

# Modicon M580

## BMECXM CANopen Modules

### User Manual

Original instructions

EIO0000002129.06

11/2023

# Legal Information

The information provided in this document contains general descriptions, technical characteristics and/or recommendations related to products/solutions.

This document is not intended as a substitute for a detailed study or operational and site-specific development or schematic plan. It is not to be used for determining suitability or reliability of the products/solutions for specific user applications. It is the duty of any such user to perform or have any professional expert of its choice (integrator, specifier or the like) perform the appropriate and comprehensive risk analysis, evaluation and testing of the products/solutions with respect to the relevant specific application or use thereof.

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

This document and its content are protected under applicable copyright laws and provided for informative use only. No part of this document 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 document or its content, except for a non-exclusive and personal license to consult it on an "as is" basis.

Schneider Electric reserves the right to make changes or updates with respect to or in the content of this document or the format thereof, at any time 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 document, as well as any non-intended use or misuse of the content thereof.**

# Table of Contents

Safety Information .....	7
Before You Begin.....	8
Start-up and Test.....	9
Operation and Adjustments .....	10
About the Book .....	11
<b>CANopen Hardware Implementation.....</b>	<b>13</b>
Generalities .....	14
Module Description.....	14
Communication Profile.....	19
Key Features of BMECXM Modules .....	21
BMECXM Module Installation and Replacement.....	25
BMECXM Module Installation .....	25
BMECXM Module Replacement .....	28
<b>CANopen Software Implementation .....</b>	<b>31</b>
Generalities .....	32
Implementation Presentation .....	32
Maximum Configuration .....	34
Device PDO Allocation.....	36
Performance .....	38
Operating Modes.....	40
Fallback Strategy.....	44
CANopen Configuration .....	47
Overview .....	47
Overview.....	47
Adding a CANopen X80 Master BMECXM module.....	50
Adding a CANopen X80 Master BMECXM Module .....	50
Bus Configuration.....	52
Access the CANopen Bus Editor .....	52
Adding Slave Devices on the CANopen Bus .....	54
Delete/Move/Duplicate a Device on the CANopen Bus .....	58
View CANopen Bus in the Project Browser .....	59
Device Configuration .....	60

Presentation of CANopen Devices .....	61
Slave Functions .....	62
Configuration Using Control Expert .....	65
Configuration Using an External Tool .....	75
Master Configuration .....	77
CANopen Master Module Configuration Window .....	77
CANopen Master Port Configuration Screen .....	79
Ethernet Services Configuration .....	84
DTM Browser .....	84
DTM User Interface .....	87
<b>Ethernet IO</b> Tab .....	92
<b>Security</b> Tab .....	96
<b>SNMP</b> Tab .....	98
<b>NTP</b> Tab .....	100
Language Objects .....	101
Implicit Process Data Exchange .....	101
Device DDT Variables .....	102
Programming .....	104
Network Management Services .....	104
Exchanges Using SDOs .....	108
READ_SDO: Reading Service Data Object .....	109
WRITE_SDO: Writing Service Data Object .....	112
Function Block Examples .....	116
Diagnostics .....	118
LED Diagnostic .....	119
Device DDT for BMECXM Modules .....	124
Device DDT for CANopen Slave Devices .....	127
BMECXM DTM Diagnosis .....	128
Sending Explicit Messages to the BMECXM Module .....	132
Embedded Web Pages .....	134
Emergency Objects .....	140
Firmware Upgrade .....	145
Firmware Update with Automation Device Maintenance .....	145
Firmware Update with Unity Loader .....	145
Appendices .....	147

---

CANopen Master Local Object Dictionary Entry .....	148
Object Dictionary Entries According to Profile DS301 .....	148
Object Dictionary Entries According to Profile DS302 .....	151
BMECXM Manufacturer Specific Object Dictionary Entries .....	153
CANopen Commands.....	156
CANopen SDO Commands .....	156
CANopen SDO Abort Code .....	158
CANopen Start Command.....	160
CANopen Slave Enabling Command.....	161
CIP Objects .....	162
<i>DIAG_FXM_Diagnostic</i> Object .....	162
<i>DIAG_CXM</i> Object .....	167
EIP Interface Diagnostic Object .....	170
I/O Connection Diagnostics Object.....	173
EtherNet/IP Explicit Connection Diagnostic Object .....	175
Glossary .....	177
Index.....	185



# 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:

(In case of divergence or contradiction between any translation and the English original, the original text in the English language will prevail.)

- 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 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 manual describes the implementation of a CANopen fieldbus on the Modicon M580 range.

