.5 mm (0.13 in) fixed amplitude from 2...8.4 Hz 9.8 m/s ² (32.15 ft/s ²) (1 g _n) fixed acceleration from 8.4...200 Hz
Mechanical shock resistance	–	147 m/s ² (482.28 ft/s ²) (15 g _n) for a duration of 11 ms	
NOTE: The tested ranges may indicate values beyond that of the IEC Standard. However, our internal standards define what is necessary for industrial environments. In all cases, we uphold the minimum specification if indicated.			

Electromagnetic Susceptibility

The M262 Logic/Motion Controller system meets electromagnetic susceptibility specifications as indicated in the following table:

Characteristic	Minimum Specification	Tested Range		
Electrostatic discharge	IEC/EN 61000-4-2	8 kV (air discharge)		
	IEC/EN 61131-2	4 kV (contact discharge)		
Radiated electromagnetic field	IEC/EN 61000-4-3	10 V/m (80...1000 MHz)		
	IEC/EN 61131-2	3 V/m (1.4...2 GHz) 1 V/m (2...3 GHz)		
Fast transients burst	IEC/EN 61000-4-4 IEC/EN 61131-2	24 Vdc main power lines	2 kV (CM ¹ and DM ²)	
		24 Vdc I/Os	2 kV (clamp)	
		Relay output	1 kV (clamp)	
		Digital I/Os	1 kV (clamp)	
		Communication line	1 kV (clamp)	
Surge immunity	IEC/EN 61000-4-5 IEC/EN 61131-2	–	CM ¹	DM ²
		DC Power lines	0.5 kV	0.5 kV
		Relay Outputs	–	–
		24 Vdc I/Os	–	–
		Shielded cable (between shield and ground)	1 kV	–
Induced electromagnetic field	IEC/EN 61000-4-6 IEC/EN 61131-2	10 Vrms (0.15...80 MHz)		
Conducted emission	IEC 61000-6-4 IEC/EN 61131-2	10...150 kHz: 120...69 dBµV/m QP 150...1500 kHz: 79...63 dBµV/m QP 1.5...30 MHz: 63 dBµV/m QP		
Radiated emission	IEC 61000-6-4	30...230 MHz: 40 dBµV/m QP		
	IEC/EN 61131-2	230...1000 MHz: 47 dBµV/m QP		
1 Common Mode				
2 Differential Mode				
NOTE: The tested ranges may indicate values beyond that of the IEC Standard. However, our internal standards define what is necessary for industrial environments. In all cases, we uphold the minimum specification if indicated.				

Certifications and Standards

Introduction

For information on certifications and conformance to standards, go to www.se.com.

For product compliance and environmental information (RoHS, REACH, PEP, EOL, etc.), go to www.se.com/green-premium.

M262 Logic/Motion Controller Installation

Installation and Maintenance Requirements

Before Starting

Read and understand this chapter before beginning the installation of your system.

The use and application of the information contained herein require expertise in the design and programming of automated control systems. Only you, the user, machine builder or integrator, can be aware of all the conditions and factors present during installation and setup, operation, and maintenance of the machine or process, and can therefore determine the automation and associated equipment and the related safeties and interlocks which can be effectively and properly used. When selecting automation and control equipment, and any other related equipment or software, for a particular application, you must also consider any applicable local, regional or national standards and/or regulations.

Pay particular attention in conforming to any safety information, different electrical requirements, and normative standards that would apply to your machine or process in the use of this equipment.

Disconnecting Power

All options and modules should be assembled and installed before installing the control system on a mounting rail, onto a mounting plate or in a panel. Remove the control system from its mounting rail, mounting plate or panel before disassembling the equipment.

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

Programming Considerations

WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Operating Environment

In addition to the **Environmental Characteristics**, refer to **Product Related Information** in the beginning of the present document for important information regarding installation in hazardous locations for this specific equipment.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Install and operate this equipment according to the conditions described in the Environmental Characteristics.

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

Installation Considerations

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Use appropriate safety interlocks where personnel and/or equipment hazards exist.
- Install and operate this equipment in an enclosure appropriately rated for its intended environment and secured by a keyed or tooled locking mechanism.
- Use the sensor and actuator power supplies only for supplying power to the sensors or actuators connected to the module.
- Power line and output circuits must be wired and fused in compliance with local and national regulatory requirements for the rated current and voltage of the particular equipment.
- Do not use this equipment in safety-critical machine functions unless the equipment is otherwise designated as functional safety equipment and conforming to applicable regulations and standards.
- Do not disassemble, repair, or modify this equipment.
- Do not connect any wiring to reserved, unused connections, or to connections designated as No Connection (N.C.).

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

NOTE: JDYX2 or JDYX8 fuse types are UL-recognized and CSA approved.

M262 Logic/Motion Controller Mounting Positions and Clearances

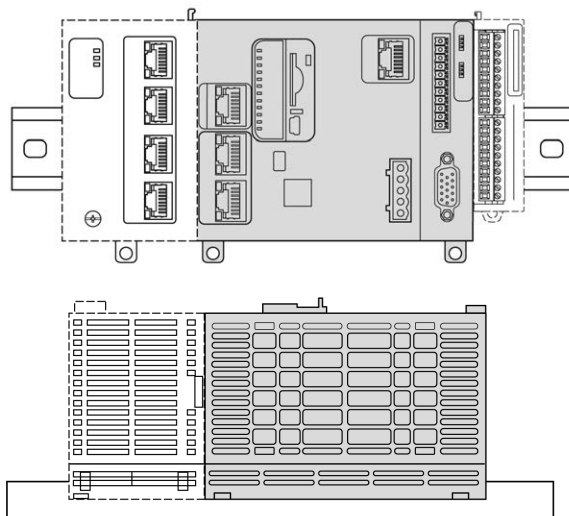
Introduction

This section describes the correct mounting positions for the M262 Logic/Motion Controller.

NOTE: Keep adequate spacing for proper ventilation and to maintain the operating temperature specified in the Environmental Characteristics, page 39.

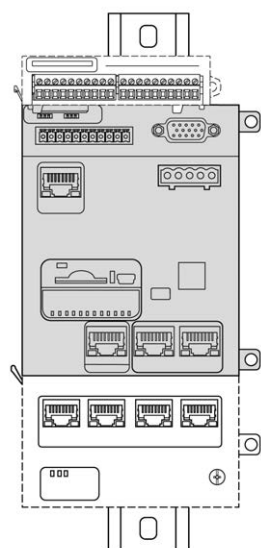
Correct Mounting Position

To obtain optimal operating characteristics, the M262 Logic/Motion Controller should be mounted as shown in the figures below:



Acceptable Mounting Positions

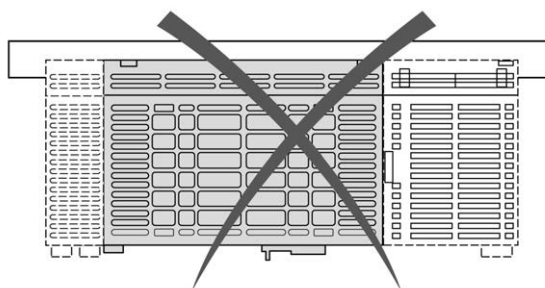
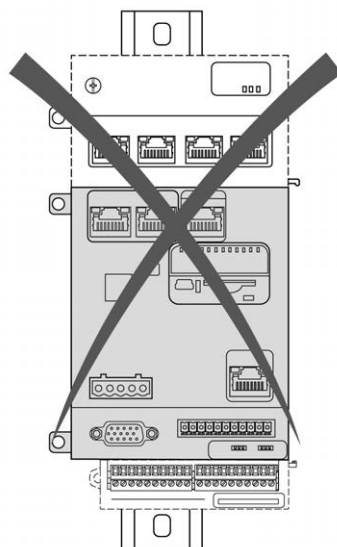
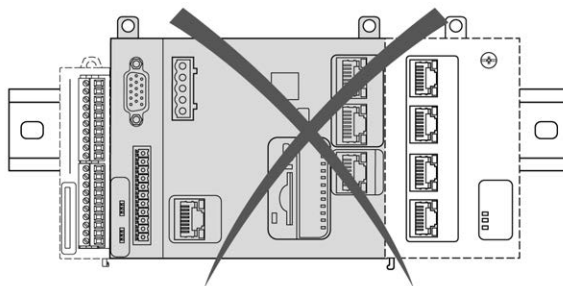
The M262 Logic/Motion Controller can also be mounted vertically on a vertical plane as shown below:



NOTE: TM3 expansion modules must be mounted above the controller.

Incorrect Mounting Positions

The M262 Logic/Motion Controller should only be positioned as shown in the Correct Mounting Position, page 44 figures. The figures below show incorrect mounting positions:



Minimum Clearances

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Place devices dissipating the most heat at the top of the cabinet and ensure adequate ventilation.
- Avoid placing this equipment next to or above devices that might cause overheating.
- Install the equipment in a location providing the minimum clearances from all adjacent structures and equipment as directed in this document.
- Install all equipment in accordance with the specifications in the related documentation.

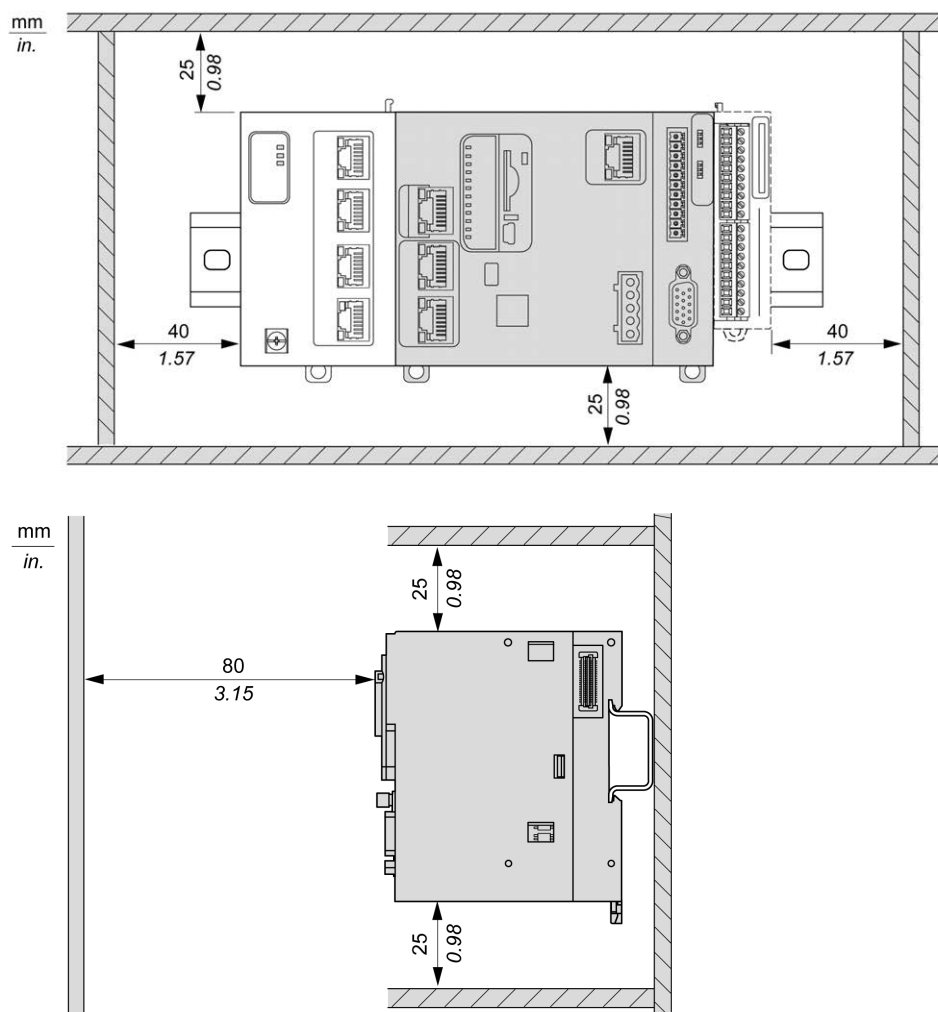
Failure to follow these instructions can result in death, serious injury, or equipment damage.

The M262 Logic/Motion Controller has been designed as an IP20 product and must be installed in an enclosure. Clearances must be respected when installing the product.

There are three types of clearances to consider:

- The M262 Logic/Motion Controller and all sides of the cabinet (including the panel door).
- The M262 Logic/Motion Controller terminal blocks and the wiring ducts to help reduce potential electromagnetic interference between the controller and the duct wiring.
- The M262 Logic/Motion Controller and other heat generating devices installed in the same cabinet.

The following figures show the minimum clearances that apply to all M262 Logic/Motion Controller references:



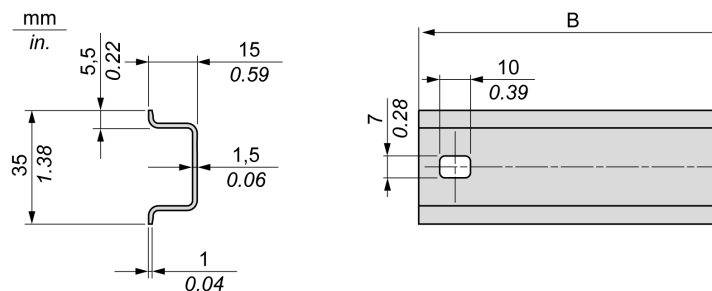
Top Hat Section Rail (DIN rail)

Dimensions of Top Hat Section Rail DIN Rail

You can mount the controller or receiver and their expansions on a 35 mm (1.38 in.) top hat section rail (DIN rail). The DIN rail can be attached to a smooth mounting surface or suspended from a EIA rack or mounted in a NEMA cabinet.

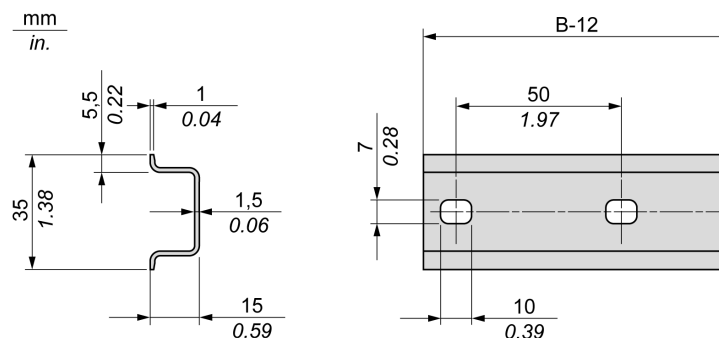
Symmetric Top Hat Section Rails (DIN Rail)

The following illustration and table indicate the references of the top hat section rails (DIN rail) for the wall-mounting range:



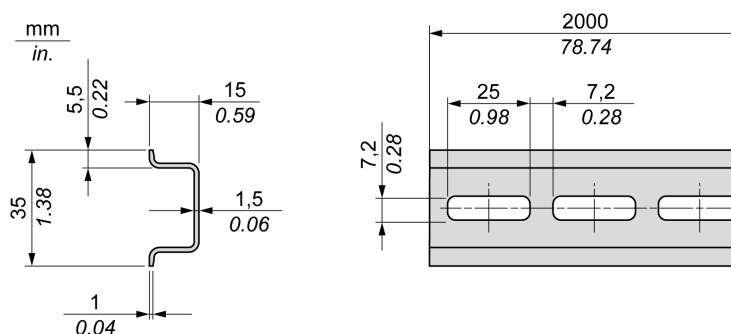
Reference	Type	Rail Length (B)
NSYSDR50A	A	450 mm (17.71 in.)
NSYSDR60A	A	550 mm (21.65 in.)
NSYSDR80A	A	750 mm (29.52 in.)
NSYSDR100A	A	950 mm (37.40 in.)

The following illustration and table indicate the references of the symmetric top hat section rails (DIN rail) for the metal enclosure range:



Reference	Type	Rail Length (B-12 mm)
NSYSDR60	A	588 mm (23.15 in.)
NSYSDR80	A	788 mm (31.02 in.)
NSYSDR100	A	988 mm (38.89 in.)
NSYSDR120	A	1188 mm (46.77 in.)

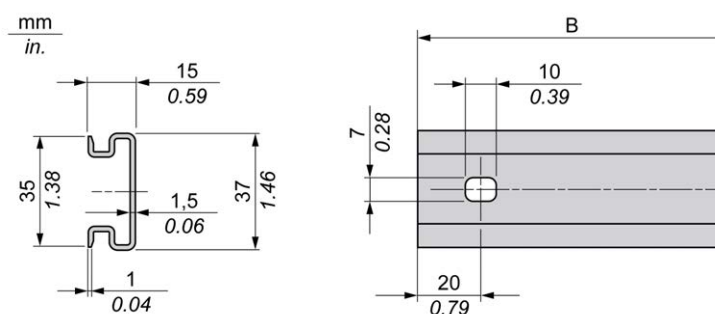
The following illustration and table indicate the references of the symmetric top hat section rails (DIN rail) of 2000 mm (78.74 in.):



Reference	Type	Rail Length
NSYS DR200 ¹	A	2000 mm (78.74 in.)
NSYS DR200D ²	A	
1 Unperforated galvanized steel		
2 Perforated galvanized steel		

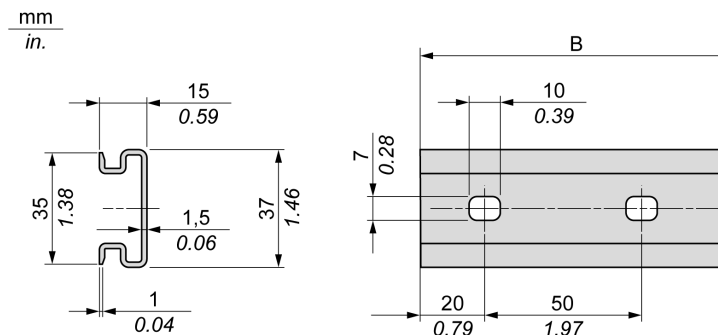
Double-Profile Top Hat Section Rails (DIN rail)

The following illustration and table indicate the references of the double-profile top hat section rails (DIN rails) for the wall-mounting range:



Reference	Type	Rail Length (B)
NSYDPR25	W	250 mm (9.84 in.)
NSYDPR35	W	350 mm (13.77 in.)
NSYDPR45	W	450 mm (17.71 in.)
NSYDPR55	W	550 mm (21.65 in.)
NSYDPR65	W	650 mm (25.60 in.)
NSYDPR75	W	750 mm (29.52 in.)

The following illustration and table indicate the references of the double-profile top hat section rails (DIN rail) for the floor-standing range:



Reference	Type	Rail Length (B)
NSYDPR60	F	588 mm (23.15 in.)
NSYDPR80	F	788 mm (31.02 in.)
NSYDPR100	F	988 mm (38.89 in.)
NSYDPR120	F	1188 mm (46.77 in.)

Installing and Removing the Controller with Expansions

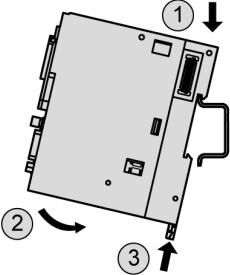
Overview

This section describes how to install and remove the controller with its expansion modules from a top hat section rail (DIN rail).

To assemble expansion modules to the controller, or to other modules, refer to the respective expansion modules hardware guide(s).

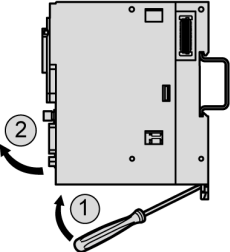
Installing a Controller with its Expansions on a DIN Rail

The following procedure describes how to install a controller with its expansion modules on a top hat section rail (DIN rail):

Step	Action
1	Fasten the top hat section rail (DIN rail) to a panel surface using screws.
2	Position the top groove of the controller and its expansion modules on the top edge of the DIN rail and press the assembly against the top hat section rail (DIN rail) until you hear the top section rail (DIN rail) click into place: 
3	Place 2 terminal block end clamps on both sides of the controller and expansion module assembly.  NOTE: Type NSYTRAAB35 or equivalent terminal block end clamps help minimize sideways movement and improve the shock and vibration characteristics of the controller and expansion module assembly.

Removing a Controller with its Expansions from a Top Hat Section Rail (DIN Rail)

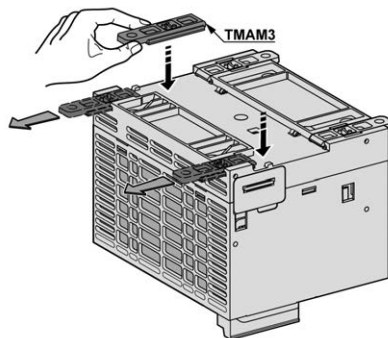
The following procedure describes how to remove a controller with its expansion modules from a top hat section rail (DIN rail):

Step	Action
1	Remove all power from your controller and expansion modules.
2	Insert a flat screwdriver into the slot of the top hat section rail (DIN rail) clip: 
3	Pull down the top hat section rail (DIN rail) clip.
4	Pull the controller and its expansion modules from the top hat section rail (DIN rail) from the bottom.

Mounting a M262 Logic/Motion Controller on a Panel Surface

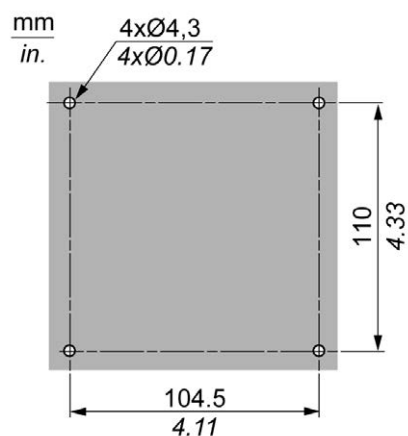
Installing the Panel Mounting Kit

Insert TMAM3, page 28 mounting strips into the slots at the top of the M262 Logic/Motion Controller:



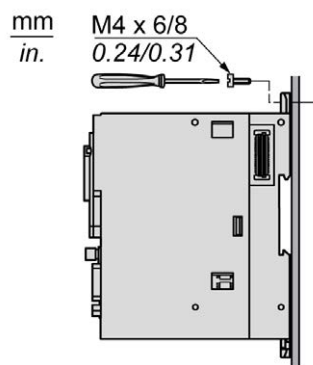
Mounting Holes

The following figure shows the mounting holes for the M262 Logic/Motion Controller:



Verify that the installation panel or cabinet surface is flat (planarity tolerance: 0.5 mm (0.019 in)), in good condition, and has no jagged edges.

Mounting the M262 Logic/Motion Controller on a Metallic Panel



If mounting the controller on a horizontal metallic panel, use flat head screws.

M262 Electrical Requirements

Wiring Best Practices

Overview

This section describes the wiring guidelines and associated best practices to be respected when using the M262 Logic/Motion Controller system.

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Wiring Guidelines

These rules must be applied when wiring a M262 Logic/Motion Controller system:

- Communication wiring must be kept separate from the power wiring. Route these 2 types of wiring in separate cable ducting.
- Verify that the operating conditions and environment are within the specification values.

- Use proper wire sizes to meet voltage and current requirements.
- Use minimum 75 °C (167 °F) copper conductors (required).
- Use twisted pair, shielded cables for encoder, networks, and serial communication connections.

Use shielded, properly grounded cables for all communication connections. If you do not use shielded cable for these connections, electromagnetic interference can cause signal degradation. Degraded signals can cause the controller or attached modules and equipment to perform in an unintended manner.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Use shielded cables for all communication signals.
- Ground cable shields for all communication signals at a single point¹.
- Route communication separately from power cables.

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

¹Multipoint grounding is permissible if connections are made to an equipotential ground plane dimensioned to help avoid cable shield damage in the event of power system short-circuit currents.

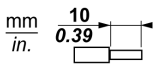
For more details, refer to [Grounding Shielded Cables](#), page 60.

NOTE: Surface temperatures may exceed 60 °C (140 °F).

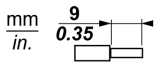
To conform to IEC 61010 standards, route primary wiring (wires connected to power mains) separately and apart from secondary wiring (extra low voltage wiring coming from intervening power sources). If that is not possible, double insulation is required such as conduit or cable gains.

Rules for Spring Terminal Blocks

The following tables show the cable types and wire sizes for the CN7 **5.08 pitch** removable spring terminal block of the embedded 24 Vdc power supply input / alarm relay terminal connector:

					
mm ²	0,2...2,5	0,2...2,5	0,25...2,5	0,25...2,5	2 x 0,5...1
AWG	24...14	24...14	22...14	22...14	2 x 20...18

The following tables show the cable types and wire sizes for the CN8 **3.81 pitch** removable spring terminal block of the embedded I/Os connector:

			
mm ²	0,2...1,5	0,2...1,5	0,25...1,0
AWG	24...16	24...16	23...21

Rules for TMA262SET8S Screw Terminal Blocks

The following tables show the cable types and wire sizes for the CN7 **5.08 pitch** removable screw terminal block of the embedded 24 Vdc power supply input / alarm relay terminal connector:

mm ²	0.2...2.5	0.2...2.5	0.25...2.5	0.25...2.5	2 x 0.2...1	2 x 0.2...1.5	2 x 0.25...1	2 x 0.5...1.5
AWG	24...14	24...14	22...14	22...14	2 x 24...18	2 x 24...16	2 x 22...18	2 x 20...16

		N•m	0.49
Ø 3,5 mm (0.14 in.)		lb-in	4.34

The following tables show the cable types and wire sizes for the CN8 **3.81 pitch** removable screw terminal block of the embedded I/Os connector:

mm ²	0.14...1.5	0.14...1.5	0.25...1.5	0.25...0.5	2 x 0.14...0.5	2 x 0.14...0.75	2 x 0.25...0.34	2 x 0.5
AWG	26...16	26...16	22...16	22...20	2 x 26...20	2 x 26...20	2 x 24...22	2 x 20

		N•m	0.28
Ø 2,5 mm (0.1 in.)		lb-in	2.48

DANGER

LOOSE WIRING CAUSES ELECTRIC SHOCK

Tighten connections in conformance with the torque specifications.

Failure to follow these instructions will result in death or serious injury.

DANGER

FIRE HAZARD

Use only the correct wire sizes for the maximum current capacity of the power supplies.

Failure to follow these instructions will result in death or serious injury.

Protecting Outputs from Inductive Load Damage

Depending on the load, a protection circuit may be needed for the outputs on the controllers and certain modules. Inductive loads using DC voltages may create voltage reflections resulting in overshoot that will damage or shorten the life of output devices.

CAUTION

OUTPUT CIRCUIT DAMAGE DUE TO INDUCTIVE LOADS

Use an appropriate external protective circuit or device to reduce the risk of inductive direct current load damage.

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

If your controller or module contains relay outputs, these types of outputs can support up to 240 Vac. Inductive damage to these types of outputs can result in welded contacts and loss of control. Each inductive load must include a protection

device such as a peak limiter, RC circuit or flyback diode. Capacitive loads are not supported by these relays.

⚠ WARNING

RELAY OUTPUTS WELDED CLOSED

- Always protect relay outputs from inductive alternating current load damage using an appropriate external protective circuit or device.
- Do not connect relay outputs to capacitive loads.

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

AC-driven contactor coils are, under certain circumstances, inductive loads that generate pronounced high-frequency interference and electrical transients when the contactor coil is de-energized. This interference may cause the logic controller to detect an I/O bus error.

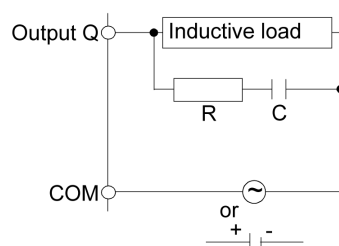
⚠ WARNING

CONSEQUENTIAL LOSS OF CONTROL

Install an RC surge suppressor or similar means, such as an interposing relay, on each TM3 expansion module relay output when connecting to AC-driven contactors or other forms of inductive loads.

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

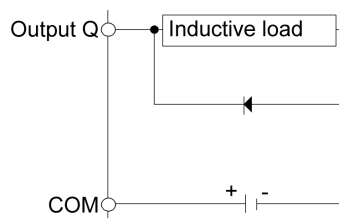
Protective circuit A: this protection circuit can be used for both AC and DC load power circuits.



C Value from 0.1 to 1 μF

R Resistor of approximately the same resistance value as the load

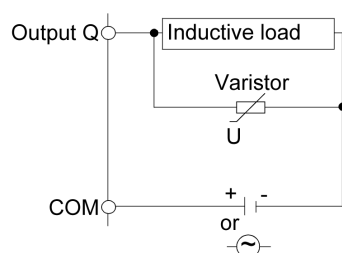
Protective circuit B: this protection circuit can be used for DC load power circuits.



Use a diode with the following ratings:

- Reverse withstand voltage: power voltage of the load circuit x 10.
- Forward current: more than the load current.

Protective circuit C: this protection circuit can be used for both AC and DC load power circuits.



In applications where the inductive load is switched on and off frequently and/or rapidly, ensure that the continuous energy rating (J) of the varistor exceeds the peak load energy by 20 % or more.

DC Power Supply Characteristics and Wiring

Overview

This section provides the characteristics and the wiring diagrams of the DC power supply.

DC Power Supply Voltage Range

If the specified voltage range is not maintained, outputs may not switch as expected. Use appropriate safety interlocks and voltage monitoring circuits.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not exceed any of the rated values specified in the environmental and electrical characteristics tables.

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

DC Power Supply Requirements

The M262 Logic/Motion Controller requires a power supply with a nominal voltage of 24 Vdc. The 24 Vdc power supply must be rated Protective Extra Low Voltage (PELV) according to IEC 61140. This power supply is isolated between the electrical input and output circuits of the power supply.

⚠ WARNING

POTENTIAL OF OVERHEATING AND FIRE

- Do not connect the equipment directly to line voltage.
- Use only isolating PELV power supplies and circuits to supply power to the equipment¹.

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

¹ For compliance to UL (Underwriters Laboratories) requirements, the power supply must also conform to the various criteria of NEC Class 2, and be inherently current limited to a maximum power output availability of less than 100 VA (approximately 4 A at nominal voltage), or not inherently limited but with an additional protection device such as a circuit breaker or fuse meeting the

requirements of clause 9.4 Limited-energy circuit of UL 61010-1. In all cases, the current limit should never exceed that of the electric characteristics and wiring diagrams for the equipment described in the present documentation. In all cases, the power supply must be grounded, and you must separate Class 2 circuits from other circuits. If the indicated rating of the electrical characteristics or wiring diagrams are greater than the specified current limit, multiple Class 2 power supplies may be used.

Controller DC Characteristics

This table shows the characteristics of the DC power supply required for the controller:

Characteristic		Value
Rated voltage		24 Vdc
Power supply voltage range		20.4...28.8 Vdc (ripple $\pm$ 10 % Un)
Power interruption time immunity		Min. 3 ms
Maximum inrush current		40 A
Maximum power consumption		82 W Including 25 W max. available for TM3 expansion modules Including 45 W max. available for TMS expansion modules
Isolation	between DC power supply and internal logic	Not isolated
	between DC power supply and grounding	780 Vdc
Reverse polarity protection		Yes

Power Interruption

The M262 Logic/Motion Controller must be supplied by an external 24 V power supply equipment. During power interruptions, the controller, associated to the suitable power supply, is able to continue normal operation for a minimum of 10 ms as specified by IEC standards.

When planning the management of the power supplied to the controller, you must consider the power interruption duration due to the fast cycle time of the controller.

There could potentially be many scans of the logic and consequential updates to the I/O image table during the power interruption, while there is no external power supplied to the inputs, the outputs or both depending on the power system architecture and power interruption circumstances.

⚠ WARNING

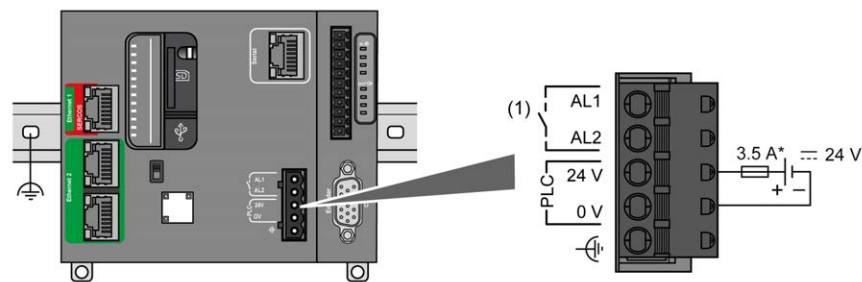
UNINTENDED EQUIPMENT OPERATION

- Individually monitor each source of power used in the controller system including input power supplies, output power supplies and the power supply to the controller to allow appropriate system shutdown during power system interruptions.
- The inputs monitoring each of the power supply sources must be unfiltered inputs.

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

Controller DC Power Supply Wiring Diagram

The following figure shows the wiring of the controller DC power supply:



(1) Alarm Relay

* Type T fuse

For more information on wiring requirements, refer to the Rules for Terminal Blocks, page 54.

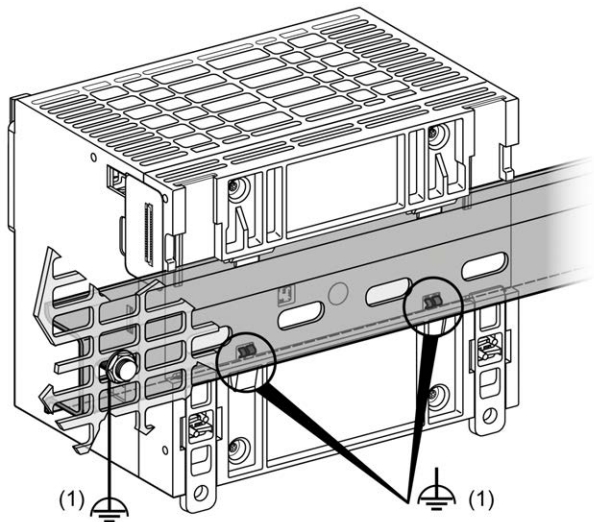
Grounding the M262 Logic/Motion Controller System

Functional Ground (FE) on the DIN Rail

The DIN rail for your M262 Logic/Motion Controller controller is common with the functional ground (FE) plane and must be mounted on a conductive backplane.

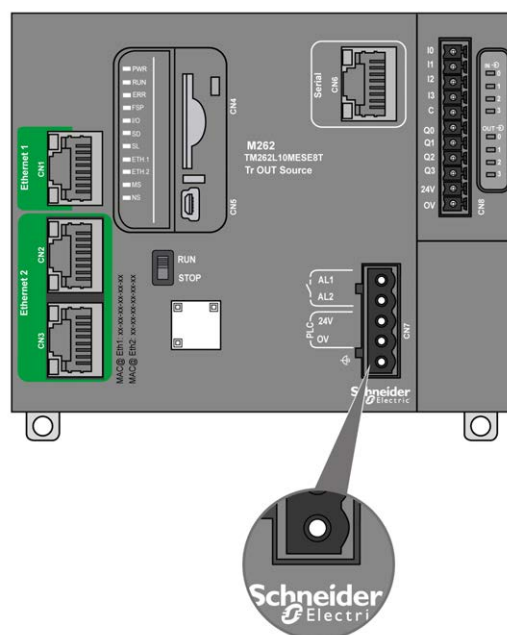
⚠ WARNING UNINTENDED EQUIPMENT OPERATION Connect the DIN rail to the functional ground (FE) of your installation. Failure to follow these instructions can result in death, serious injury, or equipment damage.

The connection between the functional ground (FE) and the M262 Logic/Motion Controller system is made by the DIN rail contacts on the back of the controller and the expansion modules.



1 Functional Ground (FE)

NOTE: When the M262 Logic/Motion Controller system is mounted on a DIN rail, the Functional Ground (FE) connector on the front face of the controller can be used to help minimize electromagnetic interference:

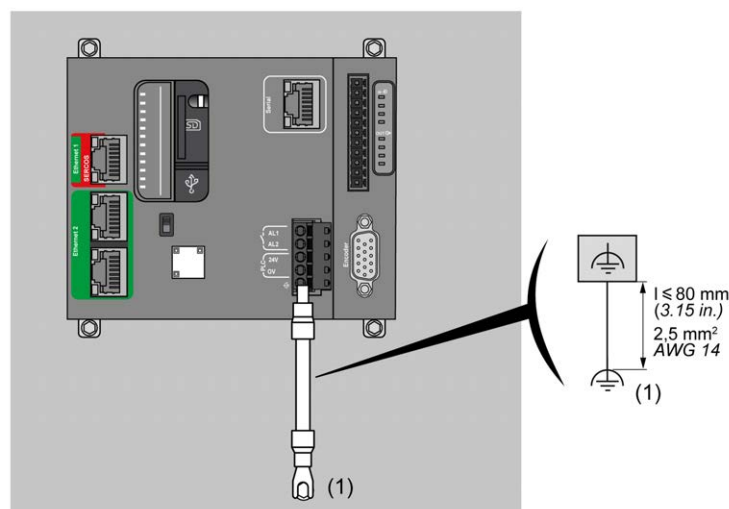


Protective Ground (PE) on the Mounting Panel

The protective ground (PE) should be connected to the conductive mounting panel by a heavy-duty wire, usually a braided copper cable with the maximum allowable cable section.

Functional Ground (FE) on the Mounting Panel

Use a functional ground cable to connect the functional ground connector to the conductive backplane:



(1) Functional ground (FE)

The functional ground cable requires a cross-section of at least 1.5 mm^2 (AWG 16) and a maximum length of 80 mm (3.15 in.).

Shielded Cables Connections

To help minimize the effects of electromagnetic interference, cables carrying fieldbus communication signals must be shielded.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Use shielded cables for communication signals.
- Ground cable shields for communication signals at a single point ¹.
- Always comply with local wiring requirements regarding grounding of cable shields.

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

¹Multipoint grounding is permissible if connections are made to an equipotential ground plane dimensioned to help avoid cable shield damage in the event of power system short-circuit currents.

The use of shielded cables requires compliance with the following wiring rules:

- For protective ground connections (PE), metal conduit or ducting can be used for part of the shielding length, provided there is no break in the continuity of the ground connections. For functional ground (FE), the shielding is intended to attenuate electromagnetic interference and the shielding must be continuous for the length of the cable. If the purpose is both functional and protective, as is often the case for communication cables, the cable must have continuous shielding.
- Wherever possible, keep cables carrying one type of signal separate from the cables carrying other types of signals or power.

The shielding must be securely connected to ground. The fieldbus communication cable shields must be connected to the protective ground (PE) with a connecting clamp secured to the conductive backplane of your installation.

The shielding of the following cables must be connected to the protective ground (PE):

- Ethernet (unless forbidden by an applicable standard)
- Serial
- Encoder (on TM262M• references)

The embedded I/O shields can be connected to either the protective ground (PE) or the functional ground (FE).

DANGER

HAZARD OF ELECTRIC SHOCK

- The grounding terminal connection (PE) must be used to provide a protective ground at all times.
- Make sure that an appropriate, braided ground cable is attached to the PE/PG ground terminal before connecting or disconnecting the network cable to the equipment.

Failure to follow these instructions will result in death or serious injury.

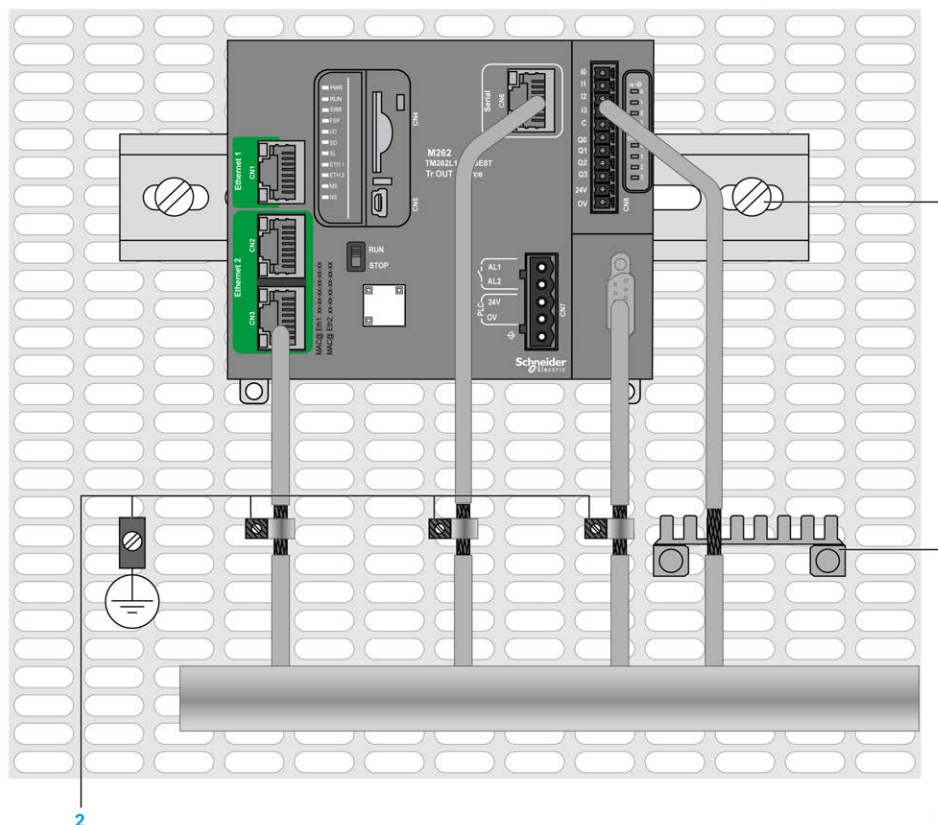
⚠ WARNING

ACCIDENTAL DISCONNECTION FROM PROTECTIVE GROUND (PE)

- Do not use the TM2XMTGB Grounding Plate to provide a protective ground (PE).
- Use the TM2XMTGB Grounding Plate only to provide a functional ground (FE).

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

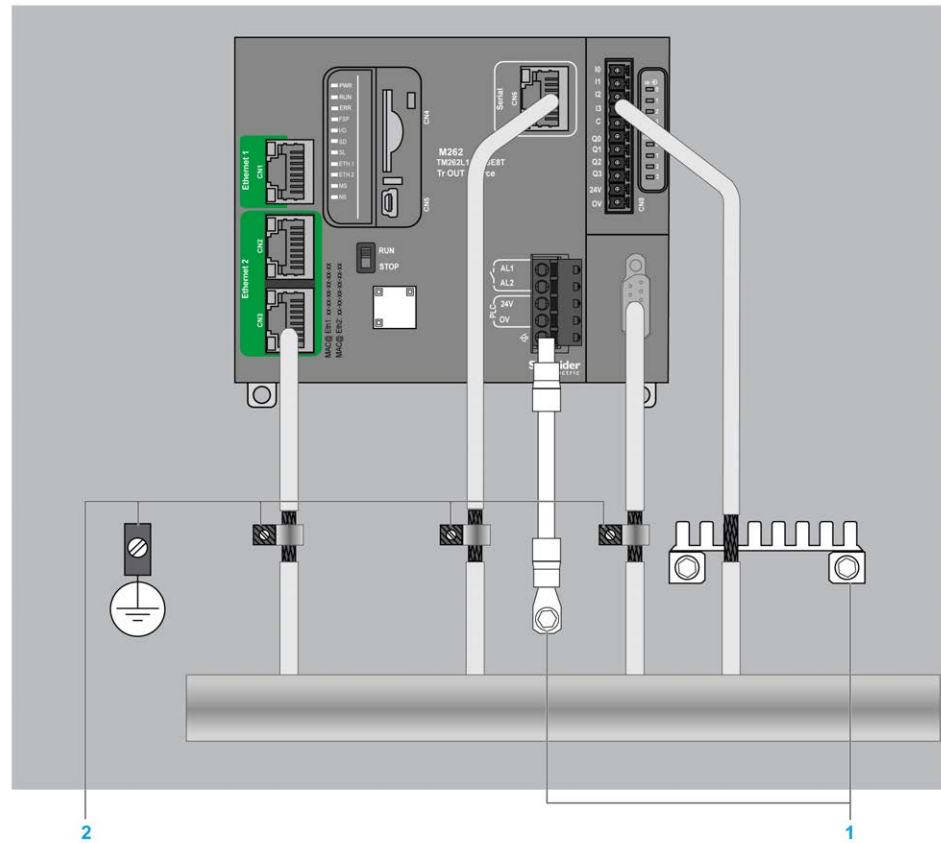
The figure below represents an M262 Logic/Motion Controller with shielded cables connected to a DIN rail:



1 Functional ground (FE)

2 Protective ground (PE)

The figure below represents an M262 Logic/Motion Controller with shielded cables connected to a mounting panel:

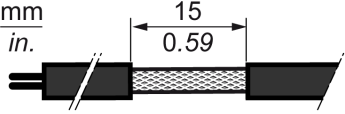
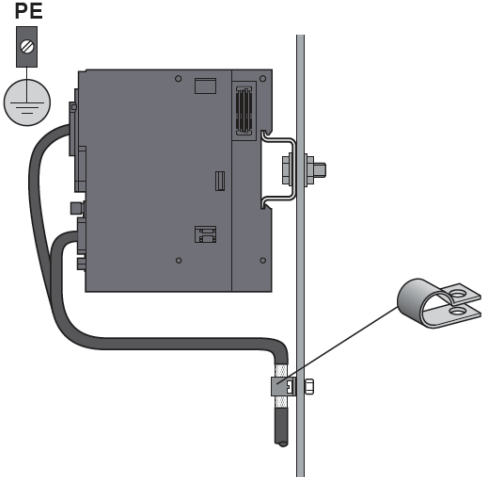


1 Functional ground (FE)

2 Protective ground (PE)

Protective Ground (PE) Cable Shielding

To ground the shield of a cable via a grounding clamp:

Step	Description	
1	Strip the shielding for a length of 15 mm (0.59 in.)	
2	Attach the cable to the conductive backplane plate by attaching the grounding clamp to the stripped part of the shielding as close as possible to the base of the M262 Logic/Motion Controller.	

NOTE: The shielding must be clamped securely to the conductive backplane to help ensure good contact.

Functional Ground (FE) Cable Shielding

Connect the shield of a cable via the grounding bar:

Step	Description	
1	Install the TM2XMTGB grounding bar directly on the conductive backplane below the M262 Logic/Motion Controller as illustrated.	
2	Strip the shielding for a length of 15 mm (0.59 in.)	
3	Tightly clamp on the blade connector (1) using a nylon fastener (2) (width 2.5...3 mm (0.1...0.12 in.)) and appropriate tool.	

Alarm Relay Wiring

Overview

The M262 Logic/Motion Controller has integrated relay connections that can be wired to an external alarm.

Wiring Stripping and Wire Sizes

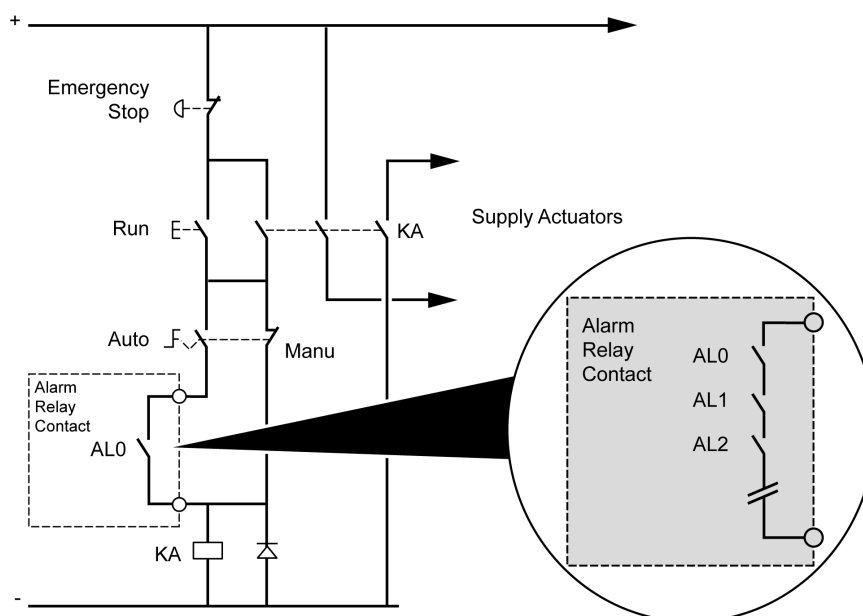
The alarm relay is wired by means of a 5.08 mm pitch removable screw terminal block on the front face of the M262 Logic/Motion Controller. For details, refer to Rules for Terminal Blocks, page 54.

Using the Alarm Relay for the Actuator Power Supply

Proceed as follows to use the Alarm relay for the actuator power supply:

Step	Action
1	Switch on the power supply of the M262 Logic/Motion Controller using the main contactor.
2	When the M262 Logic/Motion Controller is powered on, switch on the output power supply for the actuators using the KA contactor. The following wiring diagram shows an M262 Logic/Motion Controller supplied by direct current: In AUTO run mode, the KA contactor is controlled by the alarm relay from the power supply module.

If your system comprises multiple M262 Logic/Motion Controllers installed in multiple racks, set the alarm relay contacts in all controllers in series (AL0, AL1, AL2, and so on), as shown in the following diagram:



Modicon M262 Logic/Motion Controller

TM262L01MESE8T Presentation

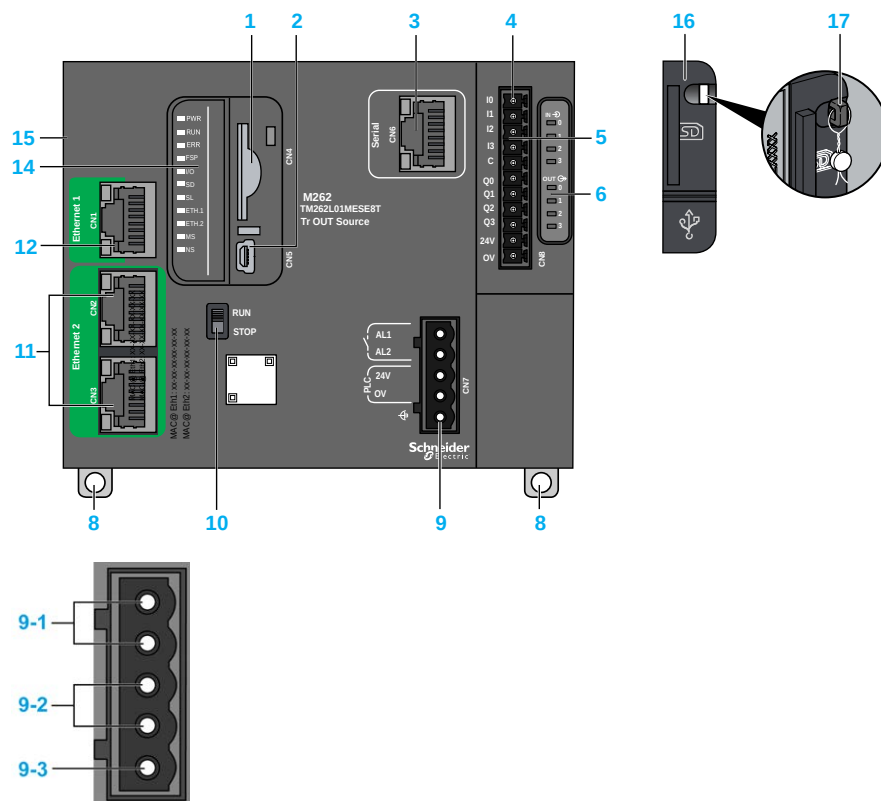
Overview

The TM262L01MESE8T logic controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port

Description

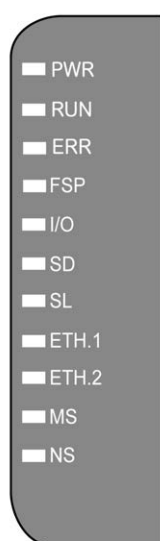
The following figure shows the different components of the TM262L01MESE8T logic controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101
		Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103
		Fast Outputs Status LEDs, page 107
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 port, page 116
12	Ethernet port 1	Ethernet 1 port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

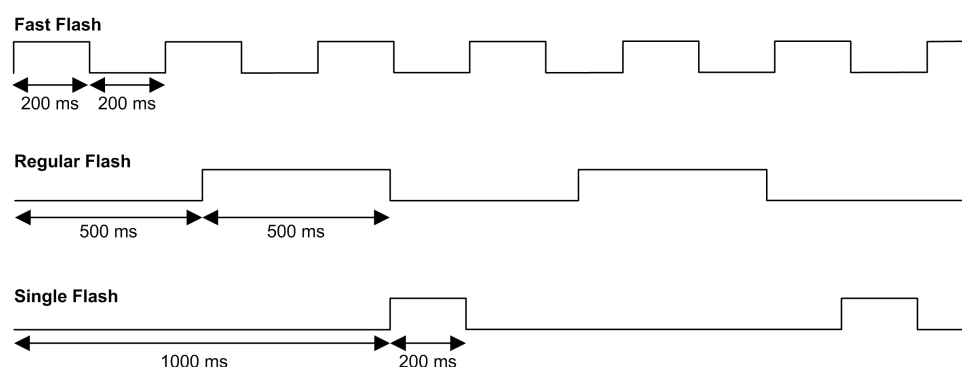


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an application error (exception) has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing regularly, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

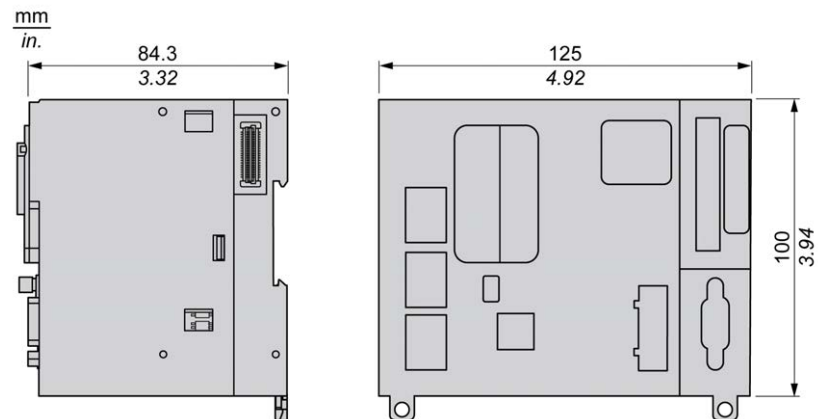
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262L01MESE8T logic controller:



Weight

655 g

TM262L10MESE8T Presentation

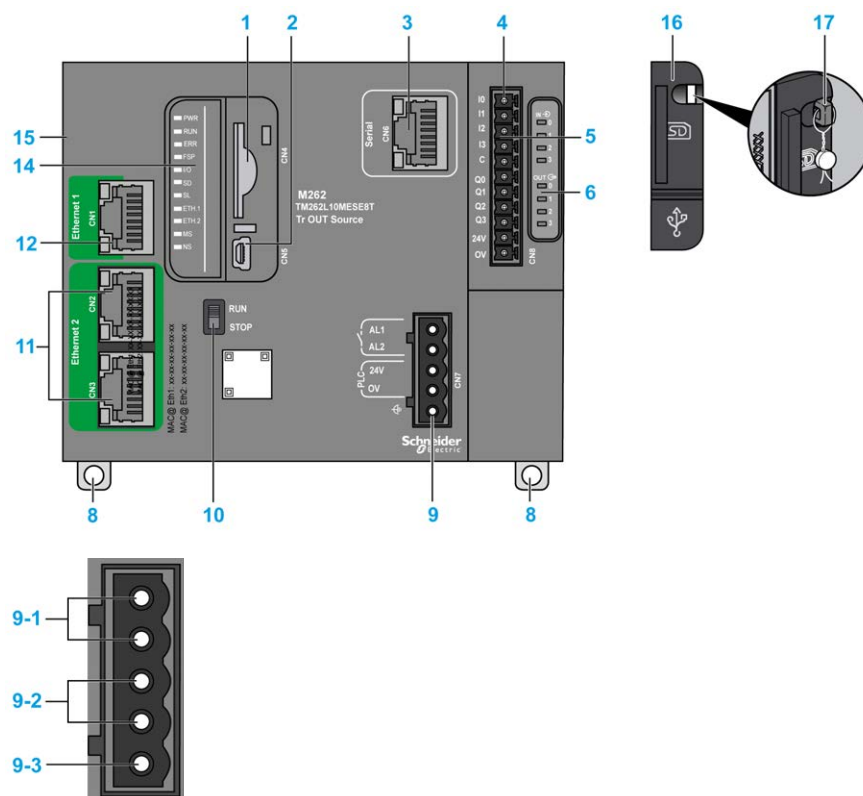
Overview

The TM262L10MESE8T logic controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port

Description

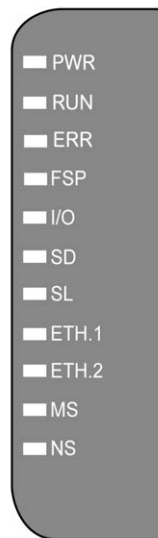
The following figure shows the different components of the TM262L10MESE8T logic controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101 Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103 Fast Outputs Status LEDs, page 107
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 port, page 116
12	Ethernet port 1	Ethernet 1 port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

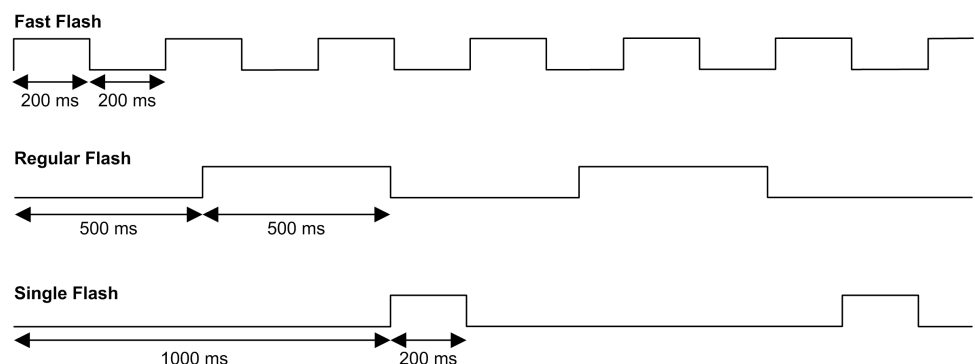


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an application error (exception) has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing regularly, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_LwSystemFault_1</code> and <code>i_LwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

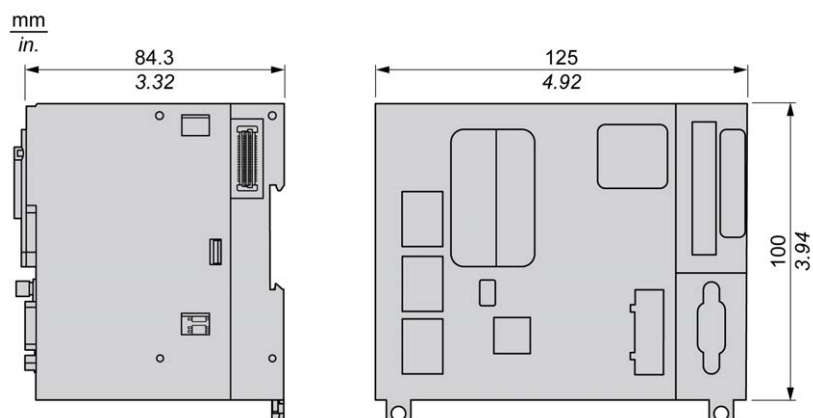
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262L10MESE8T logic controller:



Weight

655 g

TM262L20MESE8T Presentation

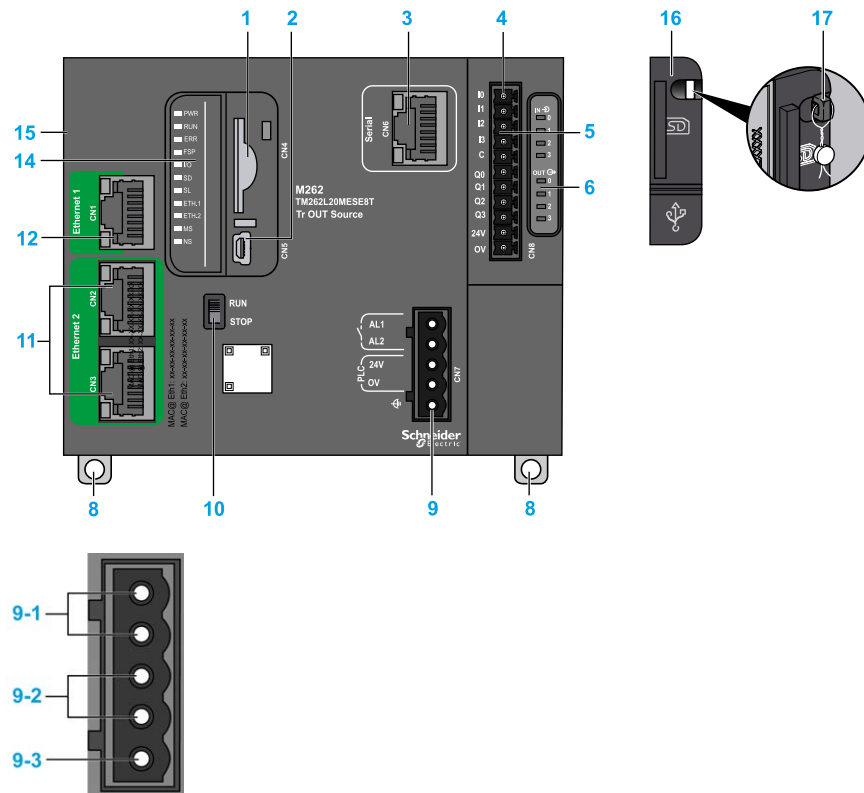
Overview

The TM262L20MESE8T logic controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port

Description

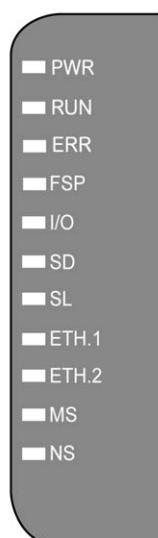
The following figure shows the different components of the TM262L20MESE8T logic controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101 Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103 Fast Outputs Status LEDs, page 107
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 port, page 116
12	Ethernet port 1	Ethernet 1 port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

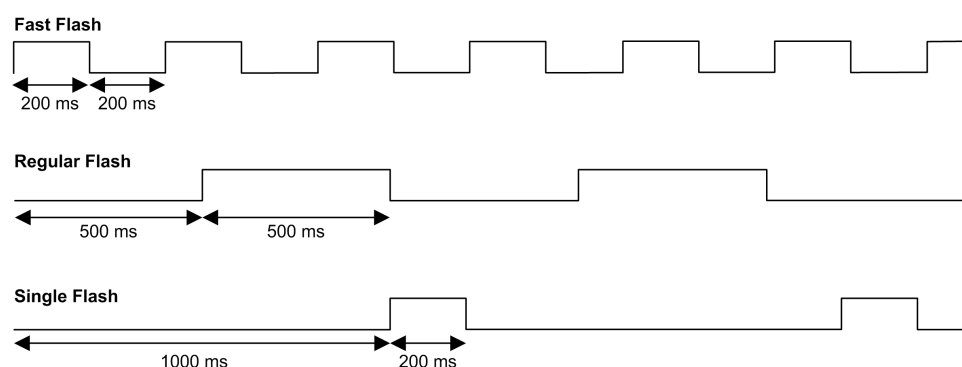


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an application error (exception) has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing regularly, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

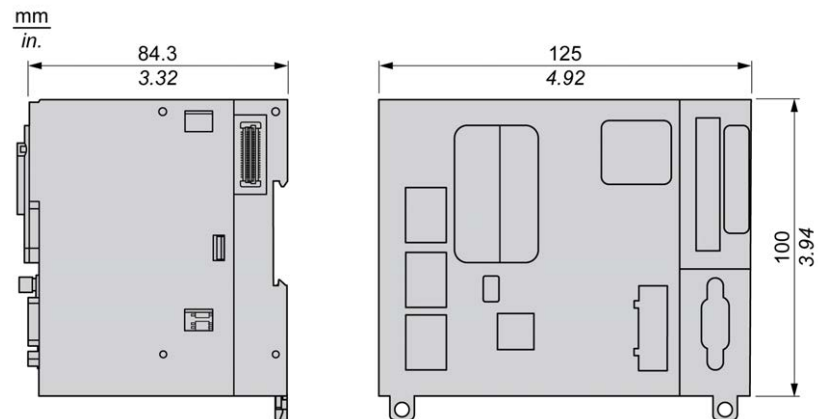
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262L20MESE8T logic controller:



Weight

655 g

TM262M05MESS8T Presentation

Overview

The TM262M05MESS8T motion controller has:

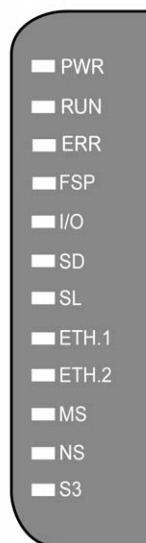
- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port for fieldbus with Sercos interface
- Encoder interface (SSI/incremental)

Diagram of the M262 TM262M96MESS8T Ti OUT Source device with numbered callouts 1-17. The device features various ports including Ethernet 1 (S/BCOS), Ethernet 2 (CN2), Ethernet 3 (CN3), Serial (CN6), and an Encoder (CN9). It also has a RUN/STOP button, a 24V OV terminal block, and a USB port. A circular inset shows a close-up of the USB port with callouts 16 and 17.

N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101
		Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103
		Fast Outputs Status LEDs, page 103
7	Encoder connector	Encoder Interface, page 109
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 Port, page 116
13	Ethernet 1/Sercos port	Ethernet 1 Port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

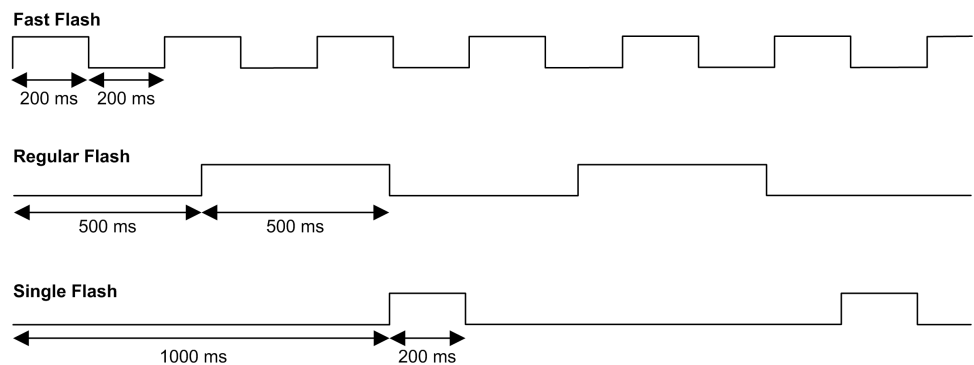


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an operating system error has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware or hardware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

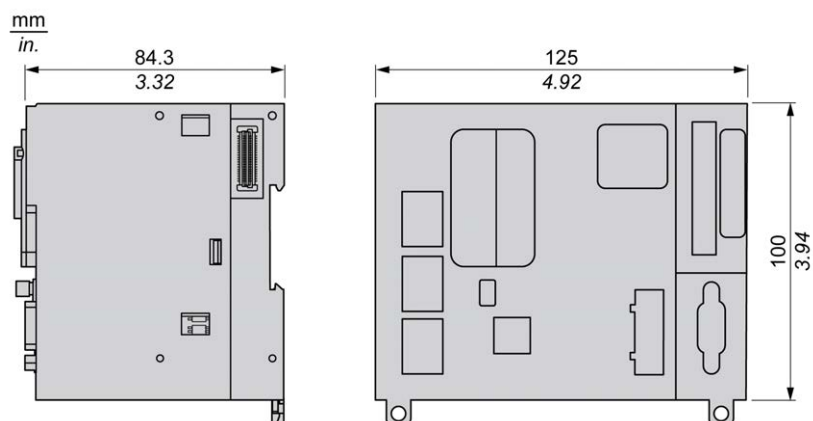
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
S3	Sercos 3 master status	-	OFF	No Sercos 3 communication.
		Orange	ON	Sercos 3 initialization (phase-up) in progress.
		Green	ON	Sercos 3 operational.
		Red	ON	Sercos 3 error.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262M05MESS8T motion controller:



Weight

670 g

TM262M15MESS8T Presentation

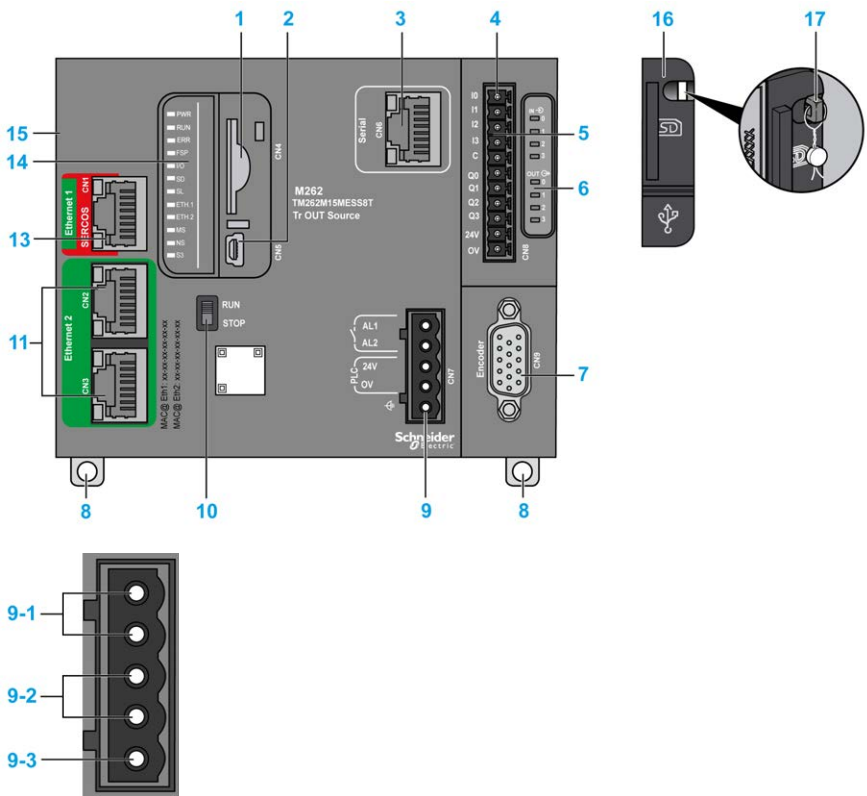
Overview

The TM262M15MESS8T motion controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port for fieldbus with Sercos interface
- Encoder interface (SSI/incremental)

Description

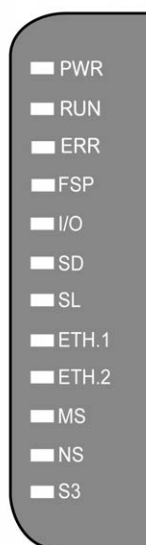
The following figure shows the different components of the TM262M15MESS8T motion controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101
		Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103
		Fast Outputs Status LEDs, page 103
7	Encoder connector	Encoder Interface, page 109
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 Port, page 116
13	Ethernet 1/Sercos port	Ethernet 1 Port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

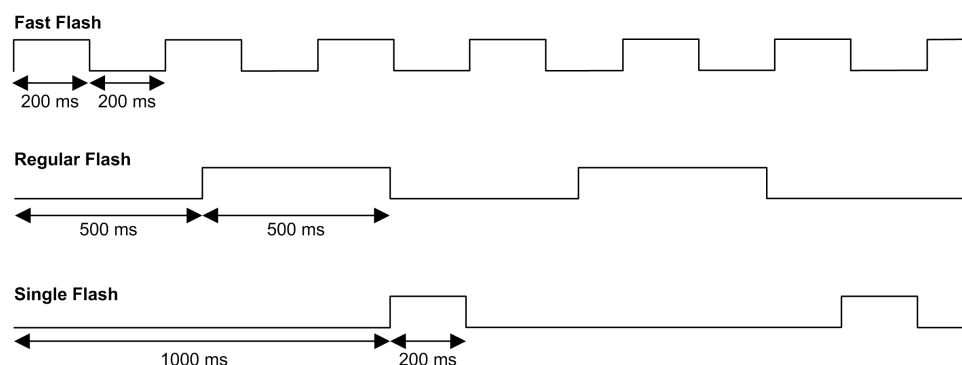


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an operating system error has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware or hardware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

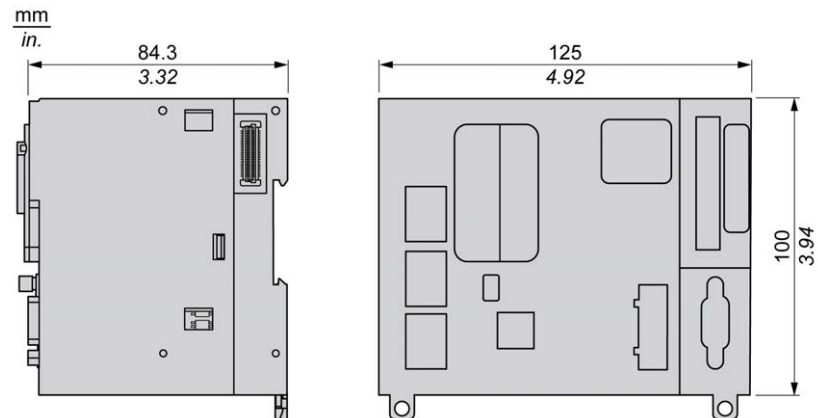
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
S3	Sercos 3 master status	-	OFF	No Sercos 3 communication.
		Orange	ON	Sercos 3 initialization (phase-up) in progress.
		Green	ON	Sercos 3 operational.
		Red	ON	Sercos 3 error.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262M15MESS8T motion controller:



Weight

670 g

TM262M25MESS8T Presentation

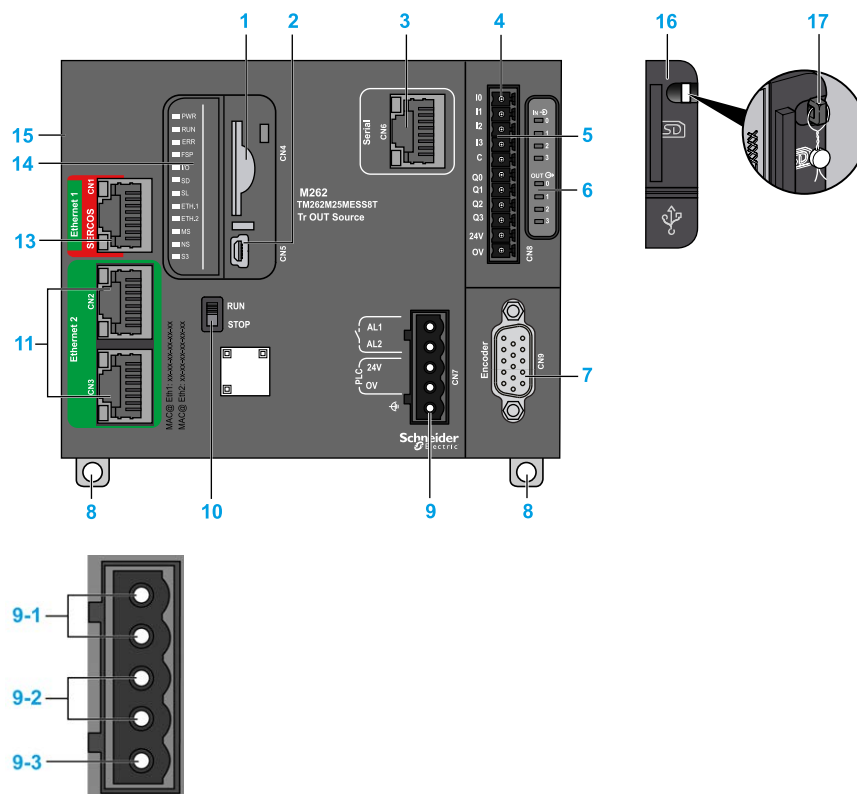
Overview

The TM262M25MESS8T motion controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port for fieldbus with Sercos interface
- Encoder interface (SSI/incremental)

Description

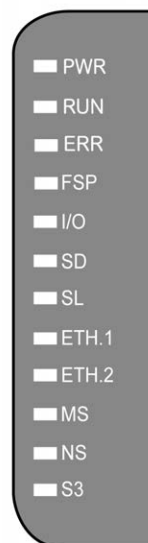
The following figure shows the different components of the TM262M25MESS8T motion controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101
		Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103
		Fast Outputs Status LEDs, page 103
7	Encoder connector	Encoder Interface, page 109
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 Port, page 116
13	Ethernet 1/Sercos port	Ethernet 1 Port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

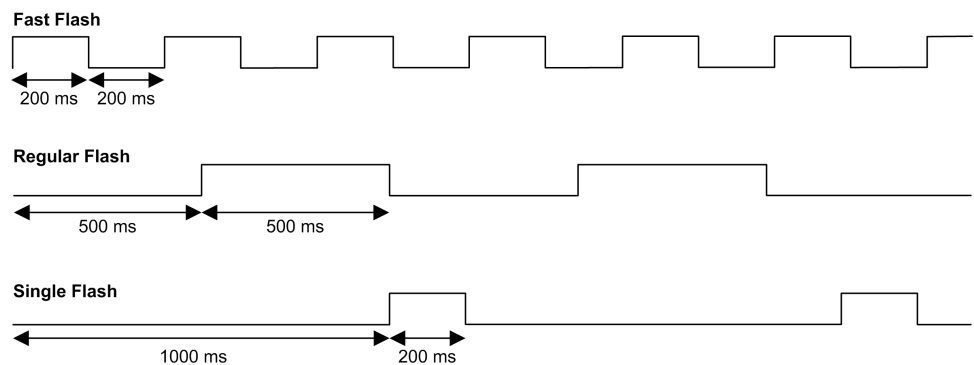


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an operating system error has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware or hardware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

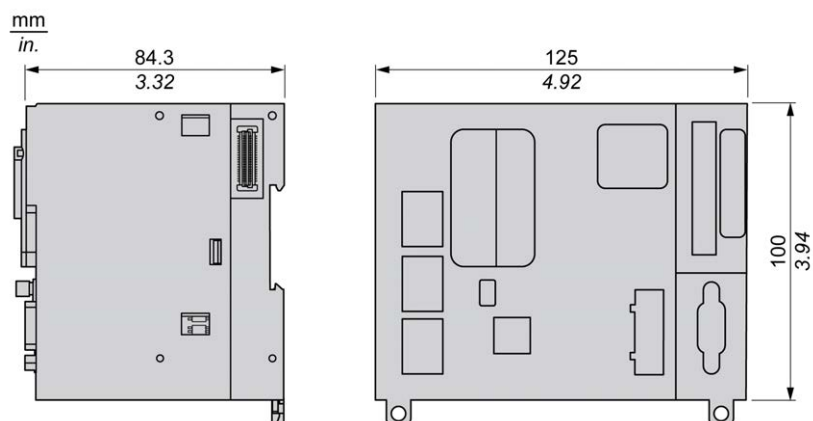
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
S3	Sercos 3 master status	-	OFF	No Sercos 3 communication.
		Orange	ON	Sercos 3 initialization (phase-up) in progress.
		Green	ON	Sercos 3 operational.
		Red	ON	Sercos 3 error.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262M25MESS8T motion controller:



Weight

670 g

TM262M35MESS8T Presentation

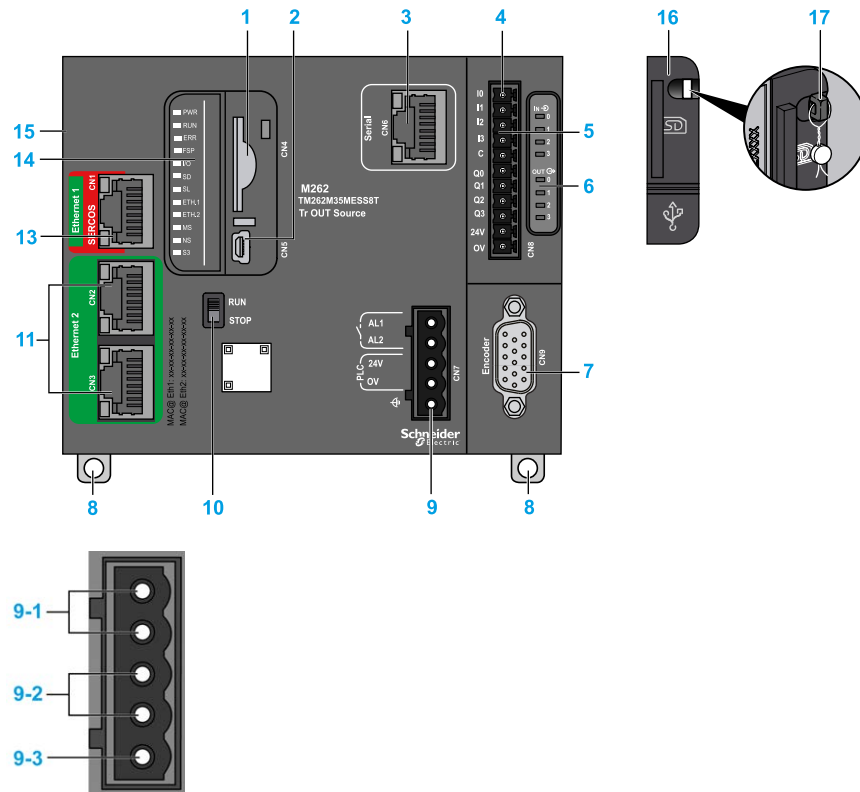
Overview

The TM262M35MESS8T motion controller has:

- 4 fast digital inputs
- 4 fast digital outputs (source)
- Communication ports:
 - 1 serial line port
 - 1 USB mini-B programming port
 - 2 Ethernet switched ports
 - 1 Ethernet port for fieldbus with Sercos interface
- Encoder interface (SSI/incremental)

Description

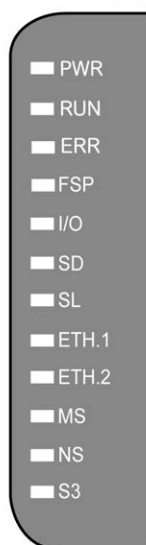
The following figure shows the different components of the TM262M35MESS8T motion controller:



N°	Description	Refer to
1	SD card slot	SD Card, page 33
2	USB mini-B programming port for terminal connection to a programming PC (EcoStruxure Machine Expert)	USB Mini-B Programming Port , page 118
3	Serial line port / type RJ45 (RS-232 or RS-485)	Serial Line, page 119
4	Inputs/outputs terminal connector	Embedded Digital Inputs, page 101
		Embedded Digital Outputs, page 104
5	TM3 bus connector	TM3 Expansion Modules, page 17
6	I/O status LEDs	Fast Inputs Status LEDs, page 103
		Fast Outputs Status LEDs, page 103
7	Encoder connector	Encoder Interface, page 109
8	Clip-on lock for 35 mm (1.38 in.) top hat section rail (DIN rail)	Installing and Removing the Controller with Expansions, page 49
9-1	Alarm relay terminal connector	Alarm Relay, page 37
9-2	24 Vdc power supply	DC Power supply Characteristics and Wiring, page 56
9-3	Functional Earth (FE) grounding connection	Grounding the M262 Logic/Motion Controller, page 58
10	Run/Stop switch	Run/Stop, page 32
11	Dual port Ethernet switch	Ethernet 2 Port, page 116
13	Ethernet 1/Sercos port	Ethernet 1 Port, page 114
14	Status LEDs	See below
15	TMS bus connector	TMS Expansion Modules (see Modicon M262 Logic/Motion Controller, Programming Guide)
16	Protective cover (for SD card slot and USB mini-B programming port)	-
17	Locking hook (optional lock not included)	-

Status LEDs

This figure shows the status LEDs:

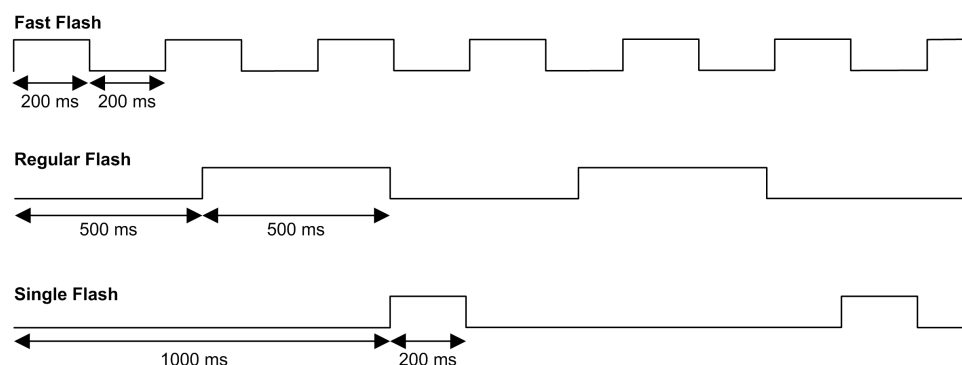


The following table describes the system status LEDs:

Label	Function Type	Color	Status	Description
PWR	Power	Green/Red	Green OFF/Red OFF	Indicates that power is removed.
			Green ON/Red OFF	Indicates that power is applied, normal operation.
			Green ON/Red 1 flash	Elevated internal operating temperature detected (over 80° C/ 176° F). Take appropriate measures to reduce the temperature.
			Green ON/Red 2 flashes	Detected error on TM3 power.
			Green ON/Red 3 flashes	Detected error on TMS power.
			Green ON/Red 4 flashes	Detected error on Serial line power.
RUN	Machine status	Green	ON	Indicates that the controller is running a valid application.
			Regular flash	Indicates that the controller is running a valid application that is stopped.
			Single flash	Indicates that the controller is running a valid application that is stopped at a breakpoint.
			OFF	Indicates that the controller does not contain a valid application.
ERR	Internal Error	Red	ON	Indicates that an operating system error has been detected. The RUN LED is flashing to indicate that the application is stopped.
			Fast flash	Indicates that the controller has detected a firmware or hardware error.
			Regular flash	Indicates either that a minor error has been detected if RUN is ON or flashing, or that no application has been detected if RUN is OFF.
FSP	Forced stop	Red	ON	Indicates that the Run/Stop switch or Run/stop input has been activated to force the controller to the STOPPED state.
			Regular flash	Indicates that at least one application variable is being forced.
I/O	I/O error	Red	ON	Indicates that I/O or expansion module errors have been detected. More details on the error detected are provided by the system variables <code>i_lwSystemFault_1</code> and <code>i_lwSystemFault_2</code> (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide), and on the Diagnostics tab of the controller Web site (see Modicon M262 Logic/Motion Controller, Programming Guide).
SD	SD card access	Green	ON	Indicates that a firmware update is completed.
		Green	Regular flash	Indicates that a firmware update or script execution is in progress.
		Yellow	ON	Indicates that a firmware update or script execution is unsuccessful. NOTE: If the script file is not executed, a log file is generated. The log file location in the controller is <code>/usr/Syslog/FWLog.txt</code> .
		Yellow	Regular flash	Indicates that the SD card is being accessed (script execution in progress).
		-	OFF	No SD card activity.
SL	Serial line	Yellow	Flashing	Indicates communication on the serial line.
			OFF	Indicates no serial communication.

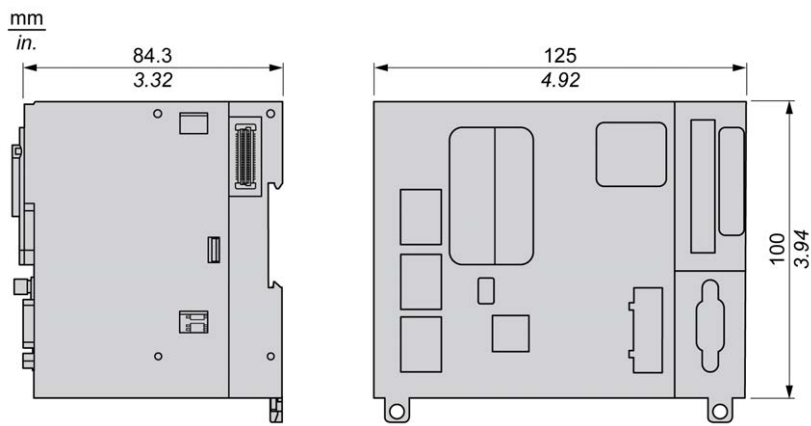
Label	Function Type	Color	Status	Description
ETH.1 ETH.2	Ethernet port status	Green	ON	Indicates that the Ethernet port is connected and the IP address is defined.
			3 flashes	Indicates that the Ethernet port is not connected.
			4 flashes	Address conflict detected. Indicates that the configured IP address is already in use.
			5 flashes	Indicates that the address is the default address. The module is waiting for a BOOTP or DHCP sequence.
			6 flashes	Indicates that the configured IP address is not valid. The default IP address is being used.
			OFF	Indicates that the Ethernet port is not configured.
MS	EtherNet/IP controller interface status	Red	ON	Indicates that an unrecoverable error has been detected.
			Regular flash	Indicates that a recoverable error has been detected.
		Green	ON	Indicates that the controller interface is functioning normally.
			Regular flash	Indicates that the configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
NS	EtherNet/IP network status	Red	ON	Indicates that one or more connections timed out, or that an error is preventing network communications (duplicate IP address, or bus powered off)
			Regular flash	Indicates that a recoverable error has been detected, for example, one or more connections timed out.
		Green	ON	Indicates that the controller interface is functioning normally and network connections are established.
			Regular flash	Indicates that the controller interface is operating normally, but network connections have not been established, or the network configuration is missing, incomplete, or incorrect.
		Red/Green	Regular flash	Indicates that an error has been detected.
		-	OFF	Indicates that the controller is powered off.
S3	Sercos 3 master status	-	OFF	No Sercos 3 communication.
		Orange	ON	Sercos 3 initialization (phase-up) in progress.
		Green	ON	Sercos 3 operational.
		Red	ON	Sercos 3 error.

This timing diagram shows the difference between the fast flash, regular flash and single flash:



Dimensions

The following figure shows the external dimensions of the TM262M35MESS8T motion controller:



Weight

670 g

Embedded I/O Channels

Overview

This chapter describes the embedded I/O channels.

Digital Inputs

Overview

The Modicon M262 Logic/Motion Controller has 4 embedded fast digital inputs.

The digital inputs are connected on the front face of the controller.

DANGER

FIRE HAZARD

Use only the correct wire sizes for the maximum current capacity of the I/O channels and power supplies.

Failure to follow these instructions will result in death or serious injury.

WARNING

UNINTENDED EQUIPMENT OPERATION

Do not exceed any of the rated values specified in the environmental and electrical characteristics tables.

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

Digital Input Characteristics

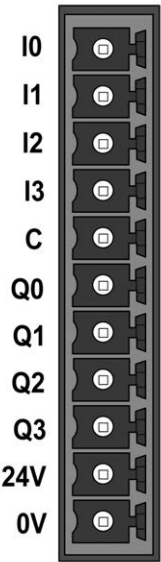
This table presents the characteristics of the digital inputs:

Characteristic		Value
Number of input channels		4 (I0...I3)
Input type		IEC61131-2 Type 1
Logic type		Sink
Rated power supply voltage		24 Vdc
Voltage limit		30 Vdc
Rated input current		7.5 mA
Input impedance		2.81 kΩ
Input limit values	Voltage at state 1	> 15 Vdc (15...30 Vdc)
	Voltage at state 0	< 5 Vdc (0...5 Vdc)
	Current at state 1	> 3 mA
	Current at state 0	< 1.5 mA
Input delay	Turn on time	< 1 μs + filter delay
	Turn off time	< 1 μs + filter delay
Isolation	Between input channels	No
	Between input and internal logic	550 Vac for 1 min.
	Between input and output	550 Vac for 1 min.
Cable	Type	Shielded cable, including COM signal
	Length	10 m (32.8 ft) max.
Connection type		Removable spring terminal block
Connector insertion/removal durability		Over 100 times

Pin Assignment

The digital inputs are connected on the front face of the controller.

This illustration describes the pin assignment of the connector:

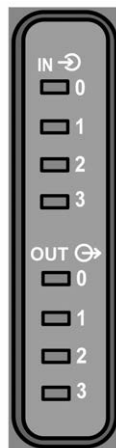


This table describes the pin assignment of the embedded I/O connector:

Pin	Label	Description
1	I0	Digital input 0
2	I1	Digital input 1
3	I2	Digital input 2
4	I3	Digital input 3
5	C	Inputs common port

Status LEDs

This figure shows the I/O status LEDs:



LED	Color	Status	Description
0...3	Green	On	The corresponding input channel is activated
		Off	The corresponding input channel is deactivated

NOTE: The LEDs indicate the logic state of each input.

Wiring Rules

See Wiring Best Practices, page 52.

Electromagnetic perturbations may cause the application to operate in an unexpected manner.

⚠ WARNING

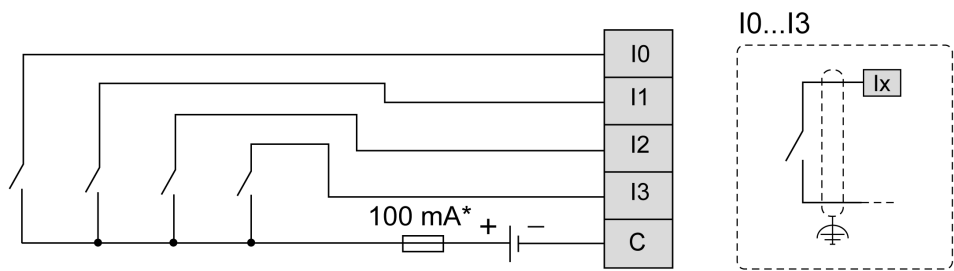
UNINTENDED EQUIPMENT OPERATION

- Adapt the programmable filtering to the frequency applied at the inputs.
- Use shielded cables wherever specified, connected to the functional ground using the TM2XMTGB grounding bar, page 28.
- Use a specific 24 Vdc supply for inputs and outputs.

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

Wiring Diagram

This illustration presents the fast inputs wiring diagram:



* Type T fuse

Digital Outputs

Overview

The Modicon M262 Logic/Motion Controller has 4 embedded fast digital outputs. The digital outputs are connected on the front face of the controller.

⚠ DANGER

FIRE HAZARD

Use only the correct wire sizes for the maximum current capacity of the I/O channels and power supplies.

Failure to follow these instructions will result in death or serious injury.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not exceed any of the rated values specified in the environmental and electrical characteristics tables.

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

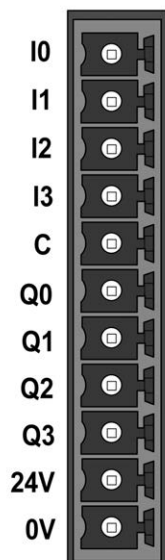
Fast Outputs Characteristics

The table below describes the characteristics of the embedded digital outputs:

Characteristic		Value
Number of output channels		4 outputs (Q0...Q3)
Output type		Transistor
Output signal type		Source (push-pull)
Rated output voltage		24 Vdc
Output current		500 mA
Total output current		2 A
Leakage current when switched off		< 0.01 mA
Maximum power of filament lamp		1.5 W max.
Turn on time		1 µs max.
Turn off time		1 µs max.
Protection against short circuit or overload		Yes. Typical current 5 A per output. Defect managed by group: Q0...Q3
Automatic rearming after short circuit or overload		Yes, 10 sec. (enabled/disabled by EcoStruxure Machine Expert software)
Isolation	Between output channels	No
	Between output and internal logic	550 Vac for 1 minute
	Between output and input	550 Vac for 1 minute
Cable length		< 30 m (98.4 ft)
Connection type		Removable spring terminal block
Connector insertion/removal durability		Over 100 times
NOTE: Refer to Protecting Outputs from Inductive Load Damage , page 54 for additional information concerning output protection.		

Pin Assignment

This illustration describes the pin assignment of the connector:



This table describes the pin assignment of the embedded I/O connector:

Pin	Label	Description
6	Q0	Digital output 0
7	Q1	Digital output 1
8	Q2	Digital output 2
9	Q3	Digital output 3
10	24V	Outputs and encoder 24 Vdc power supply
11	0V	Outputs and encoder 0 Vdc power supply

Outputs/Encoder Power Supply Characteristics

This table shows the characteristics of the power supply provided by the controller to the embedded digital outputs and the encoder interface, page 109.

Characteristic	Value
Nominal voltage	24 Vdc
Power supply voltage range	20.4...28.8 Vdc (ripple $\pm$ 10% Un)
Power supply type	PELV
Maximum input current	2.6 A
Inrush current	Not limited
Voltage drop immunity	No
Reverse polarity protection	Yes
Overload protection	No. Non-replaceable 4 A slow fuse
Overvoltage protection	No
Voltage presence detection	Yes, typically >16 V I/O Status Codes (see Modicon M262 Logic/Motion Controller, System Functions and Variables, System Library Guide) diagnostic is available in EcoStruxure Machine Expert software
Isolation	550 Vac for 1 minute
Cable length	< 3 m (9.84 ft)

Status LEDs

This figure shows the I/O status LEDs:



LED	Color	Status	Description
0...3	Green	On	The corresponding output channel is activated
		Off	The corresponding output channel is deactivated

NOTE: The LEDs indicate the logic state of each output.

Wiring Rules

See Wiring Best Practices, page 52.

Electromagnetic perturbations may cause the application to operate in an unexpected manner.

⚠ WARNING

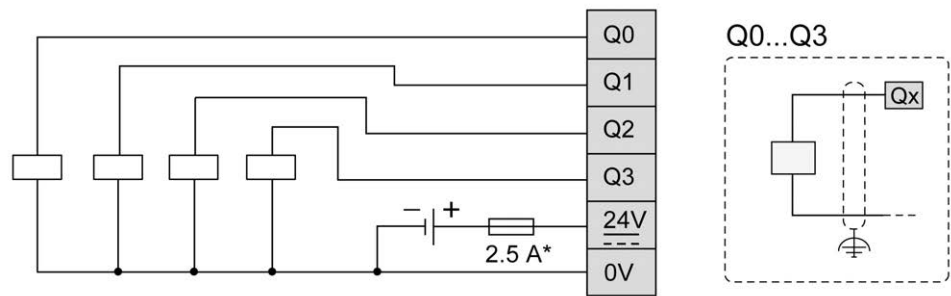
UNINTENDED EQUIPMENT OPERATION

- Adapt the programmable filtering to the frequency applied at the inputs.
- Use shielded cables wherever specified, connected to the functional ground using the TM2XMTGB grounding bar, page 28.
- Use a specific 24 Vdc supply for inputs and outputs.

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

Fast Outputs Wiring Diagram

This illustration presents the fast outputs wiring diagram:



* Use a type T fuse appropriate for the load, not to exceed 2.5 A

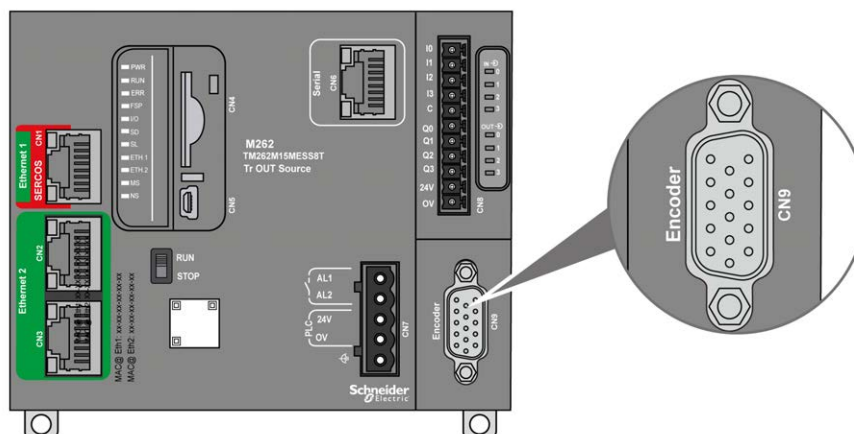
⚠ WARNING
UNINTENDED EQUIPMENT OPERATION
Ensure that the physical wiring respects the connections indicated in the wiring diagram.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

Encoder Interface

Encoder Interface

Overview

The following illustration shows the encoder interface on TM262M• references:



The encoder interface supports the following connection types:

- Incremental (RS422 (5 V or 24 V))
- Absolute (SSI)

The advantage of using an Absolute (SSI) encoder for position detection is that the position of the moving object being monitored is retained. On power-up, or restart following a power interruption, the data provided by the encoder can therefore be used without qualification by the controller.

The encoder interface can provide power to the encoder.

The power supply to the encoder interface is provided by the controller through the embedded digital outputs, page 104 power supply.

NOTE: You must take into account the consumption of the encoder when sizing the power supply for the embedded digital outputs.

Characteristics

The table below shows the characteristics of the encoder:

Characteristics	Description	
Inputs	Rated input voltage	5 Vdc
	Input voltage limits	28.8 Vdc
	Rated input current	1.5 mA @ 5 V 8 mA @ 24 V
	Input impedance	2.85 kΩ
Incremental Encoder	Type of signal	A+, A-, B+, B-, Z+, Z-
	Maximum operating frequency	200 kHz
	Number of bits	32, with configurable frame: - Number of turns - Number of bits/turn - Binary or gray format - Parity
SSI Encoder	Clock frequency	100 KHz, 250 KHz, or 500 KHz (selectable in EcoStruxure Machine Expert)
	Clock voltage	5 Vdc
Power supply to encoder (selectable in EcoStruxure Machine Expert)	None, 5 Vdc, or 24 Vdc:	
	None	No power is supplied to the encoder.
	5 Vdc	Nominal voltage: 5.1 Vdc ± 5 % Max. current: 200 mA Overcurrent and short circuit protection: No Encoder power return: Yes (selectable in EcoStruxure Machine Expert). Typical threshold: 2 V
	24 Vdc	Use a regulated and smoothed power supply on the 24 Vdc power inputs of the CN8 terminal connector, with the specific characteristics of voltage limits and ripple factor specified for the encoder Nominal voltage: 24 Vdc with -0.7 Vdc typical internal voltage drop Max. current: 200 mA Overcurrent and short circuit protection: Yes. Max. current < 1.5 A Encoder power return: Yes (selectable in EcoStruxure Machine Expert). Typical threshold: 9 V
Isolation	Between encoder signals and internal logic	550 Vac for 1 min.
Connector	Type	Removable 15-pin Sub-D HD
	Insertion/removal durability	> 100 times
Cable	Type	Twisted pairs, shielded
	Length	≤ 250 kHz: 100 m (328 ft) max. See Note below. 500 kHz: 50 m (164 ft) max. See Note below.

NOTE: Calculation of Maximum Cable Length

Max. cable length [m] = Max. voltage drop for the cable [V] x Wire cross section (mm²) / (Encoder current [A] x 0.0171 (Ω mm²/m))

where:

Max. voltage drop for the cable = (Min. module output voltage - Min. encoder input voltage) / 2

Example:

Encoder consumes 100 mA with a 4.5...5.5 V supply

Min. module output voltage = 5.1 Vdc x 0.95 = 4.845 Vdc

Max. voltage drop for the cable = (4.845 Vdc - 4.5 Vdc) / 2 = 0.1725 Vdc

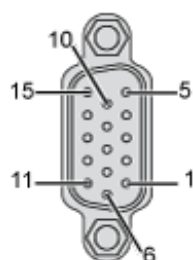
Max. cable length 0.14 mm² = 0.1725 x 0.14 / (0.1 x 0.0171) = 14 m

Max. cable length 0.50 mm² = 0.1725 x 0.50 / (0.1 x 0.0171) = 50 m

Pin Assignment

The encoder interface consists of a 15-pin Sub-D HD connector.

The following illustration describes the pins numbering:

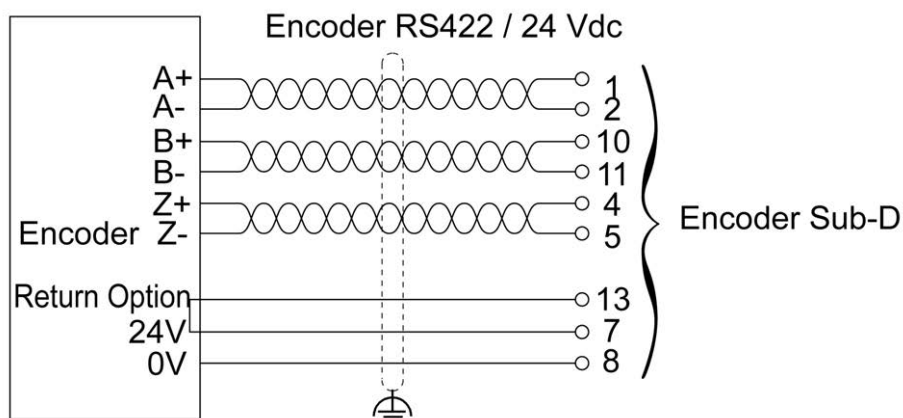


The following table describes the pins of the encoder:

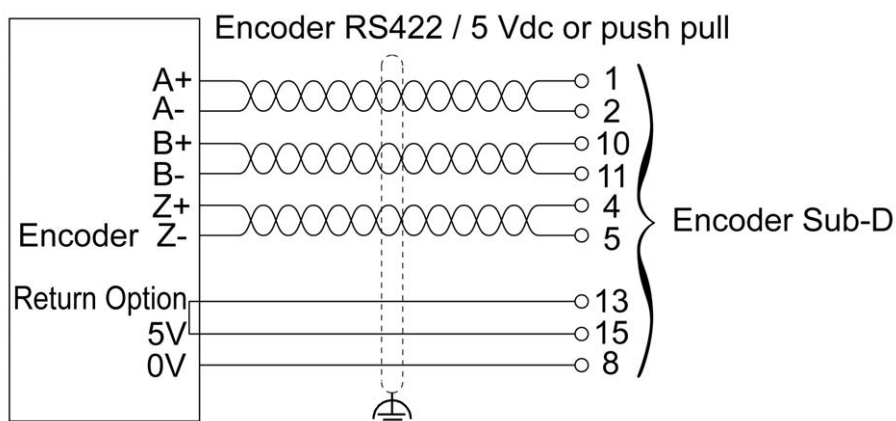
Description	Encoder	Pin	Wire colors
Incremental encoder	A+	1	red/white
	A-	2	brown
	Z+	4	orange
	Z-	5	yellow
	B+	10	white
	B-	11	purple
Absolute (SSI) encoder	SSI data +	1	red/white
	SSI data -	2	brown
	CLKSSI +	6	green
	CLKSSI -	14	light brown
5 V Encoder supply	+ 5 Vdc	15	light purple
	0 Vdc	8	pink
24 V Encoder supply	+ 24 Vdc	7	blue
	0 Vdc	8	pink
Encoder power distribution feedback ⁽¹⁾	Supply return	13	light green
Shielding		Shell	cable braided shield
⁽¹⁾ Detection of encoder supply from controller. Default: Raised if signal is absent.			

Wiring Diagram

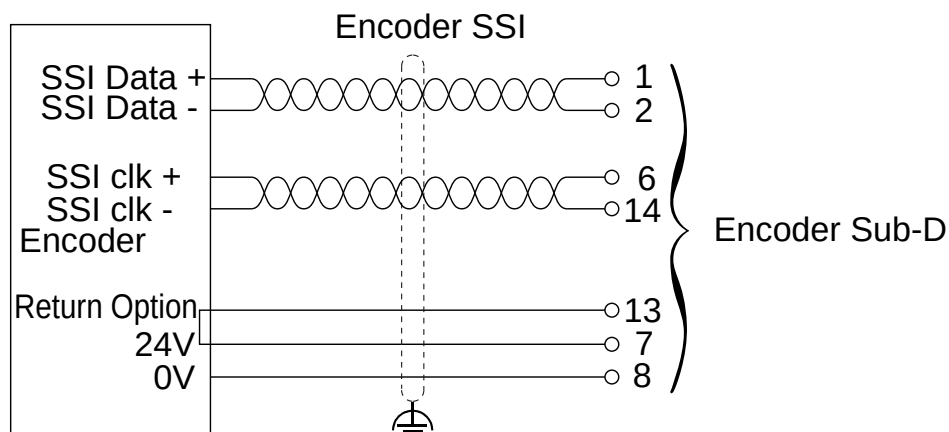
The following illustration describes the wiring diagram of an incremental encoder (RS422 / 24 Vdc) mounted on the encoder interface:



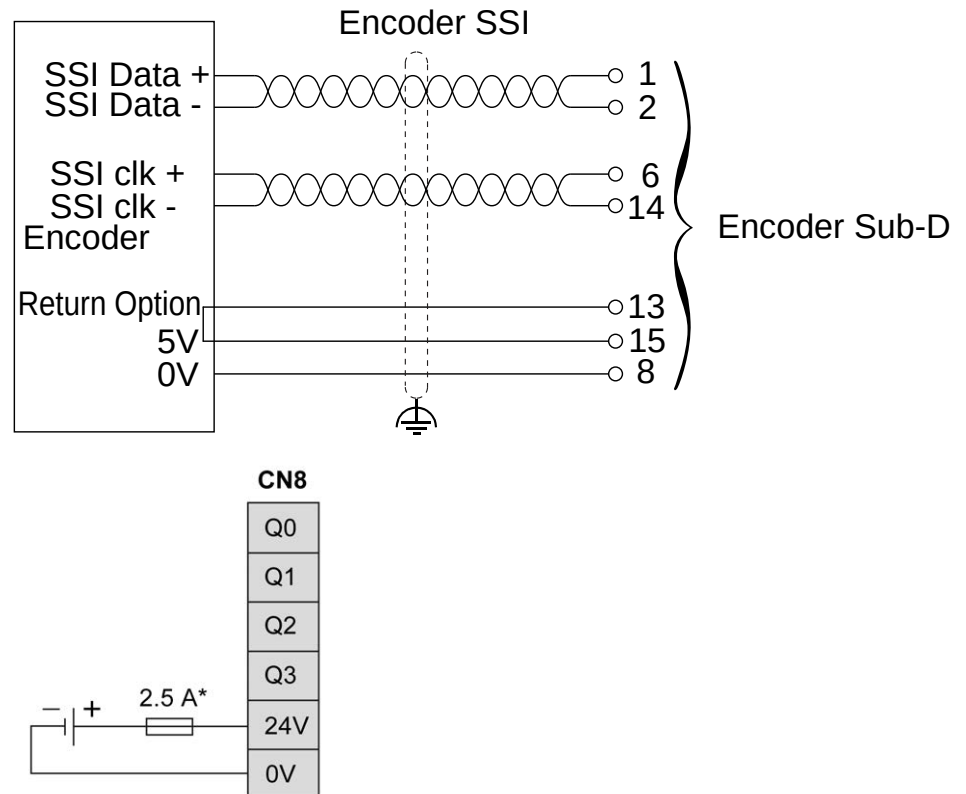
The following illustration describes the wiring diagram of an incremental encoder (RS422 / 5 Vdc or push-pull) mounted on the encoder interface:



The following illustration describes the wiring diagram of an absolute (SSI) encoder (24 Vdc) mounted on the encoder interface:



The following illustration describes the wiring diagram of an absolute (SSI) encoder (5 Vdc) mounted on the encoder interface:



* Use a type T fuse appropriate for the load, not to exceed 2.5 A

Integrated Communication Ports

Ethernet 1 Port

Overview

The M262 Logic/Motion Controller is equipped with Ethernet communications ports:

Port Name	Number of Ports	Reference
Ethernet 1	1 (100BASE-T)	TM262L•
	1 (100BASE-T / SERCOS)	TM262M•
Ethernet 2	2 (dual 1000BASE-T Ethernet switch)	TM262•

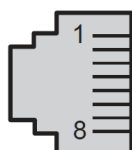
Characteristics

This table describes the physical characteristics of the Ethernet 1 port:

Characteristic	Description
Protocols	Modbus TCP, EtherNet/IP, SERCOS III (on TM262M• references)
Connector type	RJ45
Auto negotiation	From 10 Mbs half duplex to 100 Mbps full duplex
Cable type	Shielded
Automatic cross-over detection	MDI/MDIX

Ethernet 1 Pin Assignment

This figure shows the Ethernet 1 connector pin assignment:



This table describes the Ethernet 1 RJ45 connector pins:

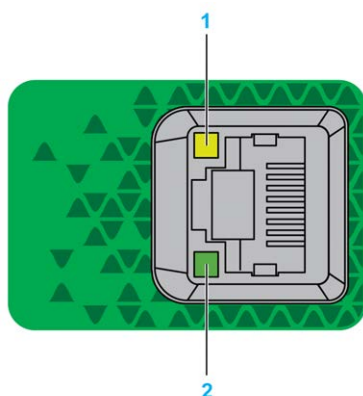
Pin N°	100BASE-T	Description
1	TD+	Transmit data +
2	TD-	Transmit Data -
3	RD+	Receive Data +
4	–	Reserved
5	–	Reserved
6	RD-	Receive Data -
7	–	Reserved
8	–	Reserved

NOTE: The controller supports the MDI/MDIX auto-crossover cable function. It is not necessary to use special Ethernet crossover cables to connect devices directly to this port (connections without an Ethernet hub or switch).

NOTE: Ethernet cable disconnection is detected every second. In case of disconnection of a short duration (< 1 second), the network status may not indicate the disconnection.

Status LED

This figure shows the RJ45 connector status LEDs:

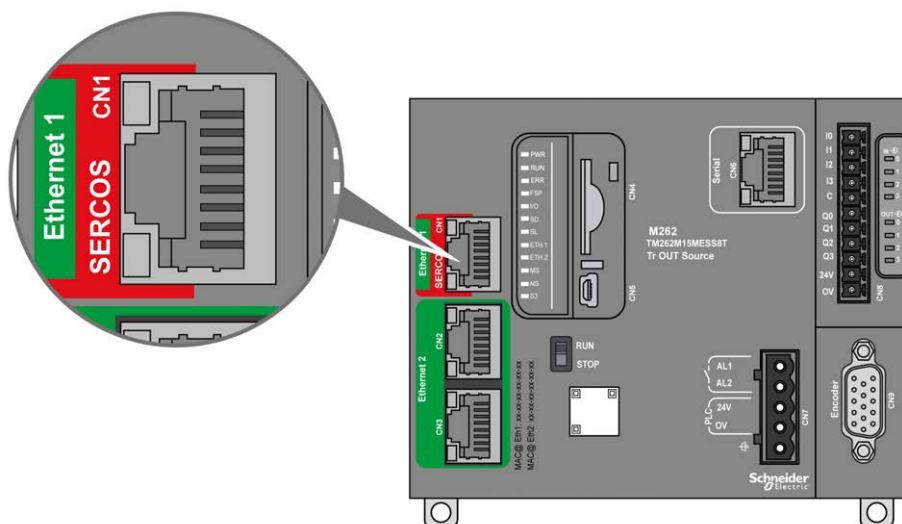


This table describes the Ethernet port status LEDs:

Label	Description	LED		
		Color	Status	Description
1	Ethernet link/speed	Green/Yellow	Off	No link
			Solid yellow	Activity at 10/100 Mbps
			Solid green	Activity at 1000 Mbps
2	Ethernet activity	Green	Off	No activity
			On	The link is detected, but there is no activity
			Flashing	Transmitting or receiving data

Sercos Port

This illustration presents the location of the Sercos port on TM262M• references:

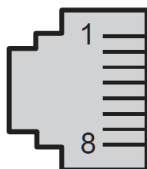


Sercos Port Characteristics

Characteristic	Description
Standard	Sercos III (Master)
Connector type	RJ45
Performances	• TM262M05MESS8T: up to 4 axes synchronized at 1 ms • TM262M15MESS8T: up to 4 axes synchronized at 1 ms • TM262M25MESS8T: ◦ up to 4 axes synchronized at 1 ms ◦ up to 8 axes synchronized at 2 ms • TM262M35MESS8T: ◦ up to 8 axes synchronized at 1 ms ◦ up to 16 axes synchronized at 2 ms ◦ up to 24 axes synchronized at 4ms

Sercos Port Pin Assignment

This illustration presents the pins of the Sercos port:



This table describes the pin assignment of the Sercos port:

Pin	Signal	Description
1	TD+	Transmit data +
2	TD-	Transmit data -
3	RD+	Receive data +
4	-	Reserved
5	-	Reserved
6	RD-	Receive data -
7	-	Reserved
8	-	Reserved

Ethernet 2 Ports

Overview

The M262 Logic/Motion Controller is equipped with Ethernet communications ports:

Port Name	Number of Ports	Reference
Ethernet 1	1 (100BASE-T)	TM262L•
	1 (100BASE-T / SERCOS)	TM262M•
Ethernet 2	2 (dual 1000BASE-T Ethernet switch)	TM262•

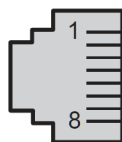
Characteristics

This table describes the physical characteristics of the Ethernet 2 ports:

Characteristic	Description
Protocols	Modbus TCP, EtherNet/IP, Machine Expert (used for data exchange between a PC running EcoStruxure Machine Expert software and the controller, page 122).
Connector type	RJ45
Auto negotiation	From 100 Mbps half duplex to 1000 Mbps full duplex
Cable type	Shielded
Automatic cross-over detection	MDI/MDIX

Ethernet 2 Pin Assignment

This figure shows the Ethernet 2 RJ45 connector pin assignment:



This table describes the Ethernet 2 connector pin assignment:

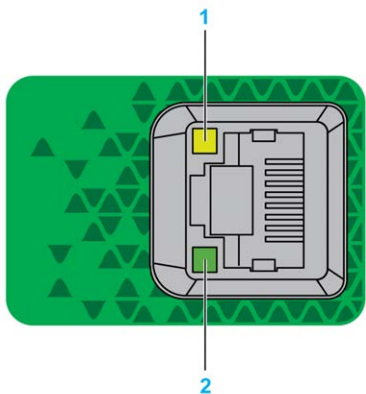
Pin N°	100BASE-T	1000BASE-T
1	TD+	DA+
2	TD-	DA-
3	RD+	DB+
4	–	DC+
5	–	DC-
6	RD-	DB-
7	–	DD+
8	–	DD-

NOTE: The controller supports the MDI/MDIX auto-crossover cable function. It is not necessary to use special Ethernet crossover cables to connect devices directly to this port (connections without an Ethernet hub or switch).

NOTE: Ethernet cable disconnection is detected every second. In case of disconnection of a short duration (< 1 second), the network status may not indicate the disconnection.

Status LEDs

This figure shows the status LEDs on the RJ45 connector:



This table describes the Ethernet port status LEDs:

Label	Description	LED		
		Color	Status	Description
1	Ethernet link/speed	Green/Yellow	Off	No link
			Solid yellow	Activity at 100 Mbps
			Solid green	Activity at 1000 Mbps
2	Ethernet activity	Green	Off	No activity
			On	The link is detected, but there is no activity
			Flashing	Transmitting or receiving data

USB Mini-B Programming Port

Overview

The USB Mini-B Port is the programming port you can use to connect a PC with a USB host port using EcoStruxure Machine Expert software. Using a typical USB cable, this connection is suitable for quick updates of the program or short duration connections to perform maintenance and inspect data values. It is not suitable for long-term connections such as commissioning or monitoring without the use of specially adapted cables to help minimize electromagnetic interference.

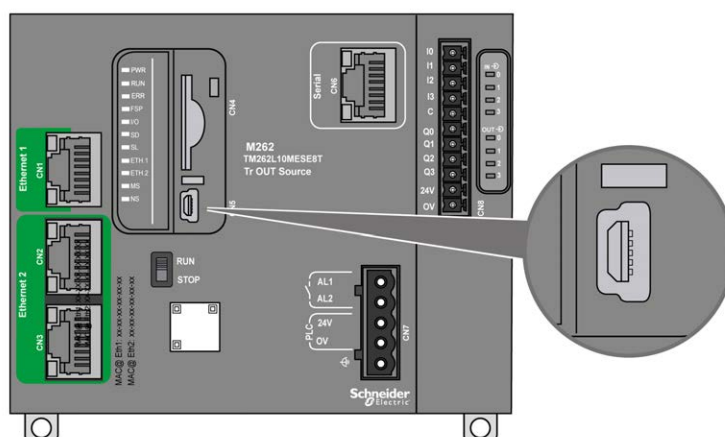
⚠ WARNING

UNINTENDED EQUIPMENT OPERATION OR INOPERABLE EQUIPMENT

- You must use a shielded USB cable such as a BMX XCAUSBH0•• secured to the functional ground (FE) of the system for any long-term connection.
- Do not connect more than one controller or bus coupler at a time using USB connections.
- Do not use the USB port(s), if so equipped, unless the location is known to be non-hazardous.

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

The following figure shows the location of the USB Mini-B programming port:



Characteristics

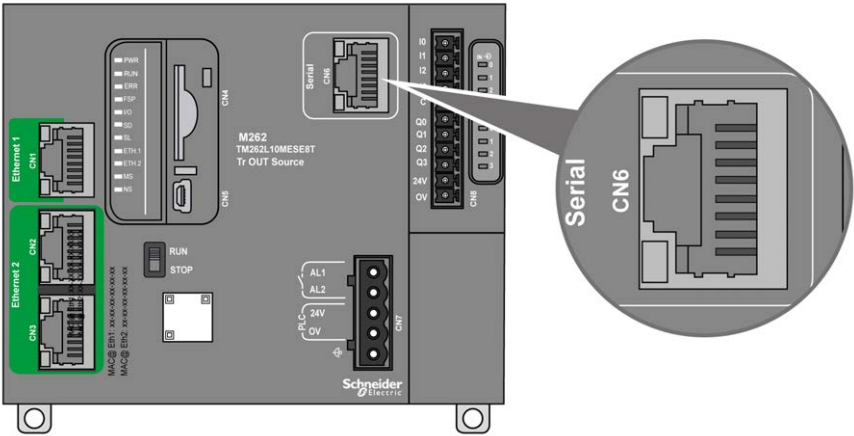
This table describes the characteristics of the USB Mini-B programming port:

Parameter	USB Programming Port
Function	Compatible with USB 2.0
Connector type	Mini-B
Isolation	550 Vac for 1 minute
Cable type	Shielded
Max. Baud Rate	12 Mbits/sec
Max. cable length	5 m (16.5 ft)
Supported protocols	Machine Expert Protocol FTP HTTP Modbus

Serial Line

Overview

The serial line can be used to communicate with devices supporting the Modbus protocol as either master or slave, ASCII protocol (printer, modem...) and Machine Expert Protocol (HMI,...).



Characteristics

Characteristic		Description
Function		RS485 or RS232 software configured
Connector type		RJ45
Isolation		550 Vac
Baud rate		300...115200 bps
Cable	Type	Shielded
	Maximum length (between the controller and an isolated junction box)	30 m (98.43 ft) for RS485 15 m (49.21 ft) for RS232
Polarization		Software configuration is used to connect 576 Ω polarization resistors when the node is configured as a master.

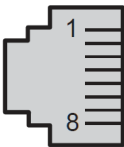
NOTE: Some devices provide voltage on RS485 serial connections. Do not connect these voltage lines to your controller as they may damage the controller serial port electronics and render the serial port inoperable.

NOTICE

INOPERABLE EQUIPMENT
Use only the VW3A8306R•• serial cable to connect RS485 devices to your controller.
Failure to follow these instructions can result in equipment damage.

Pin Assignment

The following figure shows the pins of the RJ45 connector:



This table describes the pin assignment of the RJ45 connector:

Pin	RS232	RS485
1	RxD	N.C.
2	TxD	N.C.
3	N.C.	N.C.
4	N.C.	D1
5	N.C.	D0
6	N.C.	N.C.
7	N.C.	N.C.
8	Common	Common
N.C.: No connection		

WARNING

UNINTENDED EQUIPMENT OPERATION

Do not connect wires to unused terminals and/or terminals indicated as “No Connection (N.C.)”.

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

Status LED

See the description of the **SL** status LED, page 97.

Connecting the M262 Logic/Motion Controller to a PC

Connecting the Controller to a PC

Overview

To transfer, run, and monitor the applications, you can use either a USB cable or an Ethernet connection to connect the controller to a computer with EcoStruxure Machine Expert installed.

NOTICE

INOPERABLE EQUIPMENT

Always connect the communication cable to the PC before connecting it to the controller.

Failure to follow these instructions can result in equipment damage.

USB Mini-B Port Connection

Cable Reference	Details
BMXXCAUSBH018:	Grounded and shielded, this USB cable is suitable for long duration connections.
TCSXCNAMUM3P:	This USB cable is suitable for short duration connections such as quick updates or retrieving data values.

NOTE: You can only connect 1 controller or any other device associated with EcoStruxure Machine Expert and its component to the PC at any one time.

The USB Mini-B Port is the programming port you can use to connect a PC with a USB host port using EcoStruxure Machine Expert software. Using a typical USB cable, this connection is suitable for quick updates of the program or short duration connections to perform maintenance and inspect data values. It is not suitable for long-term connections such as commissioning or monitoring without the use of specially adapted cables to help minimize electromagnetic interference.

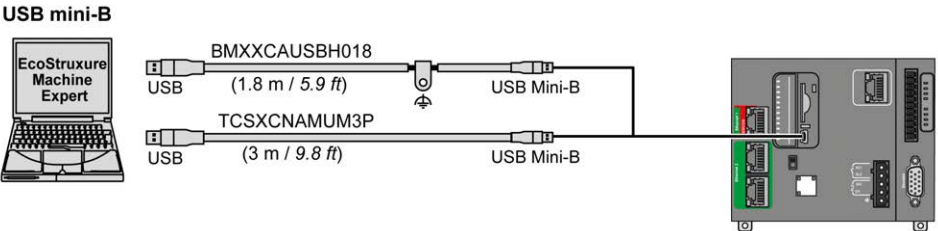
⚠ WARNING

UNINTENDED EQUIPMENT OPERATION OR INOPERABLE EQUIPMENT

- You must use a shielded USB cable such as a BMX XCAUSBH0** secured to the functional ground (FE) of the system for any long-term connection.
- Do not connect more than one controller or bus coupler at a time using USB connections.
- Do not use the USB port(s), if so equipped, unless the location is known to be non-hazardous.

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

The communication cable should be connected to the PC first to minimize the possibility of electrostatic discharge affecting the controller.

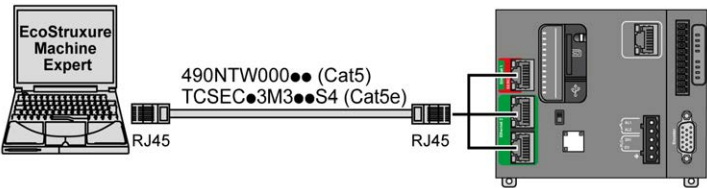


To connect the USB cable to your controller, follow the steps below:

Step	Action
1	1a. If making a long-term connection using the cable BMXXCAUSBH018, or other cable with a ground shield connection, be sure to securely connect the shield connector to the functional ground (FE) or protective ground (PE) of your system before connecting the cable to your controller and your PC. 1b. If making a short-term connection using the cable TCSXCNAMUM3P or other non-grounded USB cable, proceed to step 2.
2	Connect your USB cable to the computer.
3	Open the protective cover for the USB mini-B slot on the controller.
4	Connect the mini-B connector of your USB cable to the controller.

Ethernet Port Connection

You can also connect the controller to a PC using an Ethernet cable.



To connect the controller to the PC, do the following:

Step	Action
1	Connect the Ethernet cable to the PC.
2	Connect the Ethernet cable to any of the Ethernet ports on the controller.

Glossary

A

analog input:

Converts received voltage or current levels into numerical values. You can store and process these values within the logic controller.

application:

A program including configuration data, symbols, and documentation.

ASCII:

(*American standard code for Information Interchange*) A protocol for representing alphanumeric characters (letters, numbers, certain graphics, and control characters).

B

bps:

(*bit per second*) A definition of transmission rate, also given in conjunction with multiplier kilo (kbps) and mega (mbps).

C

CANopen:

An open industry-standard communication protocol and device profile specification (EN 50325-4).

CFC:

(*continuous function chart*) A graphical programming language (an extension of the IEC 61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to the inputs of other blocks to create complex expressions.

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.

continuous function chart language:

A graphical programming language (an extension of the IEC61131-3 standard) based on the function block diagram language that works like a flowchart. However, no networks are used and free positioning of graphic elements is possible, which allows feedback loops. For each block, the inputs are on the left and the outputs on the right. You can link the block outputs to inputs of other blocks to create complex expressions.

controller:

Automates industrial processes (also known as programmable logic controller or programmable controller).

D

DIN:

(*Deutsches Institut für Normung*) A German institution that sets engineering and dimensional standards.

E

EIA rack:

(electronic industries alliance rack) A standardized (EIA 310-D, IEC 60297, and DIN 41494 SC48D) system for mounting various electronic modules in a stack or rack that is 19 inches (482.6 mm) wide.

EN:

EN identifies one of many European standards maintained by CEN (*European Committee for Standardization*), CENELEC (*European Committee for Electrotechnical Standardization*), or ETSI (*European Telecommunications Standards Institute*).

Ethernet:

A physical and data link layer technology for LANs, also known as IEEE 802.3.

F

FBD:

(function block diagram) One of 5 languages for logic or control supported by the standard IEC 61131-3 for control systems. Function block diagram is a graphically oriented programming language. It works with a list of networks, where each network contains a graphical structure of boxes and connection lines, which represents either a logical or arithmetic expression, the call of a function block, a jump, or a return instruction.

FE:

(functional Earth) A common grounding connection to enhance or otherwise allow normal operation of electrically sensitive equipment (also referred to as functional ground in North America).

In contrast to a protective Earth (protective ground), a functional earth connection serves a purpose other than shock protection, and may normally carry current. Examples of devices that use functional earth connections include surge suppressors and electromagnetic interference filters, certain antennas, and measurement instruments.

FreqGen:

(frequency generator) A function that generates a square wave signal with programmable frequency.

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.

H

HE10:

Rectangular connector for electrical signals with frequencies below 3 MHz, complying with IEC 60807-2.

HSC:

(high-speed counter) A function that counts pulses on the controller or on expansion module inputs.

I

I/O:

(input/output)

IEC 61131-3:

Part 3 of a 3-part IEC standard for industrial automation equipment. IEC 61131-3 is concerned with controller programming languages and defines 2 graphical and 2 textual programming language standards. The graphical programming languages are ladder diagram and function block diagram. The textual programming languages include structured text and instruction list.

IEC:

(international electrotechnical commission) A non-profit and non-governmental international standards organization that prepares and publishes international standards for electrical, electronic, and related technologies.

IL:

(instruction list) A program written in the 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 (refer to IEC 61131-3).

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).

IP 20:

(ingress protection) The protection classification according to IEC 60529 offered by an enclosure, shown by the letter IP and 2 digits. The first digit indicates 2 factors: helping protect persons and for equipment. The second digit indicates helping protect against water. IP 20 devices help protect against electric contact of objects larger than 12.5 mm, but not against water.

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).

LD:

(ladder diagram) 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 (refer to IEC 61131-3).

N

NEMA:

(national electrical manufacturers association) The standard for the performance of various classes of electrical enclosures. The NEMA standards cover corrosion resistance, ability to help protect from rain, submersion, and so on. For IEC member countries, the IEC 60529 standard classifies the ingress protection rating for enclosures.

P

PE:

(*Protective Earth*) A common grounding connection to help avoid the hazard of electric shock by keeping any exposed conductive surface of a device at earth potential. To avoid possible voltage drop, no current is allowed to flow in this conductor (also referred to as *protective ground* in North America or as an equipment grounding conductor in the US national electrical code).

program:

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

PTO:

(*pulse train outputs*) A fast output that oscillates between off and on in a fixed 50-50 duty cycle, producing a square wave form. PTO is especially well suited for applications such as stepper motors, frequency converters, and servo motor control, among others.

PWM:

(*pulse width modulation*) A fast output that oscillates between off and on in an adjustable duty cycle, producing a rectangular wave form (though you can adjust it to produce a square wave).

R

RJ45:

A standard type of 8-pin connector for network cables defined for Ethernet.

RS-232:

A standard type of serial communication bus, based on 3 wires (also known as EIA RS-232C or V.24).

RS-485:

A standard type of serial communication bus, based on 2 wires (also known as EIA RS-485).

RTC:

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

RxD:

The line that receives data from one source to another.

S

SFC:

(*sequential function chart*) A language that is composed of steps with associated actions, transitions with associated logic condition, and directed links between steps and transitions. (The SFC standard is defined in IEC 848. It is IEC 61131-3 compliant.)

SSI:

(*serial synchronous interface*) A common interface for relative and absolute measurement systems like encoders.

ST:

(*structured text*) A language that includes complex statements and nested instructions (such as iteration loops, conditional executions, or functions). ST is compliant with IEC 61131-3.

T

terminal block:

(*terminal block*) The component that mounts in an electronic module and provides electrical connections between the controller and the field devices.

TxD:

The line that sends data from one source to another.

Index

A

accessories 28

B

bus coupler
specifications 26

C

certifications and standards 41
communication ports
Ethernet ports 114, 116
Serial Line 119
Communication Ports 114
USB Programming Port 118

E

Electrical Requirements
Installation 52
Electromagnetic Susceptibility 41
Environmental Characteristics 39
expansion modules
TMS 28

F

fallback
configuring modes 32
features
key features 12
fieldbus interface
specifications 26

G

Grounding 58

I

inductive load, output protection
output protection, inductive load 54
Input Management 30
installation
logic/motion controller installation 42
Installation 39
Electrical Requirements 52
intended use 6

L

Latching 31
logic/motion controller installation 42

M

Machine Expert Protocol 119
mounting positions 43

O

output management 32

P

pin assignment
encoder interface 111
Sercos 116
Power Supply 56
presentation
TM262L01MESE8T 66
TM262L10MESE8T 70
TM262L20MESE8T 75
TM262M05MESS8T 80
TM262M15MESS8T 85
TM262M25MESS8T 90
TM262M35MESS8T 95
programming languages
IL, LD, Grafcet 12

Q

qualification of personnel 5

R

real time clock 30
Run/Stop 32

S

SD Card 33
Sercos port 115
Serial Line
communication ports 119
short-circuit or over-current on transistor outputs 32

T

TMS expansion modules 28

U

USB Programming Port
Communication Ports 118

W

weight
TM262L01MESE8T 70
TM262L10MESE8T 75
TM262L20MESE8T 80
TM262M05MESS8T 85
TM262M15MESS8T 90
TM262M25MESS8T 95
TM262M35MESS8T 100
wiring 52

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time,
please ask for confirmation of the information given in this publication.

© 2022 Schneider Electric. All rights reserved.

EIO0000003659.08

Modicon M262

Embedded Safety

Integration Guide

10/2019

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

© 2019 Schneider Electric. All rights reserved.

Table of Contents



	Safety Information	5
	About the Book	9
Chapter 1	Introduction	15
	Logic/Motion Controller Application with Embedded Safety - System Overview	16
	Compatibility and Limitations	18
	Achievable Safety Integrity Level / Performance Level	20
Chapter 2	Installation	21
	Mechanical Installation	22
	Electrical Installation	23
	Fieldbus and Commissioning PC Wiring	25
	Software Installation	27
	Firmware Updates	29
Chapter 3	Application Setup in the Software Tools	31
3.1	Setting Up the Standard Application in EcoStruxure Machine Expert	32
	Creating and Adapting a Project in EcoStruxure Machine Expert	33
	Configuring the Logic/Motion Controller	35
	Configuring the Sercos Addresses and IP Address Assignment	37
	Configuring the Safety Logic Controller	38
	Configuring the Safety-related TM5/TM7 Modules	41
	Configuring the TM5NS31 Bus Coupler	42
	Commissioning the Logic/Motion Controller - Part 1	44
3.2	Setting Up the Safety Application	46
	First Steps in Machine Expert - Safety	47
	Devices Window in Machine Expert - Safety	48
	Configuring the Safety Logic Controller	49
	Configuring Safety-related TM5/TM7 Module Parameters	54
	Calculating the Safety Response Time	55
	Programming the Safety-related Application	58
	Commissioning the Safety-related Application	65
	Validation and Documentation of the Safety-Related Project	73

3.3	Interaction Between the Safety Application and the Standard Application	75
	Exchanging Data Between Logic/Motion Controller and Safety Logic Controller	76
	Enabling a Safety-related Output via the Standard Application	78
	Reading Diagnostic Signals of Safety-related Modules	80
	Downloading Modified Projects to Logic/Motion Controller and SLC . .	82
Chapter 4	Operation and Maintenance of the Integrated Application	83
	System Start-up	84
	Monitoring the Safety Application in Logic Builder	85
	Remote Controlling of the SLC	88
	Sercos Diagnostics:	89
Index		91

Safety Information



Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result in** minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

BEFORE YOU BEGIN

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

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

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

START-UP AND TEST

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check be made and that enough time is allowed to perform complete and satisfactory testing.

WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book



At a Glance

Document Scope

This document describes the steps to integrate a Safety Logic Controller (SLC) with connected safety-related TM5/TM7 I/O devices into a Logic/Motion Controller environment in EcoStruxure Machine Expert and Machine Expert - Safety.

You will learn how to

- setup the bus architecture
- configure the devices involved
- configure the standard and safety-related parameters of the devices
- setup and create a safety-related project
- commission, operate and maintain the application
- perform system diagnostics
- exchange data between the Logic/Motion Controller and the Safety Logic Controller (SLC)

The document is to be understood as a general guideline which focuses on the integration of Embedded Safety into the Logic/Motion Controller application. For detailed information on device-specific characteristics and procedures, refer to the respective user guide(s).

Validity Note

This document has been updated with the release of EcoStruxure Machine Expert V1.2.

For product compliance and environmental information (RoHS, REACH, PEP, EOL, etc.), go to www.schneider-electric.com/green-premium.

The technical characteristics of the devices described in the present document also appear online. To access the information online, go to the Schneider Electric homepage www.se.com.

The characteristics that are presented in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Product Related Information

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

For the Safety Logic Controllers:

DANGER

POTENTIAL FOR EXPLOSION

- Only use this equipment in non-hazardous locations, or in locations that comply with Class I, Division 2, Groups A, B, C and D.
- Do not substitute components which would impair compliance to Class I, Division 2.
- Do not connect or disconnect equipment unless power has been removed or the location is known to be non-hazardous.
- Do not use the USB port(s), if so equipped, unless the location is known to be non-hazardous.

Failure to follow these instructions will result in death or serious injury.

For the Logic/Motion Controllers:

This equipment has been designed to operate outside of any hazardous location. Only install this equipment in zones known to be free of a hazardous atmosphere.

DANGER

POTENTIAL FOR EXPLOSION

Install and use this equipment in non-hazardous locations only.

Failure to follow these instructions will result in death or serious injury.

WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POU's found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

WARNING

IMPROPER USE OF PROGRAM ORGANIZATION UNITS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the Program Organization Units (POUs) are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POU's.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Care must be taken and provisions made for use of this library for machine control to avoid inadvertent consequences of commanded machine operation, state changes, or alteration of data memory or machine operating elements.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Place operator devices of the control system near the machine or in a place where you have full view of the machine.
- Protect operator commands against unauthorized access.
- If remote control is a necessary design aspect of the application, ensure that there is a local, competent, and qualified observer present when operating from a remote location.
- Configure and install the Run/Stop input, if so equipped, or, other external means within the application, so that local control over the starting or stopping of the device can be maintained regardless of the remote commands sent to it.

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

Related Documents

Document title	Reference
Modicon M262 Logic/Motion Controller - Hardware Guide	EIO0000003659 (ENG); EIO0000003660 (FRE); EIO0000003661 (GER); EIO0000003662 (ITA); EIO0000003663 (SPA); EIO0000003664 (CHS); EIO0000003665 (POR); EIO0000003666 (TUR)
Safety Logic Controller 100/200 FS - Hardware Guide	EIO0000000889 (ENG); EIO0000000891 (GER);

Document title	Reference
EcoStruxure Machine Expert Programming Guide	EIO0000000067 (ENG); EIO0000000069 (FRE); EIO0000000068 (GER); EIO0000000070 (ITA); EIO0000000071 (SPA); EIO0000000072 (CHS)
M262 Logic/Motion Controller - Programming Guide	EIO0000003651 (ENG); EIO0000003652 (FRE); EIO0000003653 (GER); EIO0000003654 (ITA); EIO0000003655 (SPA); EIO0000003656 (CHS); EIO0000003657 (POR); EIO0000003658 (TUR)
TM5 Sercos III Interface - Hardware Guide	EIO0000003221 (ENG); EIO0000003222 (FRE); EIO0000003223 (GER); EIO0000003225 (ITA); EIO0000003224 (SPA); EIO0000003226 (CHS)
EcoStruxure Machine Expert - Safety - User Guide	EIO0000002147 (ENG); EIO0000002338 (GER)
Safety Modules - Reference Guide	EIO0000002265 (ENG); EIO0000002266 (GER)
SafeLogger for EcoStruxure Machine Expert - Safety	EIO0000002596 (ENG); EIO0000002597 (GER);

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.

Standard	Description
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive (2006/42/EC)* and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Chapter 1

Introduction

What Is in This Chapter?

This chapter contains the following topics:

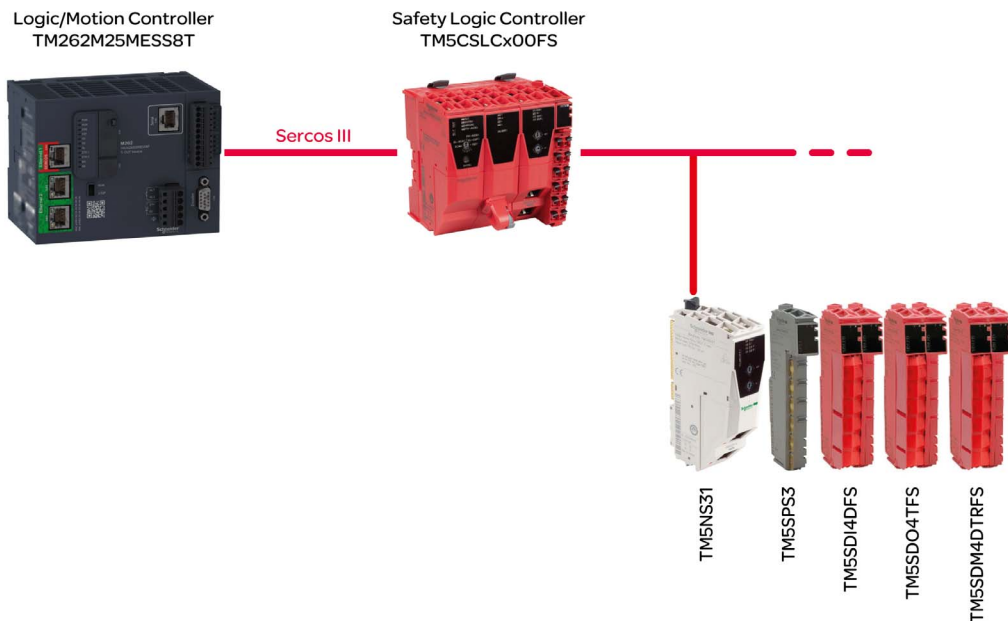
Topic	Page
Logic/Motion Controller Application with Embedded Safety - System Overview	16
Compatibility and Limitations	18
Achievable Safety Integrity Level / Performance Level	20

Logic/Motion Controller Application with Embedded Safety - System Overview

Architecture

This documentation describes the steps to integrate safety components (embedded safety) by using a Safety Logic Controller (SLC) and safety-related TM5/TM7 modules into a Logic/Motion Controller application via the Sercos III bus.

The following figure illustrates a small application setup which is used for explanation purposes in this guide.



NOTE: Observe the specific limitations for the Logic/Motion Controller used. Refer to the chapter *System Limitations* ([see page 18](#)) for details on the supported system architecture and the maximum number of connectable Sercos slaves and safety-related TM5/TM7 I/O modules.

NOTE: Term definition: Standard = non-safety-related. The term "standard" always refers to non-safety-related items/objects. Examples: a standard process data item is only read/written by a non-safety-related I/O device, that is, a standard device. Standard variables/functions/FBs are non-safety-related data. The term "standard controller" designates the non-safety-related Logic/Motion Controller.

Devices Involved

The following devices are used in the explanatory sample project of this documentation:

- TM262M25MESS8T Logic/Motion Controller (see chapter *Compatible Logic/Motion Controller Types* (see page 18))
- TM5CSLCx00FS Safety Logic Controller
- TM5NS31 TM5 SERCOS III Bus Coupler
- TM5SPS3 Power Supply Module
- TM5SDI4DFS Digital Input Safety Module
- TM5SDO4TFS Digital Output Safety Module
- TM5SDM4DTRFS Digital Mixed Safety Module

The safety-related part of the architecture is composed of safety nodes (SN). A SN is a node within the Sercos network which complies to the openSafety protocol. Safety-related modules by Schneider Electric are red. They can be identified by the appendix FS in their order number.

A typical application setup in practice may contain further Sercos slaves (such as standard drive modules), as well as more than one TM5 bus coupler connected to the Sercos bus and a higher number of TM5 and/or TM7 I/O modules. However, only one SLC can be used under the Sercos Master (which is the Sercos I/O controller inside the Logic/Motion Controller).

The Logic/Motion Controller executes the (non-safety-related) standard control application. The SLC as safety-related controller is subordinate to the Logic/Motion Controller. It manages the tasks within a safety-related application thus executing a separate safety-related application program.

Software Tools Involved

For embedding safety as described in this documentation, EcoStruxure Machine Expert with the software components Modicon and EcoStruxure Machine Expert - Safety is used. (Also refer to the chapter *Software Installation* (see page 27).)

EcoStruxure Machine Expert Logic Builder is used to:

- Configure the bus architecture consisting of standard and safety-related devices. (The safety-related devices must additionally be confirmed in Machine Expert - Safety.)
- Parameterize the standard devices and partially the safety-related devices.
- Develop the standard application program.
- Commission, control, monitor, and debug the Logic/Motion Controller.
- Perform system diagnostics, for example in online editors or using the SafeLogger.

EcoStruxure Machine Expert - Safety is used to:

- Define the safety-related parameters of the safety-related devices (SLC and safety-related I/O modules).
- Calculate the safety response time based on the defined response time relevant parameters.
- Develop the safety-related application program.
- Commission, control, monitor, and debug the SLC.
- Document the safety-related project.

The engineering tasks listed above are described in detail in the following chapters.

Compatibility and Limitations

Compatible Logic/Motion Controller Types

The following Logic/Motion Controller types with the type labels TM262Mxxx provide an Ethernet port with Sercos interface. Therefore, they support the integration of Embedded Safety by means of an SLC connected to the Sercos bus:

- TM262M15MESS8T
- TM262M25MESS8T
- TM262M35MESS8T

The types TM262Lxxx do not provide an Ethernet port for connecting a Sercos interface and therefore cannot be used in an application as described here.

NOTE: Observe the specific limitations for the Logic/Motion Controller used. Refer to the chapter *System Limitations* ([see page 18](#)) for details on the supported system architecture and the maximum number of connectable Sercos slaves and safety-related TM5/TM7 I/O modules.

Project Compatibility

In general, EcoStruxure Machine Expert projects are compatible for the different Modicon M262 Logic/Motion Controller types and can therefore be transferred as long as Sercos-related requirements and constraints are considered.

Safety-related Machine Expert - Safety projects are compatible for the Safety Logic Controller types TM5CSLC100FS and TM5CSLC200FS if the maximum number of safety-related modules is not exceeded. Refer to the section *System Limitations* ([see page 18](#)) for details.

Modicon M262 Logic/Motion Controller projects are not compatible with PacDrive 3 projects and cannot be transferred between these systems. It is, however, possible to transfer the safety-related project part via export/import. Refer to the chapter *Importing/Exporting Projects* of the *EcoStruxure Machine Expert - Safety User Guide* for details.

NOTE: The migration of a safety-related project from a PacDrive 3 system implies a following validation/re-certification of the complete safety function.

System Limitations

For Logic/Motion Controller systems with Embedded Safety as described in this document, the following limitations apply:

- One SLC is allowed per Logic/Motion Controller.
- Depending on the Logic/Motion Controller type and the Sercos cycle time, up to 40 Sercos slaves are supported. When embedding safety as described here, the SLC is also considered as one slave.
- The Logic/Motion Controller types TM262Mx5x support a maximum number of 30 safety-related TM5/TM7 I/O modules. There is no restriction how these modules are distributed to the TM5 bus couplers available in your application.

- The TM5CSLC100FS supports up to 20 safety modules connected to the bus coupler. The TM5CSLC200FS allows 30 safety modules (in combination with the Modicon M262 Logic/Motion Controller).
- A maximum number of 63 modules are supported per TM5NS31 bus coupler. Example: 30 safety-related modules (depending on the SLC type) and 33 standard modules.

Refer to *M262 Logic/Motion Controller - Programming Guide* for more information and other system limitations.

Achievable Safety Integrity Level / Performance Level

The IEC 61508, which describes the functional safety of electrical, electronic, or programmable electronic safety-related systems, specifies four levels of safety performance for a safety-related function.

These are called Safety Integrity Levels (SIL), ranging from 1 (the lowest) to 4 (the highest). The TM5CSLC100FS and TM5CSLC200FS Safety Logic Controllers are certified for use in SIL 3 applications. In SIL 3, the de-energized state is the defined safe state.

The EN ISO 13849 standard defines Performance Levels (PL) for the risk assessment. The maximum achievable level for a safety-related application with the SLC and the safety-related TM5/TM7 modules is PL e.

For details on the Functional Safety Certification as well as the classification of Schneider Electric devices and products, refer to the hardware guides of the TM5CSLCx00FS and the safety-related TM5/TM7 modules.

Chapter 2

Installation

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Mechanical Installation	22
Electrical Installation	23
Fieldbus and Commissioning PC Wiring	25
Software Installation	27
Firmware Updates	29

Mechanical Installation

General Information

Observe the enclosure requirements, environmental characteristics, and operating conditions of the devices involved.

For details, refer to the respective User Manuals and Installation Guides listed in the chapter *Related Documents* ([see page 12](#)).

NOTE: Both, the Logic/Motion Controller, and the Safety Logic Controller can be mounted horizontally and vertically. Different environmental requirements may apply to the mounting position.

The devices can be arranged in any way.

NOTE: Observe the specific limitations for the Logic/Motion Controller used. Refer to the chapter *System Limitations* ([see page 18](#)) for details on the supported system architecture and the maximum number of connectable Sercos slaves and safety-related TM5/TM7 I/O modules.

Electrical Installation

General Information

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.¹
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

¹ For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Wiring Guidelines

The following rules must be applied when wiring the devices in a Logic/Motion Controller system:

- Communication wiring must be kept separate from the power wiring. Route these 2 types of wiring in separate cable ducting.
- Verify that the operating conditions and environment are within the specification values.
- Use proper wire sizes to meet voltage and current requirements.
- Use copper conductors (required).
- Use twisted pair shielded cables for encoder, networks, and serial communication connections.

Use shielded, properly grounded cables for all communication connections. If you do not use shielded cable for these connections, electromagnetic interference can cause signal degradation. Degraded signals can cause the controller or attached modules and equipment to perform in an unintended manner.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Use shielded cables for all communication signals.
- Ground cable shields for all communication signals at a single point¹.
- Route communication separately from power cables.

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

¹Multipoint grounding is permissible if connections are made to an equipotential ground plane dimensioned to help avoid cable shield damage in the event of power system short-circuit currents.

NOTE: To conform to IEC 61010 standards, route primary wiring (wires connected to power mains) separately and apart from secondary wiring (extra low voltage wiring coming from intervening power sources). If that is not possible, double insulation is required such as conduit or cable gains.

Furthermore, you have to observe the following rules which are detailed in the respective device manuals and Installation Guides:

- Rules for Spring Terminal Blocks.
- Rules for Screw Terminal Blocks.
- Protecting Outputs from Inductive Load Damage.
- DC Power Supply Requirements, Characteristics, and Wiring.
- Grounding the Logic/Motion Controller System.
- Alarm Relay Wiring.

Fieldbus and Commissioning PC Wiring

Wiring of the Commissioning PC

Connect the commissioning PC as follows:

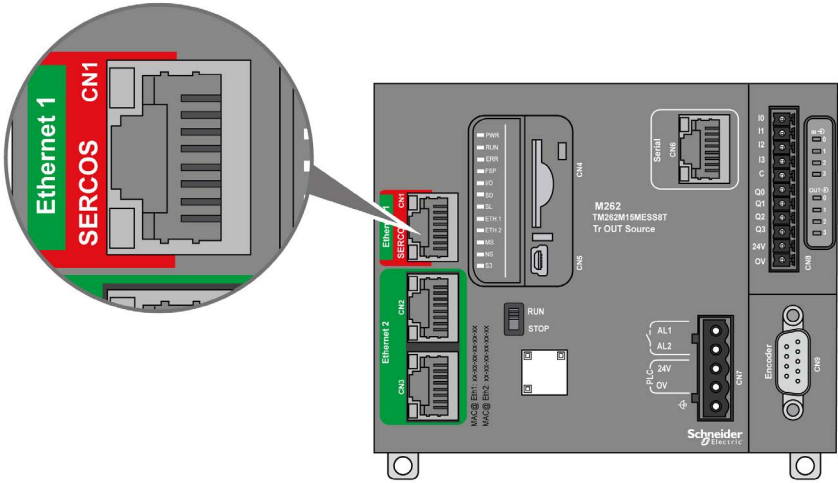
Step	Action
1	Connect the Ethernet cable to the network adapter of your commissioning PC and to one of the Ethernet 2 ports of the Logic/Motion Controller. The ports are labeled with CN2 and CN3 on the device.

Wiring of the Sercos Bus

NOTE: The Sercos bus can only be operated in a single-line architecture.

The following wiring of the Sercos bus relates to the example project described in this guide. The Sercos slaves can alternatively be connected in any other order.

Connect the devices as follows:

Step	Action
1	Connect the Sercos cable to the Ethernet 1 port of the Logic/Motion Controller. The port is labeled with SERCOS CN1 on the device. 
2	At the SLC, connect this cable to any of the Sercos III RJ45 ports.

Step	Action
3	Continue the Sercos bus by plugging a second cable to the free Sercos III RJ45 port of the SLC, and connect it to any of the Sercos ports of the TM5NS31 bus coupler.
4	If required, continue the Sercos bus to the next TM5NS31 bus coupler or any other Sercos slave (drives, and so on) used.

The TM5NS31 bus coupler detects the connected TM5/TM7 I/O devices automatically and creates a corresponding local process image of the given hardware configuration. Therefore, no further signal wiring is required.

Software Installation

Installing EcoStruxure Machine Expert with Required Add-Ons

For further information, refer to the *Machine Expert Installation Guide*.

NOTE: EcoStruxure Machine Expert - Safety and SoSafe Programmable V2.x can be installed in parallel.

Step	Action
1	Launch the Machine Expert Installer tool from your Windows Desktop or Start menu.
2	Select the option Install new software .
3	Select the Source from where the installation of the new components is to be run.
4	Select a Version from the drop-down list.
5	Click Next .
6	In the following screen, select the required components to be installed: ● Program Machine controllers (Modicon) Result: This package includes the support of the Logic/Motion Controller in EcoStruxure Machine Expert. ● Program Machine Safety Result: This installs the required EcoStruxure Machine Expert - Safety software for programming the Safety Logic Controller and parameterizing the safety-related devices. The mentioned components are the minimum requirement for the subject of this integration guide. Install further components as required.
7	Click Start Installation. Result: A time-limited trial version of EcoStruxure Machine Expert is installed. NOTE: When starting Machine Expert - Safety the first time, you have to register once in order to activate a trial version license.

Tools for Firmware Updates

The firmware update of the Logic/Motion Controller must be done using the EcoStruxure Machine Expert Controller Assistant software. To update the Safety Logic Controller and other (TM5/TM7) devices, you need the Device Assistant software. Install these tools as described in the *Machine Expert Installation Guide*, if required.

Adapting Your EcoStruxure Machine Expert Installation

If EcoStruxure Machine Expert is already installed on your PC without the safety add-on or without M262 support, adapt the existing installation using the Machine Expert Installer. Add the components **Program Machine controllers (Modicon)** and **Program Machine Safety**.

Proceed as described in the *Machine Expert Installation Guide*, chapter *Modify Installed Software*.

Activating EcoStruxure Machine Expert

Proceed as follows to activate the software permanently:

Step	Action
1	Launch the License Manager tool from your Windows Desktop or Start menu.
2	Click Activate .
3	Enter the Activation ID , click Next , and follow the instructions given in the dialog. Result: After the successful activation, the Expiration Date of the software is set to permanent in the License Manager .

Firmware Updates

General Information

The firmware of the Logic/Motion Controller, the Safety Logic Controller, and the other devices involved must correspond to the version required by the installed EcoStruxure Machine Expert version.

NOTE: For the Logic/Motion Controller, an incompatible controller firmware version is reported when connecting it for the first time.

Refer to the *Release Notes* for information on compatible software/firmware versions. You can open the Release notes via the Machine Expert Installer software.

Firmware Update of the Logic/Motion Controller

The determination of the installed firmware version and the update of the controller firmware must be done using the EcoStruxure Machine Expert Controller Assistant software.

Proceed as described in the chapter *Updating Firmware* of the *M262 Logic/Motion Controller Programming Guide*.

Firmware Update of the SLC and TM5/TM7 Devices

The determination of the installed firmware version and the update of the Safety Logic Controller firmware, the TM5 bus coupler as well as any TM5/TM7 devices must be done using the Device Assistant software.

Firmware installation packages (*.sefirmware) for the SLC and the TM5 bus couplers are stored on your PC under the full path C:\ProgramData\EcoStruxure Machine Expert\FirmwareRepository\IO.

If the communication via the Sercos bus is already established, the installed firmware version of the Sercos slaves (for example, SLC and TM5 bus coupler) can also be determined via the “electronic type label” The label is displayed in the device parameters in Logic Builder. (Double-click the device in the **Devices tree**, open the **Sercos III Parameters** or **Device Parameters** editor, and expand the **Electronic label** parameter group. The SoftwareRevision parameter indicates the version.)

To obtain TM5/TM7 module firmware packages, contact your Schneider Electric representative.

Proceed as described in the chapter *Brief Step-By-Step Instruction for Firmware Update* in the *Device Assistant User Guide*.

NOTE: If the SLC firmware update is unsuccessful (corrupt file, interruption of the update, and so on), the bus interface restarts with the previous version of the firmware. In this case, format the memory key as described in the chapter Safety Logic Controller Memory Key in the *Safety Logic Controller SLC100/200 FS Hardware Guide*.

Confirming the Safety Module Firmware Update

After you have updated the firmware of a safety-related TM5/TM7 module, you must acknowledge the changed device configuration directly at the SLC (confirm the firmware update). The orange **FW-ACKN** LED at the device indicates the status of the update operation:

LED is...	Meaning	Action
Off	Valid firmware configuration	-
Flashing	Firmware update successful	Set the selection switch to position FW_ACKN and press the confirmation button (ENTER).

Chapter 3

Application Setup in the Software Tools

What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
3.1	Setting Up the Standard Application in EcoStruxure Machine Expert	32
3.2	Setting Up the Safety Application	46
3.3	Interaction Between the Safety Application and the Standard Application	75

Section 3.1

Setting Up the Standard Application in EcoStruxure Machine Expert

What Is in This Section?

This section contains the following topics:

Topic	Page
Creating and Adapting a Project in EcoStruxure Machine Expert	33
Configuring the Logic/Motion Controller	35
Configuring the Sercos Addresses and IP Address Assignment	37
Configuring the Safety Logic Controller	38
Configuring the Safety-related TM5/TM7 Modules	41
Configuring the TM5NS31 Bus Coupler	42
Commissioning the Logic/Motion Controller - Part 1	44

Creating and Adapting a Project in EcoStruxure Machine Expert

Creating a Project Based on an Example

This documentation is based on the example project **SLC Remote Controller (M262)** which is available in EcoStruxure Machine Expert Logic Builder. In the example, devices, tasks, and code (including a visualization) are pre-configured and can be adapted, if required.

Alternatively, you can create your project from scratch based on a **Default Project** and insert the required devices manually.

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	Select File → New Project .
2	In the New Project dialog, click From Example on the left.
3	Select the TM262M25NESS8T Logic/Motion Controller type from the drop-down list. (The example is only available for this controller type. Refer to the note below this table.) Result: The available example projects are listed in the Matching Examples list.
4	Select the SLC Remote Controller (M262) entry in the Matching Examples list.
5	Enter a project name, select the file directory, and click OK . Result: The project is created and the Devices tree as well as the Application tree are prepared as described in the next section.

NOTE: Adapt the controller type if you are using a type other than the TM262M25NESS8T.

Resulting Project Based on the SLC Remote Controller Example

As you have created your project based on the **SLC Remote Controller (M262)** example, the following applies to the project:

- The **Devices tree** window reflects the bus structure as prepared in the project example.
- The **Sercos_Master** is added to the **Ethernet_1** node in the **Devices tree**. This way, the Ethernet 1 port is configured as Sercos port.
- Under the **Sercos_Master**, the TM5CSLCx00FS Safety Logic Controller is available.
- Under the **Sercos_Master**, a TM5NS31 bus coupler is available.
- Under the TM5NS31 bus coupler, the following devices are inserted: TM5SPS3 Power Supply Module, TM5SDI4DFS Digital Input Safety Module, TM5SDO4TFS Digital Output Safety Module, and TM5SDM4TRFS Digital Mixed Safety Module.
- In the **Application tree**, the folder **SlcRemoteControllerExample** is available. The example contains a program POU, an FB POU, and data type definitions. The code in these POUs is ready to compile. It can be used to read the status of the Safety Logic Controller and the connected safety-related modules, and to perform particular configuration-related commands.
- In the **Task Configuration** (in the **Application tree**), a task named **Task_SR_VisControl** is available with a program instance of the prepared program POU. You must adapt the cycle time (*see page 35*) of this task.

- The **Tools tree** contains the folder **SlcRemoteControllerExample** with a visualization of a virtual Safety Logic Controller control center. The HMI page **VIS_SlcRemoteController** enables you to visualize the status of the SLC and the connected safety-related modules, and to perform particular configuration-related commands. Refer to the chapter *Remote Controlling of the SLC* ([see page 88](#)) for further information.
- In the **Feature Configuration** editor of the SLC, the option **PacDriveCompatibility** is selected. This setting is necessary to use the Remote Controller library.

Adapting the Bus Architecture

After you have created the project based on the **SLC Remote Controller (M262)**, you can add further devices involved in your application. For instance, you can add further (safety-related and standard) TM5/TM7 modules at the existing TM5NS31 bus coupler, or further Sercos slaves including TM5NS31 bus couplers.

NOTE: Observe the system limitations ([see page 18](#)).

The steps how to add, remove, replace, and update devices in the **Devices tree** are described in the chapter *Managing Devices* of the EcoStruxure Machine Expert *Programming Guide*.

Configuring the Logic/Motion Controller

Configuring the IP Address of the Logic/Motion Controller

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the controller node.
2	Click the Communication Settings tab. Result: The tab lists the controllers that are accessible by your computer (independently of their communication parameter settings). If the particular Logic/Motion Controller controller is not listed, click the Update icon on the toolbar. Result: Logic Builder scans the network for connected controllers and adds detected devices to the list.
3	Right-click your controller in the list and select Edit Communication Settings from the context menu.
4	Define the communication parameters according to the requirements in your network, select the option Save settings permanently , and click OK .

Adapting the Task Settings

You must adapt the cycle time of the cyclic task TASK_SR_VisControl. (This task is defined in the example you have used to create your project.)

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Application tree , expand the Task Configuration node, and double-click the TASK_SR_VisControl .
2	Set the task Interval to 300 ms.

Configuring the Ethernet 1 Port

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the Ethernet_1 node. Result: The parameter editor opens on the right.
2	Enter the following fixed IP Address configuration: ● IP Address = 172.20.0.1 ● Subnet mask = 255.255.255.0

Defining the Sercos Bus Cycle Time

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the Sercos_Master node. Result: The parameter editor opens on the right.
2	Open the parameter group SercosCycleTimeConfig and enter a CycleTime value of 4 ms. Refer to <i>Best Practice</i> below this table.

Refer to the *Sercos for M262 Logic/Motion Controller User Guide* for details and further information.

Best practice: The suitable cycle time value depends on the number of connected Sercos slaves, I/O modules, and on your application requirements. If a cycle time of 4 ms results in an insufficient system performance of your application, decrease the value.

The Sercos cycle time, however, influences the available asynchronous communication. This has an impact on the start-up performance and the system reliability of the application. There are two possibilities to increase the bandwidth of the asynchronous Sercos channel:

- Increase the Sercos cycle time.
- Stop the Sercos phase-up during the second phase for a few seconds to allow the SLC to finish the parameterization of the SNs.

Configuring the Sercos Addresses and IP Address Assignment

Example-based Settings

As you have created your project based on the **SLC Remote Controller (M262)** example, the relevant communication parameters are already configured as follows. Adapt the values, if required by your application.

Settings in the editor **Schneider Electric Sercos III Parameters** (for the SLC), and **Device Parameters Parameters** (for the bus couplers), parameter group Identification:

Parameter	Meaning
IdentificationMode = Topology mode	The topological address of the Sercos slaves is relevant for the configuration. Alternatively, you can set the parameter for each Sercos slave to Sercos mode, and then set ConfiguredSercosAddress to the value which is specified using the address switches at the device. Refer to the respective hardware guide(s) and the <i>Sercos For Modicon M262 Motion Controller User Guide</i> for details and information regarding the advantages of each mode.
ConfiguredTopologicalAddress = 1 for SLC and 2 for the TM5NS31 bus coupler	The value corresponds to the topological positions of the Sercos slaves in the example project. Adapt these settings if you have connected the devices in another order than described in the chapter <i>Wiring of the Sercos Bus</i> (see page 25).
IPConfigMode = Automatic IP address assignment	The IP address of each Sercos slave is assigned automatically (by the master, starting from the IP address of the Sercos_Master), based on the defined topology address. After the Sercos has reached at least phase 2 (CP2), the assigned device IP addresses are visible in the editor during online mode.

Configuring the Safety Logic Controller

Exchange Data Configuration for the SLC

The standard application (Logic/Motion Controller) and the safety-related SLC application can directly exchange data.

When inserting an SLC into the Logic/Motion Controller project in EcoStruxure Machine Expert, a special memory area for the exchange data is reserved. In this area, exchange signals are available. Which exchange data are to be used in your project must be configured in the I/O configuration of the SLC.

NOTE: The exchange of data between the standard and the safety-related application is always defined as non-safety-related.

When configuring the data exchange, the amount of data is limited. Any detected error is reported when compiling if these limits are exceeded in your configuration.

The total amount of exchange data (in both directions) is limited to 75 points, where

- 8 Bool = 1 point
- 1 INT = 1 point
- 1 UINT = 1 point
- 1 UDINT = 1 point

The maximum number of bytes in direction SLC to Logic/Motion Controller or Logic/Motion Controller to SLC is limited to 128 bytes, where

- 8 Bool = 1 byte
- 1 INT = 2 bytes
- 1 UINT = 2 bytes
- 1 UDINT = 4 bytes

In addition, the amount of each data type is limited (according to the definition in the **Type** column of the **Schneider Electric Sercos III Parameters** editor). The limits are verified by Logic Builder during the configuration process.

Configuring the Data Exchange

Perform the following steps in the **Schneider Electric Sercos III Parameters** editor of the SLC:

Step	Action
1	Open the parameter group <code>SlcIoConfiguration</code> .
2	Define the data width of the exchange process data which is transferred: • From the SLC to the Logic/Motion Controller (labeled with <code>SLC2LMC_NumberOfxxx</code>). SLC2LMC data can be written in the safety-related application. In the standard application, only read access is allowed to these exchange signals. • From the Logic/Motion Controller to the SLC (labeled with <code>LMC2SLC_NumberOfxxx</code>). LMC2SLC data can be written by the standard application and may be read in the safety-related application (read-only permission).

Step	Action
3	Compile the project in Logic Builder to make the exchange data available in Machine Expert - Safety. Result: According to this configuration, exchange signals are available in Machine Expert - Safety (see chapter) which you can use in the safety-related code via drag and drop from the Devices window. Refer to the example shown below.

Notes on Data Exchange

Observe the following when configuring the exchange data:

- The maximum data width per transfer direction is 128 bytes.
- The value 1 for a B00LGroup reserves a group of 8 bits, that is, 8 Boolean exchange variables. The same applies correspondingly to a B00LGroupExt.
- For exchange data that has been configured in Logic Builder, at least a corresponding global variable must be declared in the safety-related application. In case of a reserved B00LGroup or B00LGroupExt, for at least one Boolean signal of the group, a Boolean global variable must be declared in the safety-related application. Otherwise, a compiler error is generated in Machine Expert - Safety.
- You can map the exchange signals in your Logic/Motion Controller application in the **Schneider Electric Sercos III I/O Mapping** editor.

Application example: A safety-related SF_EmergencyStop function block used in the safety-related application outputs a Boolean error flag. To read this value in the standard application and enable the Logic/Motion Controller to react on a function block error, proceed as described in chapter *Exchanging data between Logic/Motion Controller and SLC* (see page 76).

NOTE: In addition to the exchange signals of the SLC, the safety-related TM5 I/O modules also provide exchange signals. To map these signals in EcoStruxure Machine Expert Logic Builder, double-click the respective TM5 module in the **Devices tree** and open the editor **TM5 Module I/O Mapping**. An example can be found in the chapter *Enabling a Safety-Related Output via the Standard Application* (see page 78).

Example

In the example shown below, one B00LGroup and two Integers are reserved as exchange variables, both with the data direction SLC to Logic/Motion Controller. As a result, they are available in the safety-related application and they should be used in the code, or, at least global variables must be declared for them. In the standard application, only read access is allowed to these variables.

Machine Expert Logic Builder

SLCtoConfiguration			
SLC2LMC_NumberOfBOOLGroups	UINT(0..12)	1	①
SLC2LMC_NumberOfBOOLGroupsExt	UINT(0..32)	0	
SLC2LMC_NumberOfINTs	UINT(0..30)	2	②
SLC2LMC_NumberOfUINTs	UINT(0..30)	0	

Machine Expert - Safety

Channel Name	Slot	Variable	Logic Order Variable	Comment
SL1				SafeLOGIC I...
SL1.SM1	Safety_PL_C			SLC200 SLC ...
BOOL001		ibSafety_PL_C_SLC2LMC_BOOL0_7.0		
BOOL002		ibSafety_PL_C_SLC2LMC_BOOL0_7.1		
BOOL003		ibSafety_PL_C_SLC2LMC_BOOL0_7.2		
BOOL004		ibSafety_PL_C_SLC2LMC_BOOL0_7.3		
BOOL005		ibSafety_PL_C_SLC2LMC_BOOL0_7.4		
BOOL006		ibSafety_PL_C_SLC2LMC_BOOL0_7.5		
BOOL007		ibSafety_PL_C_SLC2LMC_BOOL0_7.6		
BOOL008		ibSafety_PL_C_SLC2LMC_BOOL0_7.7		
INT001		iiSafety_PL_C_SLC2LMC_INT0		
INT002		iiSafety_PL_C_SLC2LMC_INT1		
SafeMachineOpti...				
SL1.SM2	Safety_PL_C...			TM5SDI2DFS...
SL1.SM3	Safety_PL_C...			TM5SDO2TF...

Configuring the Safety-related TM5/TM7 Modules

Overview

In general, safety-related outputs can only be written by the SLC.

To enable the SLC to switch an output channel directly (via the safety-related application, programmed in Machine Expert - Safety) without an enable signal (acknowledgment) from the standard application, the output channel must be configured as described below.

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the respective TM5SDOxxx module to open the parameter editor and click the tab User-Defined Parameters .
2	Set the parameter <code>CentralControl_DigitalOutputs_xx</code> (where xx is the channel number) to <code>Direct</code> .

For further information on this parameter and a description how to use the enable signal from the standard application instead of direct switching by the SLC, refer to chapter *Enabling a safety-related output (acknowledge) via the standard application* ([see page 78](#)).

Configuring the TM5NS31 Bus Coupler

Configuring the TM5 Bus Cycle Time

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the respective TM5NS31 bus coupler to open the parameter editor and click the tab Device Parameters Parameters .
2	Expand the TM5Bus parameter group and set the CycleTime parameter to a value suitable for your application. Refer to the description below this table.
3	Compile the project by selecting the Build → Build menu command or pressing F11 . By compiling, the modified cycle time value is transferred to the safety-related project in EcoStruxure Machine Expert - Safety. Refer to the section <i>TM5 Bus Cycle Time Influences Safety Response Time</i> below this table.

Best practice: In architectures with many TM5/TM7 modules (standard and safety-related) connected to one bus coupler, it may be necessary to increase the TM5 bus cycle time to ensure reliable communication. If one or several TM5/TM7 modules are temporarily unavailable (indication in the **Device tree**), the TM5 bus cycle may be too short. In such cases, increase the value to support more modules at the TM5 bus.

To achieve a higher system performance, decrease the bus cycle time value.

TM5 Bus Cycle Time Influences Safety Response Time

The value of the TM5 bus cycle time influences the calculation of the response time relevant parameters (*see page 52*) and therefore the response time of the safety function.

After modifying the TM5 bus cycle time in the TM5NS31 bus coupler parameters, you must build the standard project to transfer the modified time value to the safety-related project. Based on the modified value, you must recalculate (and adapt) the response time relevant parameters and the safety response time in EcoStruxure Machine Expert - Safety

The TM5 bus cycle time can also be modified via the IEC code of the standard application. The calculation of the response time relevant parameters, however, is based on the **CycleTime** value defined in the TM5NS31 bus coupler parameter editor.

If the actual TM5 bus cycle time differs from the value set in the TM5NS31 bus coupler parameters, observe the following:

** WARNING****NON-CONFORMANCE TO SAFETY FUNCTION REQUIREMENTS**

- If the TM5 bus cycle time is set from within the IEC application program, make sure that the CycleTime parameter is set to the correct value in the TM5NS31 parameter editor.
- Recalculate the response time relevant parameters each time after editing the CycleTime value in the TM5NS31 parameter editor.

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

Commissioning the Logic/Motion Controller - Part 1

Overview

After you have configured the devices, establish a connection to the Logic/Motion Controller and test the Sercos communication. Furthermore, a Sercos phase-up to phase 2 is required.

Connecting and Downloading to Logic/Motion Controller

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the controller node.
2	Click the Communication Settings tab.
3	Double-click your controller in the list. (Update the list, if your controller is not yet listed.)
4	Verify the defined Target IP Address and Connection Mode.
5	Select Online → Login , or click the Login command on the main toolbar, or press Alt + F8 .
6	Confirm the safety-related message by pressing Alt + F .
7	Confirm the login message dialog with Yes .
8	If the controller is executing a different application, you have to confirm that this application is overwritten by the new project.
9	If the controller is already configured, you have to confirm the Post Configuration Warning with OK to continue.
10	Start the application execution by selecting the Debug → Start command or pressing F5 .
11	Confirm the operation start by clicking Yes in the dialog.

Verifying the Sercos Port Setting

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the Ethernet_1 node.
2	Compare the displayed IP address on the left side (Configured) with the right side (Current). Both should show the same value as configured according to chapter <i>Configuring the Ethernet 1 Port</i> (see page 35).

Sercos Phase-Up

The Sercos Master assigns the IP addresses of its slaves during Sercos phase 2. Therefore, you have to perform a Sercos phase-up. In the current scenario, this results in a Sercos error because the SLC is not yet configured (no safety-related program has been downloaded).

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the Sercos_Master .
2	Click the first editor tab Schneider Electric Sercos III parameters .
3	Open the parameter group SercosPhaseChanger. The current value NRT / -1 indicates that the Sercos bus is not running.
4	In the table cell DesiredPhase Prepared Value, select Phase 2 / 2.
5	Write the entered value by selecting the Debug → Write values command or pressing Ctrl + F7 . The cell ActualValue Current Value indicates the phase-up operation of the bus. Result: The phase-up results in PhaseError / 11 because the SLC has not yet received a safety-related application and its communication parameters are no yet defined. However, the IP addresses have been assigned which was the purpose of the phase-up.

Refer to the *Sercos for M262 Logic/Motion Controller User Guide* for details and further information on the Sercos bus and commissioning.

Verifying the SLC Type and Firmware Version

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the Safety PLC node.
2	In the parameter editor on the right, open the Schneider Electric Sercos III Parameters tab.
3	Open the parameter group SlcRelatedConfiguration.
4	Verify that the parameter SafeLogicType corresponds to the SLC type you are using.
5	Open the parameter group ElectronicLabel.
6	Verify that the entry SoftwareRevision matches the required firmware version as stated in the <i>Release notes</i> . If not, update your SLC firmware. You can open the Release notes via the Machine Expert Installer.

Continue with the safety-related part of your application.

Section 3.2

Setting Up the Safety Application

What Is in This Section?

This section contains the following topics:

Topic	Page
First Steps in Machine Expert - Safety	47
Devices Window in Machine Expert - Safety	48
Configuring the Safety Logic Controller	49
Configuring Safety-related TM5/TM7 Module Parameters	54
Calculating the Safety Response Time	55
Programming the Safety-related Application	58
Commissioning the Safety-related Application	65
Validation and Documentation of the Safety-Related Project	73

First Steps in Machine Expert - Safety

Starting Machine Expert - Safety, Log in and Device Confirmation

Step	Action
1	In the Devices tree in Logic Builder, right-click the Safety_PLC node (SLC) and select Machine Expert - Safety → Edit project [Ethernet_1 > Safety_PLC] from the context menu. NOTE: If you are starting Machine Expert - Safety for the first time after the software installation but you did not enter a license key for permanent software activation, you must register once to get a trial version activation. For that purpose, follow the instructions on the screen. Result: Machine Expert - Safety is started and the safety-related project is generated automatically. The Project Log On dialog appears.
2	You must create a password for the project (access) levels Development and Commissioning of the tool as you are opening the safety-related project the first time in Machine Expert - Safety. (The lowest access level Maintenance does not require a password.) The complete functional range from programming up to debugging is provided at Development level. Result: The Confirm changed SDIO Devices dialog appears. Refer to the chapter <i>Password Protection for Projects and Safety Logic Controller</i> in the <i>EcoStruxure Machine Expert - Safety - User Guide</i> for detailed information on password protection.
3	Confirm the safety-related devices by marking the appropriate check boxes and clicking OK.

Notes on Device Synchronization Between EcoStruxure Machine Expert and Machine Expert - Safety

- When opening a safety-related project, the list of safety-related devices in the standard project is synchronized with the device list contained in the safety-related project. This device synchronization is continuously repeated as long as the project remains open in Machine Expert - Safety. This way, any modification in the standard project is recognized and transferred to the safety-related project.
- If you reject the modifications in the device list, Machine Expert - Safety is closed.
- Device synchronization is only possible if you are logged on at Development level in Machine Expert - Safety.
- When applying modifications in the bus configuration to the safety-related project, each modification is entered in the Project Event log and can be traced afterwards.

NOTE: Term definition: Standard = non-safety-related. The term "standard" always refers to non-safety-related items/objects. Examples: a standard process data item is only read/written by a non-safety-related I/O device, that is, a standard device. Standard variables/functions/FBs are non-safety-related data. The term "standard controller" designates the non-safety-related Logic/Motion Controller.

Devices Window in Machine Expert - Safety

Devices Window with Safety-Related Configuration

When you have confirmed the device structure in Machine Expert - Safety, the **Devices** window shows the safety-related part of the architecture configured in Logic Builder.

The **Devices** window is composed of two panes:

- The devices tree on the left contains the safety-related devices. Standard devices are not listed here.

The safety-related part of the Sercos Master is the root element (**SL1**). The SLC, which is always the first safety node (**SL1.SM1**) and the available safety-related I/O modules are contained as child elements.

The tree structure can only be edited in the Logic Builder **Devices** window.

If you select a tree node, its parameters and properties are loaded into the grid-based editor on the right (see next item).

- The **Device Parameterization Editor** on the right contains the editable parameters of the device selected in the tree on the left.

Refer to the chapter *Bus Navigator* in the *EcoStruxure Machine Expert - Safety - User Guide* for detailed information.

Signals and Exchange Variables

The tree nodes of each device can be expanded. Under each device node, the signals (process data items) of the device are listed.

You can drag these signals into the code and use them in your safety-related application.

Depending on the device type, different signal types (control or diagnostic signals) are available.

Under the SLC node, the exchange signals are provided which you have defined in Logic Builder (**Schneider Electric Sercos III Parameters** editor of the **Safety_PLC** node, parameter group **SlcIoConfiguration**).

Refer to the section *Process Data Items* of the respective device-specific chapter in the *Safety Modules - Reference Guide* for detailed information.

Configuring the Safety Logic Controller

SLC Communication Path

As you have created your project based on the **SLC Remote Controller (M262)** example, the communication path between your PC and the SLC is already configured to **SLC connected through LMC**. With this setting, data transmission operations (such as downloading the project, handling of debug data, uploading online values, and so on) are performed via the Logic/Motion Controller which communicates with the SLC.

Modify this setting, if a direct connection to the SLC is required, for instance for testing purposes. To edit the communication path in Machine Expert - Safety, select **Online → TCP/IP Communication parameters** and activate the **SLC100 or SLC200 directly connected** option in the dialog.

For details and further information, refer to the chapter *Communication Settings* in the *EcoStruxure Machine Expert - Safety - User Guide*.

Editing Safety-Related Device Parameters - General Steps

Procedure in Machine Expert - Safety:

Step	Action
1	In the tree on the left of the Devices window, double-click the module to be configured. NOTE: When selecting a module with a left click, the module type and a short description are displayed on the upper border of the window. Result: The module parameters can be edited in the grids on the right.
2	Locate and edit the parameter to be set. If desired, use the tabs on the bottom of the grid to display only a particular parameter category.

Defining the SLC Cycle Time in the SLC

The **CycleTime** parameter sets the cycle time of the SLC. The value must be greater than the processing time for the safety-related application. If the **CycleTime** parameter is smaller than or too close to the processing time, a cycle time error (watchdog timeout) may be detected.

The **CycleTime** value must be an integer multiple of the Sercos cycle time.

Best practice:

Step	Action
1	Set the maximum CycleTime value (20000) as a temporary commissioning value. Result: Due to this maximum cycle time, the safety response time of the safety function may be not suitable for your safety function during this commissioning phase. Refer to the hazard message below this table.
2	Build and download the safety-related application (<i>see page 67</i>) to the SLC.

Step	Action
3	Select Online → SafePLC while EcoStruxure Machine Expert - Safety is running in online mode. Result: The SafePLC control dialog opens.
4	In the SafePLC control dialog, click the Info button. Result: The SafePLC Info dialog opens, displaying the current processing time.
5	Determine the SLC cycle time by rounding up the displayed processing time value to the next multiple of the Sercos cycle time. Enter this value as CycleTime in the parameter editor.
6	Rebuild the safety-related project and download it again to the SLC. Result: After the restart, the SLC should run in normal operation.

WARNING

NON-CONFORMANCE TO SAFETY FUNCTION REQUIREMENTS

- Verify the impact of the increased safety response time.
- Make certain that appropriate procedures and measures (according to applicable sector standards) have been taken to help avoid hazardous situations during the commissioning phase.
- Do not enter the zone of operation while running the SLC with the maximum cycle time.
- Ensure that no other persons can access the zone of operation while running the SLC with the maximum cycle time.
- Use appropriate safety interlocks where personnel and/or equipment hazards exist.

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

Defining the SSD0Creation Parameter

The parameter **SSD0Creation** (SSDO = Safety protocol Service Data Object) defines the number of acyclic processing steps per SLC cycle.

The SLC sends SSDO telegrams mainly during the start-up of the system in order to assign and verify the SADR (Safety protocol Address) of the SNs, to verify the uniqueness of the UDIDs within the safety network, and for downloading parameters and DVI (Device Vendor Information) to the SNs.

The **SSD0Creation** parameter can be used to optimize the restart behavior of the system provided that the Sercos bus has enough bandwidth on its asynchronous channel. (Chapter *Defining the Sercos Bus Cycle Time* ([see page 36](#)) describes possibilities to increase the bandwidth of the asynchronous Sercos channel.)

The higher the number of asynchronous processing steps per SLC cycle the faster the restart of the safety system is.

Details on the possible values can be found in the chapter *TM5CSLCx00FS Safety Logic Controller* (section *Basic*) of the *Safety Modules - Reference Guide*.

Best practice: The boot-up time of the system can be reduced when setting `SSDOCreation` to 5 per cycle. With this setting, the SLC transmits five SSDO telegrams per cycle. This allows the SLC to find the safety nodes in a short time.

Defining the `NodeGuardingTimeout` Parameter

The `NodeGuardingTimeout` parameter sets the period (timeout value) to put the safety-related modules in pre-operational state when the SLC is incommunicative or in case of communication interruptions detected between the safety-related module and the SLC. It also defines the delay for the SLC to detect an unavailable module.

The `NodeGuardingTimeout` value is not critical to functional safety. The time for turning off actuators is determined independently using the safety response time relevant parameters.

Details on the possible values can be found in the chapter *TM5CSLCx00FS Safety Logic Controller* (section *Basic*) of the *Safety Modules - Reference Guide*.

NOTE: After a Sercos phase-down, the next Sercos phase-up should not be executed before the duration specified via the `NodeGuardingTimeout` parameter has expired. Otherwise, the SNs in a large system are possibly not scanned and configured by the SLC within the set time limit which then results in a timeout.

Defining the `NumberOfScans` Parameter

The `NumberOfScans` parameter specifies the number of module scans the SLC performs before it indicates if modules are unavailable (MXCHG flashing rapidly). Scanning is continued even after the SLC has triggered the LED for unavailable modules.

Defining the `RemoteControlAllowed` Parameter

The `RemoteControlAllowed` parameter enables or disables the remote control of the Safety Logic Controller.

Set this parameter to Yes-ATTENTION as the example project is based on the **SlcRemoteController** example.

Take into consideration the inherent hazards involved in a remote control operation to avoid unintentional equipment operation.

WARNING

UNINTENDED EQUIPMENT OPERATION

Ensure that there is a local, competent, and qualified observer present when operating from a remote location.

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

Configuring the Safety Response Time Relevant Parameters

The safety response time is the time between the arrival of the sensor signal on the input channel of a safety-related input module and the shut-off signal at the output channel of a safety-related module.

The SLC as well as each safety-related TM5/TM7 module provide three parameters which influence the safety response time of the safety-related application.

The safety response time relevant parameters are used for timing validation purposes:

- `MinDataTransportTime` defines the minimum time that is required to transmit a data telegram from a safety-related producer to a consumer. If a telegram is received earlier (by the consumer) than specified by this parameter value, communication is considered as invalid.
- `MaxDataTransportTime` defines the maximum time that is allowed to transmit a data telegram from a producer to a consumer. If a telegram is received later (by the consumer) than specified by this parameter value, communication is considered as invalid.
- `CommunicationWatchdog` defines the maximum time period within which a consumer must receive a valid data telegram from a producer in order to consider the safety-related communication as valid and continue the application.

For the SLC, these parameters are combined in the group `SafetyResponseTimeDefaults`.

Select **Project → Response Time Relevant Parameters** to open the parameter calculation dialog. In the calculation dialog, open the **Default** tab. Proceed as described in the chapter *TM5SLCx00FS Safety Logic Controller* (section *Group: SafetyResponseTimeDefaults*) of the *Safety Modules - Reference Guide* to determine the correct parameter values for your application.

Best practice:

- Set the **Network Package Loss** parameter in the **Response Time Relevant Parameters** dialog to 1 (default value). This is identical to the configuration of Sercos (one data loss is allowed).
- If the SNs do not reach the operational state with the calculated values (for example, in a large system, or if optional devices are configured), slightly increase the `MaxDataTransportTime` parameter. Maximum value: 1.5 times the amount of the calculated value.

For further and detailed background information, also refer to the chapter *Safety Response Time* of the *EcoStruxure Machine Expert - Safety - User Guide*.

The response time relevant parameters are influenced by the value of the TM5 bus cycle time. After modifying the TM5 bus cycle time in the TM5NS31 bus coupler parameters ([see page 42](#)), you must build the standard project to transfer the modified time value to the safety-related project. Based on the modified value, you must recalculate (and adapt) the response time relevant parameters and the safety response time in EcoStruxure Machine Expert - Safety.

The TM5 bus cycle time can also be modified via the IEC code of the standard application. The calculation of the response time relevant parameters, however, is based on the `CycleTime` value defined in the TM5NS31 bus coupler parameter editor.

If the actual TM5 bus cycle time differs from the value set in the TM5NS31 bus coupler parameters, you must observe the following:

WARNING

NON-CONFORMANCE TO SAFETY FUNCTION REQUIREMENTS

- If the TM5 bus cycle time is set from within the IEC application program, make sure that the CycleTime parameter is set to the correct value in the TM5NS31 parameter editor.
- Recalculate the response time relevant parameters each time after editing the CycleTime value in the TM5NS31 parameter editor.

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

Current Status of your Safety Project

The safety-related device parameters of the SLC are now configured and, due to the Sercos partial phase-up (to phase 2) you have already performed (see section *Sercos Phase-Up* ([see page 44](#))), an IP address has been assigned by the Sercos Master. You can now connect to the SLC.

If you compile the empty safety-related project at this stage for test purposes, the compiler reports errors. This is because your safety-related project contains unused safety-related TM5/TM7 modules. Unused means that none of the signals, which are listed under the device nodes in the Machine Expert - Safety **Devices** window, are used in the safety-related project. At least one signal of each module must be assigned to a global safety-related variable in Machine Expert - Safety. The same applies to the SLC exchange signals you have defined in Logic Builder (refer to section *Exchange Data Configuration for the Safety PLC* ([see page 38](#))).

Configuring Safety-related TM5/TM7 Module Parameters

General Information on Editing Parameters

The parameters of the safety-related modules can be set in the safety-related **Device Parameterization Editor**. This editor is part of the **Devices** window in Machine Expert - Safety.

To edit the parameters of a safety-related module, left-click the respective tree node in the **Devices** window. Its parameters can then be edited in the tabs on the right.

Refer to the *Safety Modules - Reference Guide* for details on device parameters.

Relevant Module Parameters

The following safety-related device parameters are relevant in the described example project. As you have created your project based on the **SLC Remote Controller (M262)**, they are mostly already configured correctly. Adapt the values if required.

- Parameter `Optional` - defines a module as optional or mandatory. Select No for this example project to define the module as mandatory.
- Parameter group `SafetyResponseTime` which is composed of several parameters concerning the minimum and maximum data transport time and the communication watchdog. Refer to the next section ([see page 54](#)) for a detailed procedure to set the parameters.
- Input channel related parameters, such as filter times, pulse source/mode, or parameters concerning the antivalence/equivalence monitoring.
- Output channel related parameters which define the restart behavior of the module.

Refer to the respective module-specific chapter in the *Safety Modules Parameter Guide* for further information on the parameters.

Response Time Relevant Parameters for TM5/TM7 Modules

For the described example project, set the `ManualConfiguration` parameter of each module to No. With this setting, the default values defined in the parameter group `SafetyResponseTimeDefaults` of the Safety Logic Controller are also applied to the safety-related TM5/TM7 modules. This means that their individual response times only differ due to module-specific processing times as they use a common `MinDataTransportTime`, `MaxDataTransportTime`, and `CommunicationWatchdog` value.

Refer to the chapter *Determining and Setting the Safety Response Time Parameters* ([see page 52](#)) for further information.

In case that you want a module to use its own parameter values, set `ManualConfiguration` to Yes. Then you must calculate and enter the response time relevant parameters for the respective module.

For further and detailed background information, also refer to the chapter *Safety Response Time* of the *EcoStruxure Machine Expert - Safety - User Guide*.

Calculating the Safety Response Time

Preconditions

Preconditions for calculating the safety response time of your safety-related application:

- Correct values are defined for the relevant parameters of the Safety Logic Controller (see chapter *Determining and Setting the Safety Response Time Parameters (see page 52)*).
- Parameter ManualConfiguration is set to No for the I/O modules. With this setting, the values defined for the Safety Logic Controller are also applied to the I/O modules.
- If the ManualConfiguration parameter is set to Yes for the I/O modules: Make sure that correct values are defined for the relevant parameters of each module involved. Refer to the chapter *Relevant Module Parameters (see page 54)* for details.

General Information

The safety response time is the time between the arrival of the sensor signal on the safety-related input module and the output of the request signal for the defined safe state at the safety-related output module.

Based on the calculated safety response time, the safety equipment must be planned and installed. For example, the safety response time delivers the required minimum distance of a safety-related sensor (such as a light beam) from the zone of operation.

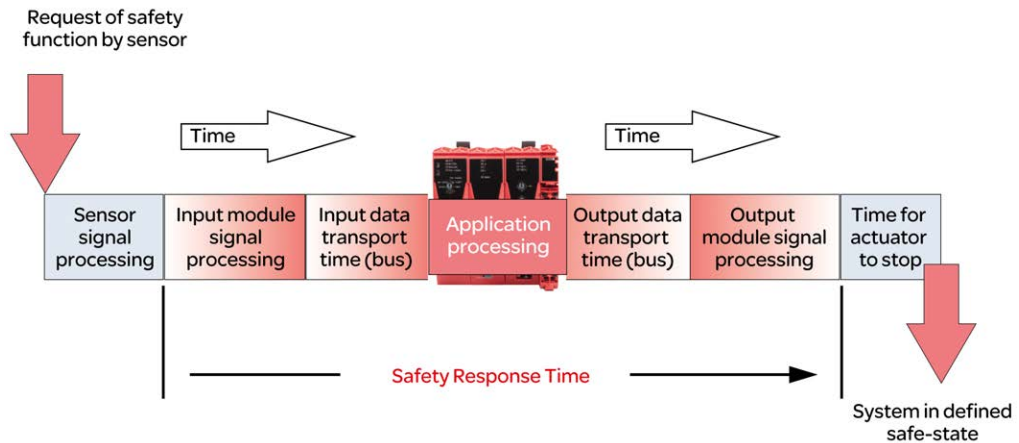
WARNING

UNINTENDED EQUIPMENT OPERATION

- Place operator devices of the control system near the machine or in a place where you have full view of the machine.
- Protect operator commands against unauthorized access.

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

The following figure illustrates the influences on the safety response time:



The safety response time (SRT) is calculated as follows:

SRT = processing time in the safety-related input module (see next section)

- + input transport time (bus transfer input module -> SLC)
- + application processing time in the SLC
- + output transport time (bus transfer SLC -> output module)
- + processing time in the safety-related output module (see next section)

As shown in the figure above, the total lag time of the entire system from the occurrence of the safety function requesting event until the machine/plant is in the defined safe-state, also comprises the signal processing time of the safety-related sensor as well as the time for the actuator to stop.

For further and detailed background information, also refer to the chapter *Safety Response Time* of the *EcoStruxure Machine Expert - Safety - User Guide*.

Processing Times in I/O Modules

For safety-related Schneider Electric I/O modules, the following signal processing times must be considered.

Schneider Electric input modules:

- Configured filter value of the switch-off filter
- 5000 µs when configuring the external clock signals
- I/O update time for TM5SAI4AFS (analog current measurement) and TM5STI4ATCFS (analog temperature measurement)
- Module processing time (timebase + I/O update time) for counter (TM5SDC1FS) module

NOTE: The I/O update time value depends on the configured input filter parameter. The module processing time depends on the configured timebase.

Schneider Electric output modules:

- TM5SDOxxxx modules: 800 µs max.
- TM5SDM4DTRFS: 50 ms max. (integrated relay)
- TM7SDM12DTFS: 1 ms max.

Calculating the Safety Response Time

Procedure in Machine Expert - Safety:

Step	Action
1	Select Project → Response Time Calculator Result: The Result section at the bottom shows the overall worst case response time for the user-defined functional safety system.
2	You can now calculate the safety response time for individual input-output signal paths as follows: Select the input module for which the response time is to be calculated, and, if applicable, an input channel of this module. Result: The response time relevant parameters set for the selected input module/channel are shown in the area below. While no module is selected, the parameter values are zero.
3	In the right list, select the output Module for which the response time is to be calculated. Result: The dialog automatically shows the calculated response times for the selected path. Document these values for verification purposes. Observe the hazard message below this table.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the signal processing time within the sensor is included in the overall safety response time calculations.
- Verify that the time required by the actuator to come to a standstill is included in the overall safety response time calculations.
- Validate the total lag-time of the system and thoroughly test the application controlling for lag-time.

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

NOTE: During commissioning and operation of the system, the safety response time has to be optimized, if required.

Programming the Safety-related Application

General information on the Safety-related Project

The following list provides essential facts on the user interface of Machine Expert - Safety and the characteristics of safety-related code and variables.

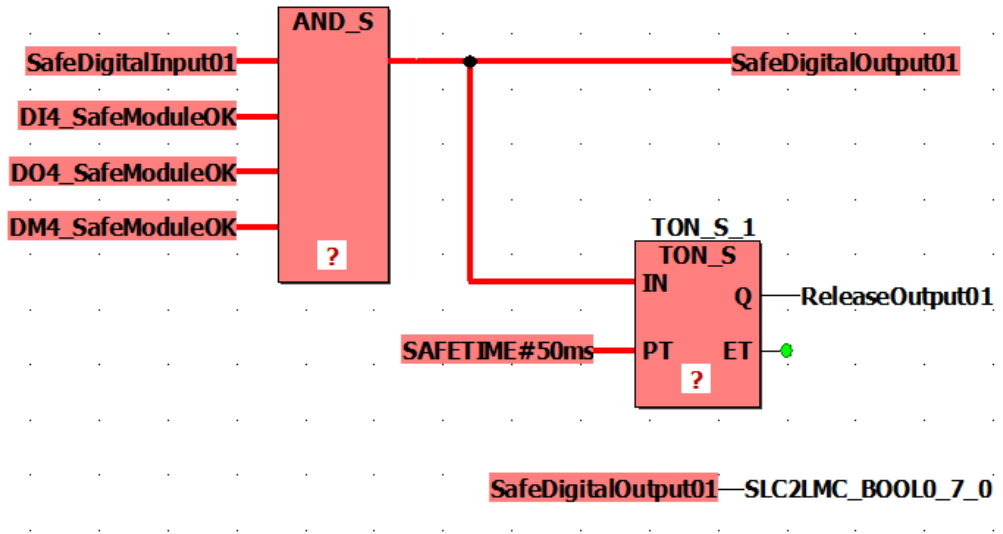
For further information and details, refer to the *EcoStruxure Machine Expert - Safety - User Guide*.

- POU's are organized in the **Project Tree** Window.
- A safety-related project contains exactly one POU of the IEC 61131-3 type Program named Main. This POU cannot be deleted or renamed and no further user-defined IEC 61131-3 programs can be added (only FBs).
- The safety-related task in which this program is executed is also predefined but not visible in Machine Expert - Safety. You cannot edit this task configuration.
NOTE: As only one safety-related task is executed by the SLC, a modification of the SLC cycle time (*see page 49*) has the same effect as changing the task cycle time.
- You can create user-defined safety-related function blocks (according to IEC 61131-3) but no functions.
- You can insert libraries which provide safety-related functions and function blocks.
- Each POU is composed of one or several code worksheets and a variable table with local variable declarations. Double-click a tree icon to open the corresponding worksheet for editing.
- Global variable declarations are contained in a separate variable grid. Click the **Global decl.** icon on the main toolbar to open this table.
- The **Edit Wizard** provides functions and function blocks. After you have added a POU library (via the context menu of the **Libraries** folder in the project tree), the contained blocks can be selected in a separate **Group**.
- Safety-related and standard code is strictly distinguished in Machine Expert - Safety. Therefore, also safety-related and standard variables, or more precise, safety-related and standard data types, are distinguished. It is, for example, not possible to connect a variable with a standard data type to a formal parameter which expects a safety-related variable.
Safety-related variables are displayed with a red background in the code. Variables of standard data types are shown without background.
- Safety-related system FUs/FBs as well as safety-related library FBs are displayed in red. Standard blocks are shown gray-blue. User FUs/FBs are displayed in green.
- When mixing safety-related and standard variables, Machine Expert - Safety performs a data flow analysis in the FBD/LD code and highlights the leading safety-related signal paths of a network by displaying them as thick red lines. A safety-related path always ends either at a safety-related output variable or, in case of a standard output variable, at the last object input located before this output. If a standard signal path ends at a safety-related output, this output is shown with a background hatched in red.

Safety Application Example

The following simple program considers the TM5 I/O modules configured in the example project. The procedures to develop this example are described in the following sections.

Refer to the chapter *FBD/LD Code Development* in the *EcoStruxure Machine Expert User Guide* for a comprehensive description of the editor functions.



The input signal `SafeDigitalInput01` of the TM5SDI4DFS module is read and mapped via the `AND_S` function to the output signal `SafeDigitalOutput01` of the TM5SDO4TFS module. Due to the `AND_S` function, the `SafeModuleOK` diagnostic signals of the safety-related I/O modules are evaluated. A failure detected in any module switches the `SafeDigitalOutput01` signal off (`SAFEFALSE`).

In addition, the `SafeDigitalOutput01` is written to the Boolean exchange variable `SLC2LMC_Boolean_7_0` which belongs to the `SLC2LMC` exchange data configured in the SLC device configuration in Logic Builder. This way, the standard application can read the status of the output. (The `SafeDigitalOutputxx` signal is available for digital output modules. It signals the standard application whether the safety-related output is set by the safety application). The direct connection of the safety-related variable to the standard exchange variable `SLC2LMC_Boolean_7_0` is possible because type conversions from safety-related to standard data types are allowed.

The timer function block `TON_S` delays the `ReleaseOutput01` signal. This release signal deactivates an active restart inhibit and enables the output channel of the `SafeDigitalOutput01` signal of the TM5SDO4TFS module. The delay time is set to 50 ms.

NOTE: This programmed delay time influences the overall safety response time of the system (see page 57).

Inserting a Function/Function Block into the Code

Execute the following steps for the AND_S function and the TON_S function block:

Step	Action
1	Open the Main program worksheet by double-clicking its icon in the Project Tree Window .
2	In the Edit Wizard selection area, select the desired block. If the block is not visible, you must first select the Group <all FUs and FBs> .
3	Drag the block from the Edit Wizard selection area into the code worksheet, left-click to insert the block outline, and drop it with another left mouse click at the desired position.
4	In case of a function block (TON_S in the example), an instance variable must be declared. Result: The Variable dialog appears proposing an instance name which you can modify, if desired.
5	In the Variable dialog, click OK . Result: The function block instance is inserted into the code and the related instance variable is inserted into the local declarations of the Main POU. You can open the declaration worksheet by clicking the ToggleWS icon in the main toolbar.
6	In the example, the AND_S function needs four inputs. To adapt the function, right-click the block icon and select Object Properties from the context menu. Select IN2 in the Formal Parameters list and click Duplicate FP twice to add two further inputs. Close the dialog with OK .

Inserting Device Signals into the Code

The following procedure applies to the device signals that are provided under the device nodes in the **Devices** window. This includes exchange variables defined for the SLC as well as diagnostic and control signals of the safety-related I/O modules.

Procedure in Machine Expert - Safety:

Step	Action
1	Open the code worksheet where you want to insert the signal.
2	In the Devices window, open the devices tree on the left and expand the node of the desired module (SL1.SMx).
3	Drag the desired signal into the code worksheet. Result: When releasing the mouse button, the Variable dialog appears.
4	In the Variable dialog, accept the proposed name, select an existing global variable, or declare a new global variable. Refer to the figure for the variable names used in the example.
5	Confirm the Variable dialog by clicking OK and drop the variable at the desired position with a left click. Result: The variable is inserted into the code and its variable declaration is automatically inserted into the global variable worksheet. You can directly drop the variable on a block output or input to connect it on insertion.

For the example, insert the following signals in the described way:

- SafeDigitalInput01 of the TM5SDI4DFS module connected to an AND_S input.
- SafeModule0K of each I/O module connected to an AND_S input.
- SafeDigitalOutput01 of the TM5SDO4TFS module connected to the AND_S output.
Insert the variable a second time and drop it at a free position without any connection.
- ReleaseOutput01 of the TM5SDO4TFS module connected to the TON_S output.
- SLC2LMC_Boolean_7_0 exchange variable of the SLC connected to the input (blue connection point) of the unconnected SafeDigitalOutput01 variable. This way, the output variable is written to the Boolean exchange variable.

Inserting Constant (Literals) into the Code

The following procedure describes how to insert literals into the code. Literals have to be used to enter constant values in the code. They can be used without specifying a declaration.

Step	Action
1	You can insert unconnected or connected/assigned constants: • To insert a constant already connected to a function or function block, double-click the desired formal parameter. • To insert a constant not connected to any object, click at a free worksheet position and press F5 or click the Variable icon on the editor toolbar. Result: The Variable dialog appears.
2	Specify Scope = Constant .
3	A data type is proposed in the Type combo box. Adapt this setting, if required.
4	Enter the desired literal (constant) in the Name field. Observe the rules below this table.
5	Press OK. Result: The constant is inserted into the FBD/LD code.

Refer to the chapter *Constants (Literals): Inserting and Declaring* in the *EcoStruxure Machine Expert User Guide* for further details on constants and the special case “Global Constants”.

Rules for constants:

- Literals must always be entered including the data type (for example, SAFEINT#1000).
Exceptions: TRUE and FALSE are always handled as BOOL and SAFETRUE/SAFEFALSE are always handled as SAFEBOOL. It is, for example, not necessary to enter BOOL#TRUE.
- Standard INT constants can be entered without data type (for example, 1000 means INT#1000) as decimal inputs are automatically interpreted as INT.
Exception: 0 and 1 if used with Boolean data type.

Refer to the chapter *Constants vs. Literals* in the *EcoStruxure Machine Expert User Guide* for further information on literals according to the IEC 61131-3 standard.

Inserting New Variables into the Code

The following procedure describes how to insert new variables into the code. The declaration is automatically inserted into the respective declaration worksheet.

Step	Action
1	You can insert unconnected or connected/assigned variables: ● To insert a variable already connected to a function or function block, double-click the desired formal parameter. ● To insert a variable not connected to any object, click at a free worksheet position and press F5 or click the Variable icon on the editor toolbar. ● To insert a variable for a contact or coil, double-click the particular LD object. Result: The Variable dialog appears.
2	Select the Scope of the variable. Result: For local variables, the declaration is inserted into the declaration worksheet of the current POU (to be opened using the ToggleWS icon). A global declaration is inserted into the global declaration worksheet which you can open by clicking the Global Decl. icon.
3	Specify the data type of the new variable, enter a variable Name, and define the remaining properties.
4	Press OK. Result: The variable is inserted into the FBD/LD code and the declaration into the corresponding declaration worksheet.

There are more possibilities for declaring variables. Refer to the chapter *Variables: Inserting and Declaring* in the *EcoStruxure Machine Expert User Guide* for details.

Connecting Objects in the Graphical Code

To draw lines between objects and block formal parameters, you must activate the connection mode by clicking the **Connect** icon on the editor toolbar.

Clicking the **Mark** icon on the editor toolbar switches the editor to mark mode in which you can select and move objects.

Declaring a Safety-Related Variable for a Device Signal

The safety-related project must not contain unused safety-related TM5/TM7 modules. Unused means that none of the signals, which are listed under the device node in the Machine Expert - Safety **Devices** window, is used in the safety-related project. At least one signal of each module must be assigned to a global safety-related variable in Machine Expert - Safety. Otherwise, the compiler reports errors.

The same applies to the SLC exchange signals you have defined in Logic Builder (see section *Exchange Data Configuration for the Safety PLC* ([see page 38](#))).

NOTE: Declaring a safety-related variable and assigning it to a device signal without using it in the code is useful during project development as it makes the safety-related project compilable. In a practical application, you must make sure that the relevant variables are read or written in the safety-related application program.

The following steps apply to each type of signal provided in the Machine Expert - Safety **Devices** window:

Step	Action
1	In Machine Expert - Safety, open the global variables worksheet by clicking the Global decl. icon on the toolbar.
2	Right-click into the grid and select New Variable from the context menu. Result: A new variable with a default name (which you can modify) is created.
3	In the Devices window, open the device tree on the left. Expand the tree node of the device of which a device terminal is to be used.
4	Drag the device signal to be connected into the global variables worksheet and drop it on the desired declaration. Result: • The Channel Name of the connected device signal is now visible in the Terminal column of the global declaration in the variables worksheet. • The data type of the global variable has been adapted to the data type of the assigned device signal. • In the Devices window, the name of the connected variable is shown for this device signal in the Variable column.

NOTE: With this procedure, you can also replace existing assignments between global variables and device signals. Observe the hazard messages in the chapter *Connecting/Disconnecting Process Data Items and Global I/O Variables* of the *EcoStruxure Machine Expert - Safety - User Guide*.

To insert a declared variable into the code, use the **Variable** dialog which you open via the **Variable** icon on the editor toolbar. Refer to the chapter *Variables: Inserting and Declaring* of the *EcoStruxure Machine Expert - Safety - User Guide* for details.

Compiling the Safety-related Project

After you have finished the development of the safety-related project, you must compile it. (If a POU is marked with an asterisk (*) in the project tree, it has not yet been compiled after editing variables or code. After the successful compilation, the asterisk is removed.)

Procedure in Machine Expert - Safety:

Step	Action
1	Press F9 or click the Compile icon on the toolbar.
2	Correct any errors that the compiler has detected and reported in the message window. Double-click an error message to jump to the suspected error position.

Step	Action
3	After having compiled the project without any errors, proceed with the download of the project to the SLC. Refer to chapter <i>Downloading the safety-related application</i> (see page 65).

Commissioning the Safety-related Application

Safety Logic Controller Password

The SLC password protects the configuration on the Safety Logic Controller against unauthorized access and unauthorized switching of the operating mode.

If you are connecting the first time to an unconfigured Safety Logic Controller, you have to define an SLC password. If a password has already been defined (for example in an earlier session or via the SlcRemoteController visualization), enter this password and click **OK** to log on.

The minimum password length is six characters. The password is case-sensitive and can be a mix of up to 10 characters. Refer to the chapter *Password Protection for Projects and Safety Logic Controller* in the *EcoStruxure Machine Expert - Safety - User Guide* for detailed information.

Safety Logic Controller Operating Modes

The Safety Logic Controller can run in two different operating modes. The operating can be controlled via the **SafePLC** dialog. To open this control dialog, click the **SafePLC** icon on the main toolbar.

Refer to the section *Safety Logic Controller Password* ([see page 65](#)) for details on the logon procedure and password definition.

Description of the **SafePLC** operating modes:

SLC operating mode	Meaning
Safe mode	The safe mode restricts operations in the project that would otherwise influence the state or mode of operation of the SLC. In safe mode, it is possible to: ● Switch the SLC to debug mode. ● Display the variables status (view online values of variables). ● Read out SLC errors via the Error button in the control dialog. In safe mode, the SafePLC dialog is red. The Debug button is available in the SafePLC dialog to switch to debug mode.

SLC operating mode	Meaning
Debug mode	Switching to debug mode means leaving the safe mode of operation. This is only possible after entering the correct SLC password (see section <i>Safety Logic Controller Password</i> (see page 65) for details). NOTE: Switching to debug mode does not stop the program execution on the SLC. In debug mode it is possible to: • Switch the SLC to safe mode. • Download the project (see page 67) to the SLC (only while not executing the program). • Start or stop program execution. • Display the variables status (see page 68) (online values). • Execute debug commands (see page 70) such as forcing and overwriting. • Read out SLC errors via the Error button in the control dialog. In debug mode, the SafePLC dialog is gray. The Safe button is visible in the SafePLC dialog to switch to safe mode.

After you have clicked the **Debug** or **Safe** button to activate the other mode, you must confirm the mode transition within 30 seconds to activate the desired mode.

Debug Watchdog

If the SLC runs in debug mode and the connection between Machine Expert - Safety and SLC is interrupted, or the control dialog is closed and the variable status is deactivated, a debug watchdog timer starts. If the connection to the SLC can be reestablished and you continue debugging or switch the target back to safe mode within 10 minutes, the debug watchdog is reset. If the debug watchdog timer exceeds 10 minutes, the SLC sets the state to STOP [Debug] and writes an error to the error stack. The machine is signaled to assume the defined safe-state. You cannot switch to safe mode again. In this case, you have to restart the SLC.

Safety Logic Controller States

The state machine of the Safety Logic Controller knows several different states. The current state is displayed in the **SafePLC** dialog. You can open this control dialog by clicking the **SafePLC** icon on the main toolbar.

Possible states are:

SLC state	Meaning
On	SLC power supply on, no program downloaded.
No Execution	Program downloaded, start up in progress.
STOP [Safe]	Program loaded but not executing. I/O images are not updated.
RUN [Safe]	Program is executing. Variable status possible.
STOP [Debug]	Program is not executing. Download possible.

SLC state	Meaning
RUN [Debug]	Program is executing. Variable status and forcing/overwriting/single cycle mode possible.
HALT [Debug]	Program is halted in single cycle mode.

NOTE: If the Sercos bus is not at least in phase 2 (or if it is in NRT state), the state display in the **SafePLC** control dialog differs from the `SlcProjectStatus` displayed in EcoStruxure Machine Expert Logic Builder. Even in Sercos NRT state, the SLC may run in RUN [Safe] mode.

Downloading and Starting the Safety Application

After you have compiled your project without errors (see section *Compiling the Safety-Related Project* (see page 63)), you must download it to the Safety Logic Controller. The download includes the machine-readable application code as well as the parameterization data.

 WARNING
UNINTENDED EQUIPMENT OPERATION • Ensure that suitable organizational measures (according to applicable sector standards) have been taken to avoid hazardous situations if the safety logic application operates in an unintended or incorrect way, or an incorrect target for the download was selected. • Do not enter the zone of operation while the machine is operating. • Ensure that no other persons can access the zone of operation while the machine is operating. • Observe the regulations given by relevant sector standards while the machine is running in any other operating mode than "operational". • Use appropriate safety interlocks where personnel and/or equipment hazards exist. Failure to follow these instructions can result in death, serious injury, or equipment damage.

Download procedure in Machine Expert - Safety:

Step	Action
1	Before downloading the project, ensure that you have set the SLC communication path (see section <i>Defining the Communication Path</i>) and the SLC is connected, and switched on.
2	Click the SafePLC icon on the toolbar. Result: • The system verifies whether it was previously connected to the same or a different SLC and the program in the SLC is different from the compiled project in EcoStruxure Machine Expert - Safety. If so, confirm the appearing dialog with Yes. • The login dialog appears if you are not yet logged-on to the SLC.

Step	Action
3	If you are connecting the first time to an unconfigured Safety Logic Controller, you have to define a Safety Logic Controller password. Refer to the section <i>Safety Logic Controller Password</i> (see page 65) for details. Result: The SafePLC control dialog appears. NOTE: If the simulation mode is activated and safe mode is simulated, the SafePLC dialog only shows a red border instead of a red background. In debug mode, no difference is visible between simulation and Safety Logic Controller. Make certain that the desired target (Safety Logic Controller or simulation) is connected when working with the dialog. Refer to the chapter <i>Using the Simulation in the EcoStruxure Machine Expert - Safety - User Guide</i> for information how to activate/deactivate the simulation mode.
4	In the SafePLC control dialog, click the Debug button to switch the Safety Logic Controller to debug mode (if not yet activated). Result: A confirmation message box appears.
5	Observe the message and confirm the dialog within 30 seconds. Result: • If the SLC is stopped, the Download button is active. • If the SLC is in RUN [Debug] state, click Stop to enable the Download button.
6	In the SafePLC control dialog, click the Download button. Result: • If another project is stored on the SLC or another user has downloaded the same project, click Yes to overwrite it. • The status bar indicates the download process and a message informs about the successful project download.
7	Confirm this message. Result: The SLC is restarted and then transitions to a RUN [Safe] state automatically. Depending on the configuration, this may take some time. Observe the note below the table. For detailed information on the possible SLC states, refer to the chapter <i>Safety Logic Controller States</i> in the <i>EcoStruxure Machine Expert - Safety - User Guide</i>.
8	Perform a functional test on the project and monitor the application (see page 68).

NOTE: If the Sercos bus is not at least in phase 2 (or if it is in NRT state), the SLC enters the RUN [Safe] state after the download. This enables the debugging of the safety-related application even if no Logic/Motion Controller is connected or the Sercos bus is down. Therefore, the state display in the **SafePLC** control dialog in Machine Expert - Safety differs from the SLCProjectStatus displayed in EcoStruxure Machine Expert Logic Builder.

Functional Test and Monitoring the Safety Application

After you have downloaded the project to the SLC, followed by the automatic transition to RUN [Safe] state, you must perform a functional test to ensure that the SLC is working correctly and, therefore, that the safety logic and the wiring are working correctly as well. The functional test must also include the positioning of the safety equipment and the verification of the correctly set safety response time ([see page 55](#)).

WARNING

NON-CONFORMANCE TO SAFETY FUNCTION REQUIREMENTS

Be sure that the functional testing you perform entirely corresponds to your risk analysis and consider each possible operating mode and scenario the safety-related application should cover.

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

When testing and commissioning the system, unintentional system states or incorrect responses must be anticipated.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Make certain that the functional testing cannot result in hazardous situations for persons or material.
- Make certain that requesting the safety function during the functional testing cannot result in hazardous situations for persons or material.
- Do not enter the zone of operation while the machine is operating.
- Ensure that no other persons can access the zone of operation while the machine is operating.
- Observe the regulations given by relevant sector standards while the machine is running in any other operating mode than "operational".
- Use appropriate safety interlocks where personnel and/or equipment hazards exist.

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

To support you in functional testing, Machine Expert - Safety enables you to open code/variables worksheets in online mode and display the variable status. This means that the variables values are cyclically read from the SLC and displayed in the worksheets as they are stored in the I/O image at the end of an execution cycle. The variable status corresponds to online monitoring of worksheets.

Variable status is possible while the SLC is running in safe mode and in debug mode.

Online monitoring of the safety application:

Step	Action
1	Click the Variable status icon on the toolbar or press F10. Result: ● The system verifies whether it was previously connected to the same or a different SLC and if the program in the SLC fits to the compiled project in EcoStruxure Machine Expert - Safety. If so, confirm the appearing dialog with Yes. ● The open worksheets are switched automatically to online mode.

Step	Action
2	If open function block code worksheets are instantiated several times and you want to display the variable status for these worksheets, a message appears. This dialog indicates that you must use the Open instance menu command to call these worksheets in online mode.

Refer to the chapter *Monitoring: Displaying the Variable Status* in the *EcoStruxure Machine Expert - Safety - User Guide* for details on the layout and colors used in online worksheets. This applies also to the watch window which you can use for collecting variables from different worksheets and displaying their online values (help chapter *Monitoring: Using the Watch Window*).

Debugging the Safety Application (Forcing, Overwriting)

As a supplement to the functional system test, you can use the debug mode in Machine Expert - Safety while commissioning the application. In debug mode, you can force and overwrite variables.

Forcing and overwriting means assigning a new value to a variable. Overwriting is possible for variables without assigned signal (only memory variables but not I/O variables). The value is overwritten (set) only once at the beginning of the task execution cycle. Then, the variable is processed normally. Thus, the new value of the variable remains until a write access is performed within the application. Forcing is only possible for variables connected to process data items (I/O variables). Forcing means setting the I/O variable to the force value, regardless of the logic of the I/O image until forcing is manually reset.

NOTE: Generally, forcing is performed once per cycle. Inputs are forced at the beginning of a cycle before processing the input variable. This way, the Safety Logic Controller application uses the forced value. Outputs are forced at the end of a cycle. The variable value processed by the application is finally replaced by the forced value in the output image.

 **WARNING**

UNINTENDED EQUIPMENT OPERATION

- Ensure that suitable organizational measures (according to applicable sector standards) have been taken to avoid hazardous situations if the safety logic application operates in an unintended or incorrect way, or an incorrect target for debugging was selected.
- Verify the impact of forcing or overwriting variables or using the single cycle operation prior to their use.
- Do not enter the zone of operation while the machine is operating.
- Ensure that no other persons can access the zone of operation while the machine is operating.
- Observe the regulations given by relevant sector standards while the machine is running in any other operating mode than "operational".
- Use appropriate safety interlocks where personnel and/or equipment hazards exist.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have a thorough understanding of how forcing will affect the outputs relative to the tasks being executed.
- Do not attempt to force I/O that is contained in tasks that you are not certain will be executed in a timely manner, unless your intent is for the forcing to take affect at the next execution of the task whenever that may be.
- If you force an output and there is no apparent affect on the physical output, do not exit EcoStruxure Machine Expert - Safety without removing the forcing.

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

Forcing/overwriting a variable in graphical FBD/LD code:

Step	Action
1	Click the SafePLC icon on the toolbar and log-on to the SLC. See section <i>Downloading and Starting the Safety Application</i> (see page 67) for details.
2	In the control dialog, click the Debug button.
3	Observe the appearing message and confirm the dialog within 30 seconds.
4	Open the worksheet to be debugged in variable status by clicking the Variable status icon on the toolbar or pressing F10 .
5	Double-click the variable to be forced or overwritten. Result: The Debug dialog appears.
6	In the Debug dialog, enter the desired value for a non-Boolean variable or select TRUE or FALSE for a Boolean variable.
7	Click Force or Overwrite depending on the desired operation and variable type. Result: The forcing/overwriting is applied as described at the beginning of this section. Forced variables are shown on a pink background.

Unforcing variables:

Step	Action
1	Select Debug dialog... from the context menu of the variable (in variable status). Result: The Debug dialog appears.
2	Click Reset force to unforce the selected variable. Click Reset force list to unforce each forced variable.

In debug mode, Machine Expert - Safety provides an additional debug function referred to as single cycle operation. In single cycle operation, the Safety Logic Controller interrupts the continuous cyclic processing.

Refer to the chapter *Debugging: Forcing, Overwriting, Single Cycle Operations* in the *EcoStruxure Machine Expert - Safety - User Guide* for details on forcing/overwriting and single cycle mode.

Validation and Documentation of the Safety-Related Project

General Information

After commissioning and completing the safety-related project, an acceptance procedure must be performed. Once this procedure is passed, any additional project modifications lead to further project reviews and re-validations. To help avoid this, Machine Expert - Safety provides the possibility to certify the validated project. A certified project is protected by password against particular modifications and has to be unlocked before re-editing.

Validating the Safety-Related Project

The following procedure is only possible if the safety-related project has been compiled successfully.

Defining a project as validated:

Step	Action
1	Select Project → Project Certification... Result: The Project Certification dialog opens.
2	Define a certification password by entering it twice into both input fields and confirm the dialog. Result: The project is now certified and locked. Only particular operations are possible. The validation (certification) is displayed in the status bar.

Refer to the chapter *Project Certification* in the *EcoStruxure Machine Expert - Safety - User Guide* for further information on the operations which are possible while a project is locked.

Removing the validation and re-editing:

Step	Action
1	Select Project → Project Certification... Result: The Project Certification dialog opens.
2	Enter the certification password and confirm the dialog Result: The project is unlocked for editing.

POU Verification Flag

For marking validated POU's, Machine Expert - Safety provides a POU verification flag. After verifying the code of a POU, the verification flag can be set for this particular POU by selecting the **Set verification** item from the context menu of the POU icon.

Refer to the chapter *POU Verification* in the *EcoStruxure Machine Expert - Safety - User Guide* for more information.

Project Documentation

Machine Expert - Safety helps you to document the safety-related project. For that purpose, the dialog **Project Info** is provided which you open via the **Project → Project Information** menu command.

Dialog fields with a colored header must be filled out each time a new project version is developed. Fields with a gray header are optional. However, you should enter data in the fields, even if they are optional.

The **Project** area in the **Project** dialog tab is read-only as these data are polled by Machine Expert - Safety. Some of these data can be copied to the Clipboard.

The **Project** dialog tab displays various checksums for certain parameters or data which have been calculated by Machine Expert - Safety. You can use these checksums to find out if projects differ in their parameters or data. By comparing the checksums that are calculated separately for the various parameter and data groups, you can see which parts of the projects are different.

The data in the **Checks** dialog tab are part of the acceptance test procedure.

Refer to the chapter *'Project Info' Dialog* in the *EcoStruxure Machine Expert - Safety - User Guide* for more information and detailed description of the various CRCs.

Printing the Project Documentation

After the successful commissioning of the safety-related application and editing the project documentation, it is mandatory to print the whole project. The **File** menu in Machine Expert - Safety provides commands to define printer settings, display a preview and to print the entire project or parts of it.

Refer to the chapter *Printing and Preview* in the *EcoStruxure Machine Expert - Safety - User Guide* for more information and detailed description of the various CRCs.

Section 3.3

Interaction Between the Safety Application and the Standard Application

What Is in This Section?

This section contains the following topics:

Topic	Page
Exchanging Data Between Logic/Motion Controller and Safety Logic Controller	76
Enabling a Safety-related Output via the Standard Application	78
Reading Diagnostic Signals of Safety-related Modules	80
Downloading Modified Projects to Logic/Motion Controller and SLC	82

Exchanging Data Between Logic/Motion Controller and Safety Logic Controller

General Information

The standard application (Logic/Motion Controller) and the safety-related application can directly exchange data.

When inserting an Safety Logic Controller into the Logic/Motion Controller project in EcoStruxure Machine Expert Logic Builder, a special memory area for exchange data is reserved. In this area, exchange signals are available. Which exchange data are to be used in your project must be configured in the I/O configuration of the Safety Logic Controller. Refer to the chapter *Exchange Data Configuration for the SLC* (see page 38) for details, limitations, and a step-by-step procedure.

NOTE: Exchange data between the standard and the safety-related application are always non-safety-related (standard) variables.

NOTE: In addition to the exchange signals of the Safety Logic Controller, also the TM5/TM7 I/O modules provide exchange signals. To map these signals in Logic Builder, double-click the respective TM5/TM7 module in the **Devices tree** and open the editor **TM5/TM7 Module I/O Mapping**. Refer to the section *Reading the Status of Safety-related Output Channels* (see page 80) for an example.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the relevant diagnostic process data items provided by the Safety Logic Controller and the I/O modules involved in your safety-related application are monitored and evaluated so that your standard application can determine the state of the functional safety-related system.
- Validate that the machine is set to the application-specific defined safe state (according to your risk analysis) depending on the safety-related diagnostic process data evaluation.
- Use appropriate safety interlocks where personnel and/or equipment hazards exist.
- Validate the overall safety-related function and thoroughly test the application.

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

Reading an SLC Exchange Variable in the Logic/Motion Controller

This chapter describes the possible use of an exchange variable with the data direction Safety Logic Controller to Logic/Motion Controller based on the following example:

A safety-related SF_EmergencyStop function block used in the safety-related application outputs a Boolean error flag.

The following procedures describe how to set up and configure the data exchange and read this value in the standard application thus enabling the Logic/Motion Controller to react on a function block error.

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the parameter group SLCIoConfiguration of the Safety_PLC (Devices tree) , reserve one Boolean variable group SLC2LMC_NumberOfBoolGroups as described in section Configuring the Data Exchange (<i>see page 38</i>). Result: Eight Boolean exchange variables are available in the safety-related application.
2	Open the Schneider Electric Sercos III I/O Mapping editor of the Safety_PLC and map a variable to the exchange flag. You have two possibilities: ● Enter a variable name into the Variable table cell. This way, a new global variable is declared if the entered name is not yet declared. ● Or double-click the ... icon to select an existing variable from the Input Assistant.
3	Program a read access to the mapped variable in your application code (flag evaluation). Evaluate the variable in a way that the standard application reacts on a function block error, that is, if the variable is TRUE.

Procedure in Machine Expert - Safety:

Step	Action
1	Open the code worksheet where you want to insert and write the exchange signal. In the example, this is the code where the SF_EmergencyStop function block is used.
2	In the Devices window, open the devices tree on the left and expand the SLC (SL1.SM1) tree node.
3	Drag the Boolxxx exchange variable into the code worksheet. Result: When releasing the mouse button, the Variable dialog appears.
4	In the Variable dialog, accept the proposed name, select an existing global variable, or declare a new global variable.
5	Confirm the Variable dialog by clicking OK and drop the variable at the desired position with a left click. Result: The variable is inserted into the code and its variable declaration is automatically inserted into the global variable worksheet. You can directly connect the variable to another object (for example, a formal parameter) or drop it unconnected at any free position.

Enabling a Safety-related Output via the Standard Application

General Information

In general, safety-related outputs can only be written by the Safety PLC. Depending on the setting in the respective safety-related TM5/TM7 output module (SDO), the standard controller must additionally enable the safety-related output (acknowledge the safety-related signal).

For that purpose, the **User Defined Parameters** editor of the SDO module provides one `CentralControl_DigitalOutputs_xx` parameter per output channel with two possible parameter values:

- **Direct**: switching the output channel is directly possible in the SLC (safety-related application, programmed in Machine Expert - Safety) without an acknowledgment from the standard application.
- **Central**: for switching the output channel, the standard Logic/Motion Controller application must enable (acknowledge) the safety-related signal coming from the SLC.



WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the enable signal only controls the process directly if it does not adversely affect the safety-related function.
- Verify that the `SafeDigitalOutputxx` signal is only used in the safety-related application as long as the related diagnostic signals are `SAFETRUE` if demanded by the results of your risk analysis.
- Validate the overall safety-related function, including the start-up behavior of the process, and thoroughly test the application.

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

Using an Enable Signal

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the <code>TM5SDOxxx</code> module to open the parameter editor, and click the tab User-Defined Parameters .
2	Set the parameter <code>CentralControl_DigitalOutputs_xx</code> (where <code>xx</code> is the channel number) to Central .

Step	Action
3	Open the tab TM5 Module I/O Mapping and map a variable to the SafeDigitalOutputxx signal (where xx is the channel number) which is the enable signal for this output. You have two possibilities: • Enter a variable name into the Variable table cell. This way, a new global variable is declared if the entered name is not yet declared. • Or double-click the ... icon to select an existing variable from the Input Assistant. Observe the note below the following table.
4	Use the mapped variable in your application code in such a way that it can be used to acknowledge the setting of the safety-related output.

If it is necessary to process the enable signal also in the safety-related application, you can insert it into the code as described in the following procedure.

Optional procedure in Machine Expert - Safety:

Step	Action
1	Open the code worksheet where you want to insert the enable signal.
2	In the Devices window, open the devices tree on the left and expand the SLC (SL1.SM1) tree node.
3	Drag the enable signal into the code worksheet. Result: When releasing the mouse button, the Variable dialog appears.
4	In the Variable dialog, accept the proposed name, select an existing global variable, or declare a new global variable. Result: The variable is inserted into the code and its variable declaration is automatically inserted into the global variable worksheet.
5	Confirm the Variable dialog by clicking OK and drop the variable at the desired position with a left click. Observe the note below this table.

NOTE: The column **LogicBuilder Variable** shows the name of the variable which has been mapped to the enable signal in the **TM5 Module I/O Mapping** editor in Logic Builder. This representation in the Machine Expert - Safety **Devices** window must not be misinterpreted: Although the enable signal (**ChannelName**), the safety-related **Variable** name, and the **LogicBuilder Variable** are shown in one row, the **LogicBuilder Variable** cannot write the safety-related output. The LogicBuilder variable only agrees to the activation of the output. The physical activation of the output, however, can exclusively be initiated by the SLC.

Reading Diagnostic Signals of Safety-related Modules

General Information

The standard application (Logic/Motion Controller) and the safety-related application can directly communicate.

Beside the Safety Logic Controller exchange variables (see section Exchanging Data between Logic/Motion Controller and Safety Logic Controller (*see page 76*)), also the TM5/TM7 I/O modules provide diagnostic exchange signals.

After you have inserted a safety-related I/O module into your bus architecture (**Devices tree** in Logic Builder), and then confirmed the modified bus configuration in Machine Expert - Safety, these diagnostic signals are available in the **Devices** window.

NOTE: These signals are diagnostic signals for evaluation purposes in the standard application. They have no impact on the safety function. Diagnostic exchange signals always have a standard data type.

WARNING

UNINTENDED EQUIPMENT OPERATION

- Verify that the relevant diagnostic process data items provided by the Safety Logic Controller and the I/O modules involved in your safety-related application are monitored and evaluated so that your standard application can determine the state of the functional safety-related system.
- Validate that the machine is set to the application-specific defined safe state (according to your risk analysis) depending on the safety-related diagnostic process data evaluation.
- Use appropriate safety interlocks where personnel and/or equipment hazards exist.
- Validate the overall safety-related function and thoroughly test the application.

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

Reading the Physical State of a Safety-related Output Channel

This chapter describes the use of a diagnostic signal of a safety-related TM5/TM7 I/O module based on an application example: A safety-related TM5 output module provides one diagnostic signal per channel which mirrors the physical state of the safety-related output. These signals can be read in the standard Logic/Motion Controller application.

The following procedure describes how to map this signal in the standard application thus enabling the Logic/Motion Controller to react on the physical state of the safety-related output channel.

Procedure in EcoStruxure Machine Expert Logic Builder:

Step	Action
1	In the Devices tree , double-click the safety-related output module of which you want to read the output channel status.

Step	Action
2	Open the TM5 Module I/O Mapping editor.
3	In the signal table, locate the signal <code>PhysicalStateChannelxx</code> , where <code>xx</code> is the channel number. This signal is in the <code>Status</code> signal group.
4	Map a variable. You have two possibilities: ● Enter a variable name into the Variable table cell. This way, a new global variable is declared if the entered name is not yet declared. ● Or double-click the ... icon to select an existing variable from the Input Assistant.
5	Program a read access to the mapped variable in your application code (flag evaluation) and evaluate the variable in a way that the standard application can react on an undesired or unintended output channel status.

Downloading Modified Projects to Logic/Motion Controller and SLC

Downloading the Changed Projects to the Controllers

After you have modified your standard project and your safety-related project as described in this chapter, you must compile the projects in EcoStruxure Machine Expert Logic Builder and Machine Expert - Safety.

After you have compiled the projects without errors, you must update the configurations of both the Logic/Motion Controller and the Safety Logic Controller.

Proceed as described in the chapters

- *Connecting and Downloading to the Logic/Motion Controller (see page 44)*
- *Downloading and Starting the Safety Application (see page 67)*

Chapter 4

Operation and Maintenance of the Integrated Application

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
System Start-up	84
Monitoring the Safety Application in Logic Builder	85
Remote Controlling of the SLC	88
Sercos Diagnostics:	89

System Start-up

Best Practice for Sercos Phase-up

During the boot-up of the safety-related system, the SLC sends a large amount of process data through the asynchronous Sercos channel in order to configure the connected safety nodes. An overload of the asynchronous Sercos may result in problems for the SLC to detect the safety-related modules.

To enlarge the SERCOS III bandwidth of its asynchronous channel, stop the Sercos phase-up during phase 2 until the SLC has finished the scanning of the system. Then continue the Sercos phase-up to phase 4.

For that purpose, use the **Schneider Electric Sercos III parameters** editor. (Double-click the **Sercos_Master** in the **Devices tree** in Logic Builder to open this editor.) The **DesiredPhase** parameter allows you to control the Sercos phase-up and **ActualValue** indicates the current phase.

Open the **VIS_SlcRemoteController** visualization in your example (via the **Applications tree** in Logic Builder) to verify if the SLC has finished the scanning of the system.

Monitoring the Safety Application in Logic Builder

Online Object Editors

You can use the object editors of the devices involved to display online values of device objects while Logic Builder is connected to the running Logic/Motion Controller.

Displaying the Object Status in Logic Builder:

Step	Action
1	Select Online → Login or click the Login command on the main toolbar or press Alt + F8 .
2	If the application is not running, start the application execution by selecting the Debug → Start command or pressing F5 .
3	Double-click the Safety_PLC node in the Devices tree.
4	Click the Schneider Electric Sercos III parameters tab to display the online values of the SLC objects in the Current Value column.
5	Expand the desired parameter group in the grid, for example, SLCProjectInformation or SLCProjectStatus, and so on.
6	Perform the steps 1 to 4 appropriately for the safety-related and standard TM5/TM7 I/O modules to display their object values.

SLC-related information can also be displayed in the **SafePLC Info** dialog. You can open this dialog from the **SafePLC** control dialog in Machine Expert - Safety.

NOTE: The ProjectTime displayed in Logic Builder may differ from the time shown in the **SafePLC Info** dialog in Machine Expert - Safety. Refer to section *System-specific Notes (see page 87)* for details.

Online Mapping Editors Display Variable Values

You can use the mapping editors of the SLC and the I/O modules involved to display online values of variables and signals while Logic Builder is connected to the running Logic/Motion Controller.

Displaying the Variable/Signal Status in Logic Builder:

Step	Action
1	Select Online → Login or click the Login command on the main toolbar or press Alt + F8 .
2	If the application is not running, start the application execution by selecting the Debug → Start command or pressing F5 .
3	Double-click the Safety_PLC node in the Devices tree.
4	Click the Schneider Electric Sercos III I/O Mapping tab to display the online values of signals in the Current Values column.
5	Expand the desired parameter group in the grid, for example, SLC2LMC_B00L.
6	Perform the steps 1 to 4 appropriately for the safety-related and standard TM5/TM7 I/O modules to display their variable/signal values.

SafeLogger

The SafeLogger in EcoStruxure Machine Expert collects messages which are generated by the safety-related system objects (openSafety messages) and transfers them over the Sercos bus. These messages provide diagnostic information which can be used for troubleshooting purposes. In the SafeLogger, the messages are shown with a time stamp. Corresponding managing functions for handling the messages are provided.

Refer to the section *System-specific Notes* (see page 87) for details regarding the SafeLogger entries.

The messages are classified according to three message types:

- Information messages, for example, status information
- Alert messages (yellow)
- Error messages (red)

For detailed information on the SafeLogger, its configuration and the handling of messages, refer to the *SafeLogger User Guide*.

Opening the SafeLogger in EcoStruxure Machine Expert:

Step	Action
1	Double-click the Logic/Motion Controller in the Devices tree.
2	Click the SafeLogger tab in the editor area.

SafeLogger Entries Merged to GlobalLogger

Entries shown in the SafeLogger can be merged into the GlobalLogger. If no GlobalLogger is available, proceed as follows:

Procedure in EcoStruxure Machine Expert while logged off from the Logic/Motion Controller:

Step	Action
1	In the Devices tree, right-click the MyController node and select Add Object → GlobalLogger from the context menu.
2	Enter a name for the new GlobalLogger and click Add . Result: The new logger is inserted under the MyController node and opened in the editor area.
3	In the GlobalLogger window, click the Get messages from Logger command and select the Safe Logger check box to include its entries into the GlobalLogger.

System-specific Notes

Observe the following notes regarding the entries in the SafeLogger when used in the specific architecture of a Logic/Motion Controller with Embedded Safety:

- **Differing time stamps:** The time stamps displayed in Logic Builder (for example, in the SafeLogger or for the ProjectTime parameter in the SLC **Schneider Electric Sercos III Parameters**) may differ from the time stamps shown in Machine Expert - Safety (for example, in the **SafePLC Info** or **Project Info** dialog).
The reason is that the tools may use different time bases. Machine Expert - Safety always converts time stamps to local time while the SafeLogger uses the setting in the **Services** editor for the Logic/Motion Controller. (The time setting is written to the slaves during Sercos phase-up.) Refer to the chapter Services (**M262 Logic/Motion Controller Programming Guide**) for details. In the object editors, always UTC-based time stamps are shown.
- **Differing project CRC:** The ProjectCRC parameter is shown as decimal value in the object editor of the SLC (Logic Builder) and possibly as hexadecimal value in Machine Expert - Safety. The format used in Logic Builder can be set via the **Display Mode** context menu of the **VIS_SlcRemotController** visualization page (while in online mode).
- **Topological Station ID:** In the SafeLogger, the station ID is displayed as topological path. TopoAdr : 1 identifies the first slave node under the Sercos Master. The node itself, for example, the SLC or a bus coupler, has StructureInstance : 0. TM5/TM7 I/O modules under the bus coupler follow with a StructureInstance > 0.

Remote Controlling of the SLC

General Information

As you have created your project based on the **SLC Remote Controller (M262)** example, it provides a prepared visualization example of a virtual Safety Logic Controller control center.

The visualization page **VIS_SlcRemoteController** enables you to:

- Log on to the SLC and modify its password.
- Visualize the status of the SLC and the connected safety-related I/O modules.
Some of this information can also be displayed in the **SafePLC Info** dialog which you can open from the **SafePLC** control dialog in Machine Expert - Safety.
- Execute the **Application Download** command.
- Execute SafeKey-related commands, such as formatting or copying the SafeKey, or confirming the key exchange (instead of acknowledging the exchange directly at the SLC device).
- Execute configuration-related commands regarding the connected safety-related modules (confirm firmware update or module exchange, system scan).

Using the Remote Control Visualization

Procedure in Logic Builder:

Step	Action
1	Log in to the Logic/Motion Controller (Online → Login). Result: Open editors are switched to online mode.
2	If the application is not running, start the application execution by selecting the Debug → Start command or pressing F5 .
3	Open the Tools tree and expand the Application (...) folder.
4	Expand the folder SlcRemoteControllerExample and double-click the VIS_SlcRemotController node. Result: The visualization worksheet opens showing the virtual SLC control center.
5	In the virtual SLC control center, click the Enable_Vis button and then the Remote Control button.
6	Enter the SLC password into the field below the Remote Control button and press Enter . Result: The data shown in the visualization screen are read from the SLC and displayed on your screen.

If you have executed the Application Download command via the visualization, you must confirm the project CRC. Machine Expert - Safety shows this CRC as hexadecimal value (for example, in the **SafePLC Info** dialog). The visualization in Logic Builder may expect a decimal value. You can set format via **Display Mode** context menu of the visualization page.

Refer to the *EcoStruxure Machine Expert - Safety Programming Guide* for information on the Application Download feature.

Sercos Diagnostics:

Evaluating the ConnectionState Parameter of Sercos Slaves

In case of an interrupted Sercos bus, the Sercos Master in the Logic/Motion Controller remains in phase CP4 (except if the connection is interrupted between the Sercos Master and the first slave). This occurs although some Sercos slaves are no longer reachable.

In this scenario, the values of the suspended Sercos slaves, which are shown in the respective object editors in Logic Builder, are frozen to the last updated value and are outdated.

To help prevent that such outdated values are used in the application, evaluate the `ConnectionState` parameter of each Sercos slave involved.

The SLC provides this parameter in the **Schneider Electric Sercos III Parameters** editor, group `Sercos Diagnostics`. For other slaves, such as the BC_TM5NS31, this parameter is available in the **Device Parameters** editor.

The value indicates the actual state of the Sercos connection to the particular slave. Use this parameter in your code in such a way that any values delivered from this slave are only considered as valid if `ConnectionState = Operational`.



A

- activating licenses, *28, 47*
- acyclic processing steps per SLC cycle, *50*
- Application Download, *88*
- application start-up, *44*
- architecture
 - embedded safety, *16*
 - Sercos line, *25*
 - TM5 cycle time with many modules, *42*

B

- best practice
 - boot-up time of system, *51*
 - response time relevant parameters, SLC, *52*
 - Sercos cycle time, *36*
 - Sercos phase-up, *84*
 - SLC cycle time, *49*
 - TM5 cycle time, *42*
- boot-up time of safety system, *50*
- bus coupler
 - configuration, *42*
 - in sample application, *17*

C

- calculating
 - response time relevant parameters, *52*
 - safety response time, *57*
- CentralControl_DigitalOutputs_xx, *41*
- checksum, *87*
- commissioning
 - Logic/Motion Controller, *44*
 - Safety Logic Controller, *65*
- commissioning PC wiring, *25*
- communication path, SLC, *49*
- communication settings, *35*
- CommunicationWatchdog, *52, 54*
- compatible Logic/Motion Controller types, *18*

- compatible projects, *18*
- compiling safety-related project, *63*
- configuration
 - Ethernet 1 port, *35*
 - exchange data of the SLC, *38*
 - Safety Logic Controller, *38*
 - Sercos cycle time, *36*
 - SLC, *49*
 - task settings, *35*
 - TM5 cycle time, *42*
 - TM5/TM7 modules, *41, 54*
 - TM5NS31 bus coupler, *42*
- ConfiguredSercosAddress, *37*
- ConfiguredTopologicalAddress, *37*
- confirming changed SDIO devices, *47*
- connecting
 - Logic/Motion Controller, *44*
 - Safety Logic Controller, *67*
- ConnectionState, *89*
- Controller Assistant, *27*
- Controller Assistant software, *29*
- CRC, *87*
- creating projects, *33*
- CycleTime
 - SLC, *49*
 - TM5 bus, *42*

D

- data telegram, timing validation, *52, 54*
- debug mode (SLC), *66, 68*
- debug watchdog (SLC), *66*
- debugging, *70*
- declaring variables in EcoStruxure Machine Expert - Safety, *62*
- DesiredPhase, *45*
- Development level in EcoStruxure Machine Expert - Safety, *47*
- Device Assistant software, *27, 29*
- Device Parameterization Editor, *48, 54*

- device parameters
 - in EcoStruxure Machine Expert - Safety, 48
 - Safety Logic Controller, 38, 49
 - TM5/TM7 modules, 41, 54
 - TN5NS31 bus coupler, 42
- device synchronization, 47
- Devices tree, 33
- devices tree in EcoStruxure Machine Expert - Safety, 48
- devices used in example project, 17
- Devices window, 48, 54
- diagnostics, 89
- dialog
 - Confirm changed SDIO Devices, 47
 - login dialog for SLC, 67
 - Project Info, 74
 - Response Time Calculator, 57
 - Response Time Relevant Parameters, 52
 - SafePLC, 66, 67
 - SafePLC Info, 88
- documentation, safety-related project, 74
- downloading
 - Logic/Motion Controller, 44
 - Safety Logic Controller, 67

E

- EcoStruxure Machine Expert, 17
 - activating license, 28, 47
 - components required, 27
 - installation, 27
- EcoStruxure Machine Expert - Safety, 17
 - Devices window, 48
 - first steps, 47
 - launching, 47
 - passwords, 47
 - safety-related device parameters, 48
- EcoStruxure Machine Expert Logic Builder, 17
- electrical installation, 23
- ElectronicLabel, 45
- embedded safety architecture, 16
- EN ISO 13849, 20
- enabling safety output via Logic/Motion Con-

- troller, 78
- Ethernet 1 port, 25, 35
- Ethernet 2 port, 25
- Ethernet_1 (node in Devices tree), 33
- example project, 17
- exchange data, 48, 76
 - configuration for SLC, 38
 - mapping in standard application, 39
 - rules and notes, 39

F

- fieldbus wiring, 25
- firmware updates, 29
- firmware updates confirmation, 30
- forcing, 70
- functional testing, 68
- FW-ACKN, 30

G

- GlobalLogger, 86
- guidelines for wiring, 24

H

- HALT [Debug], SLC state, 67

I

- IdentificationMode, 37
- IEC 61508, 20
- input transport time, 56
- installation
 - Controller Assistant software, 27
 - Device Assistant software, 27
 - electrical, 23
 - mechanical, 22
 - software, 27
 - software for firmware update, 27
- interval of task, 35
- IP address, 44
 - Logic/Motion Controller, 35
- IPConfigMode, 37

L

launching EcoStruxure Machine Expert - Safety, 47
 license activation, 28, 47
 License Manager, 28
 limitations
 exchange data (amount of), 38
 system, 18
 LMC2SLC_NumberOfxxx, 38
 Logic/Motion Controller, 16
 commissioning, 44
 connecting to, 44
 downloading project, 44
 Ethernet port configuration, 35
 firmware updates, 29
 IP address, 35
 SERCOS CN1 port, 25
 Sercos cycle time, 36
 task configuration, 35

M

Machine Expert Installer, 27
 maintenance, 83
 ManualConfiguration, 54
 MaxDataTransportTime, 52, 54
 mechanical installation, 22
 message categories, SafeLogger, 86
 migration from PacDrive 3 system, 18
 MinDataTransportTime, 52, 54
 minimum distance of safety equipment, 55
 module scans, no. of, 51
 monitoring, 68, 69
 online object editors, 85
 online variable editors, 85
 SafeLogger, 86
 Sercos communication state, 89
 MXCHG flashing, 51

N

Network Package Loss, 52
 No Execution, SLC state, 66
 NodeGuardingTimeout, 51
 NumberOfScans, 51

O

online mode, 69
 online values
 EcoStruxure Machine Expert Logic Builder, 85, 85
 openSafety protocol, 17
 operating modes
 Safety Logic Controller, 65
 operation, 83
 Optional (parameter), 54
 output transport time, 56
 overwriting, 70

P

password
 Safety Logic Controller, 65, 67
 safety-related project, 47
 PC wiring, 25
 performance, 36
 TM5 cycle time, 42
 Performance Level, 20
 phase-up, 44, 51, 84
 PL, 20
 planning of safety equipment, 55
 Post Configuration Warning, 44
 POU verification flag, 73
 printing, safety-related project, 74
 processing time
 application in SLC, 56
 in safety input modules, 56
 in safety output module, 56
 in safety output modules, 57
 in sensor, 56
 Program Machine controllers (Modicon), software component, 27
 Program Machine Safety, software component, 27
 programming
 connecting objects in safety-related code, 62
 constants in safety-related code, 61
 declaring safety-related variable for de-

- vice signals, *62*
- device signals in safety-related code, *60*
- FUs/FBs in safety-related code, *60*
- mapping device signals to variables, *63*
- safety-related code, *58*
- variables in safety-related code, *62*

project

- compatibility, *18*
- compiling (safety-related), *63*
- CRC, *87*
- creation in EcoStruxure Machine Expert Logic Builder, *33*
- description of example, *33*
- documentation, *74*
- downloading to Logic/Motion Controller, *44*
- downloading to SLC, *67*
- from example, *33*
- information (dialog), *74*
- passwords for safety-related, *47*
- printing, *74*
- validation, *73*

R

reading

- diagnostic signals of safety modules, *80*
- exchange variable in Logic/Motion Controller, *76*

register, *27*

registering, *47*

- remote controlling, *34, 51, 88*

- RemoteControlAllowed, *51*

- Response Time Calculator, *57*

- response time relevant parameters, *52, 54*

- RUN [Debug], SLC state, *67*

- RUN [Safe], SLC state, *66*

S

- safe mode (SLC), *65*

- SafeKey, *88*

- SafeLogger, *86*

- SafeLogicType, *45*

safety application

- debugging, *70*

- example project, *58*

- functional testing, *68*

- monitoring, *68, 69*

- safety equipment, *55*

- Safety Integrity Level, *20*

- Safety Logic Controller, *16*

- allowing remote controlling, *51*

- commissioning, *65*

- communication path, *49*

- configuration, *38, 49*

- confirming firmware updates of modules, *30*

- connecting to, *67*

- cycle time, *49*

- downloading project, *67*

- firmware updates, *29*

- firmware version, verifying, *45*

- FW-ACKN, *30*

- in devices tree, *48*

- MXCHG flashing, *51*

- NodeGuardingTimeout, *51*

- NumberOfScans, *51*

- operating modes, *65*

- password, *65*

- remote controlling, *34*

- RemoteControlAllowed, *51*

- response time relevant parameters, *52*

- scanning modules, *51*

- Sercos III RJ45 ports, *25*

- SSDIOCreation, *50*

- state machine, *66*

- timing validation, *52*

- type, *45*

- visualization of states, *34*

- safety nodes, *17*

- Safety PLC, *45*

- safety response time

- calculation, *57*

- preconditions for calculation, *55*

- recalculate after cycle time modification, *42*

- relevant parameters, *52, 54*

- TM5 cycle time, influence, *42, 52*

Safety_PLC, *48*
 safety-related code, *58*
 safety-related parameters, *48, 49*
 SafetyResponseTimeDefaults (SLC), *52*
 scanning the safety network, *51*
 Schneider Electric Sercos III parameters, *45*
 Schneider Electric Sercos III Parameters, *48*
 Sercos, *67, 68*
 address assignment, *37, 44*
 asynchronous communication, *36, 84*
 cycle time, *36*
 diagnostics, *89*
 phase-up, *36, 37, 44, 51, 84*
 Sercos III RJ45 ports (SLC), *25*
 Sercos wiring, *25*
 Sercos_Master, *17, 33, 36, 45, 48, 84*
 SercosCycletimeConfig, *36*
 SercosPhaseChanger, *45*
 SIL, *20*
 single-line architecture, *25*
 SLC
 remote controlling, *88*
 SLC connected through LMC, *49*
 SLC Remote Controller (M262) (project example), *33*
 SLC2LMC_NumberOfxxx, *38*
 SlcIoConfiguration, *38*
 SlcRelatedConfiguration, *45*
 SlcRemoteControllerExample, *33*
 SN, *17*
 software installation, *27*
 SoftwareRevision, *45*
 SSDIOCreation, *50*
 standard (term definition), *16*
 start-up, *44, 84*
 state machine (SLC), *66*
 STOP [Debug], SLC state, *66*
 STOP [Safe], SLC state, *66*
 synchronization of safety-related devices, *47*

system
 architecture, *16*
 boot-up time, *50*
 compatibility, *18*
 limitations, *18*
 maintenance, *83*
 operation, *83*
 performance, *36, 42*
 SIL/PL, achievable, *20*
 start-up, *84*

T

task
 configuration, *33*
 configuration of Logic/Motion Controller, *35*
 safety-related, *58*
 TASK_SR_VisControl, *35*
 Task_SR_VisControl (example task), *33*
 TCP/IP Communication parameters, *49*
 time stamps, *87*
 time-limited trial version, *27*
 timeout
 debug mode of SLC, *66*
 in safety-related communication, *52*
 while scanning safety devices, *51*
 timing validation of safety application, *52, 54*
 TM5 bus cycle time, *42*
 TM5/TM7 devices
 firmware updates, *30*
 TM5/TM7 modules
 configuration, *41, 54*
 confirming changed architecture, *47*
 device synchronization, *47*
 enabling safety output via Logic/Motion Controller, *78*
 firmware updates, *29*
 ManualConfiguration, *54*
 processing time, *56*
 reading diagnostic signals, *80*
 response time relevant parameters, *54*
 timing validation, *54*
 visualization of states, *34*

TM5NS31 bus coupler
 configuration, *42*
tools
 activating license, *28, 47*
 for embedding safety, *17*
 for firmware updates, *27, 29*
 License Manager, *28*
Tools tree, *34, 88*
Topology mode, *37*
total safety response time, *56*
trial version, time-limited, *27*

U

user-defined parameters, *41*

V

validation, safety-related project, *73*
variable status, *69*
VIS_SlcRemoteController, *34, 88*
visualization (example), *34, 88*

W

watchdog
 for debug mode of SLC, *66*
 for safety-related communication, *52*
wiring
 guidelines, *24*
 Sercos, *25*
writing values, *45*

Z

zone of operation, *55*