**NOTE:** Regarding Safety considerations, “Emergency objects” and “Fatal error” are mentioned in this manual in conformance with the definition from the DS301 document of the CiA (CAN in Automation).

## Validity Note

This documentation is valid for EcoStruxure™ Control Expert 15.0 or later.

The characteristics of the products described in this document are intended to match the characteristics that are available on [www.se.com](http://www.se.com). As part of our corporate strategy for constant improvement, we may revise the content over time to enhance clarity and accuracy. If you see a difference between the characteristics in this document and the characteristics on [www.se.com](http://www.se.com), consider [www.se.com](http://www.se.com) to contain the latest information.

## Related Documents

Title of Documentation	Reference Number
Modicon M580, Hardware, Reference Manual	EIO0000001578 (English), EIO0000001579 (French), EIO0000001580 (German), EIO0000001582 (Italian), EIO0000001581 (Spanish), EIO0000001583 (Chinese)
Modicon M580 Standalone, System Planning Guide for Frequently Used Architectures	HRB62666 (English), HRB65318 (French), HRB65319 (German), HRB65320 (Italian), HRB65321 (Spanish), HRB65322 (Chinese)
EcoStruxure™ Control Expert, Hardware Catalog Manager, Operation Guide	EIO0000002141 (ENG) EIO0000002142 (FRE) EIO0000002143 (GER) EIO0000002144 (ITA) EIO0000002145 (SPA) EIO0000002146 (CHS).
Modicon Controllers Platform Cyber Security, Reference Manual	EIO0000001999 (English), EIO0000002001 (French), EIO0000002000 (German), EIO0000002002 (Italian), EIO0000002003 (Spanish), EIO0000002004 (Chinese)
CANopen, Hardware Setup Manual	35010857 (ENG) 35010859 (FRE) 35010858 (GER)

Title of Documentation	Reference Number
	35010861 (ITA) 35010860 (SPA) 33004206 (CHS).
Electrical installation guide	EIGED306001EN (English)
EcoStruxure™ Control Expert, Communication, Block Library	33002527 (English), 33002528 (French), 33002529 (German), 33003682 (Italian), 33002530 (Spanish), 33003683 (Chinese)
EcoStruxure™ Automation Device Maintenance, User Guide	EIO0000004033 (English), EIO0000004048 (French), EIO0000004046 (German), EIO0000004049 (Italian), EIO0000004047 (Spanish), EIO0000004050 (Chinese)
Unity Loader, User Guide	33003805 (English), 33003806 (French), 33003807 (German), 33003809 (Italian), 33003808 (Spanish), 33003810 (Chinese)
EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual	35006144 (English), 35006145 (French), 35006146 (German), 35013361 (Italian), 35006147 (Spanish), 35013362 (Chinese)

You can download these technical publications and other technical information from our website at [www.se.com/ww/en/download/](http://www.se.com/ww/en/download/) .

## Product Related Information

### **⚠ WARNING**

#### **UNINTENDED EQUIPMENT OPERATION**

- The application of this product requires expertise in the design and programming of control systems. Only persons with such expertise should be allowed to program, install, alter, and apply this product.
- Follow all local and national safety codes and standards.

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

---

# CANopen Hardware Implementation

## What's in This Part

Generalities ..... 14  
BMECXM Module Installation and Replacement..... 25

## Subject of This Part

This part describes the various hardware configuration possibilities of a CANopen bus architecture.

# Generalities

## What's in This Chapter

Module Description .....	14
Communication Profile .....	19
Key Features of BMECXM Modules .....	21

## Introduction

This chapter presents the BMECXM modules equipped with a CANopen port.

## Module Description

### Overview

The CANopen X80 master modules (BMECXM) provide access to CANopen bus on a M580 PAC.

### Ruggedized Version

The BMECXM0100H (hardened) equipment is the ruggedized version of the BMECXM0100 (standard) equipment. It can be used at extended temperatures and in harsh chemical environments.

For more information, refer to chapter *Installation in More Severe Environments* (see Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications).

**NOTE:** In Control Expert, there is only one device part number BME CXM 0100 in the **Hardware Catalog** to declare and configure both CANopen X80 master module references.

### Altitude Operating Conditions

The characteristics in the tables below apply to the modules BMECXM0100 and BMECXM0100H for use at altitude up to 2000 m (6560 ft). When the modules operate above 2000 m (6560 ft), apply additional derating.

For detailed information, refer to chapter Operating and Storage Conditions (see Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications).

## Operating Temperature

### ⚠ WARNING

#### UNINTENDED EQUIPMENT OPERATION

Do not operate this equipment outside of its specified temperature range.

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

Module Reference	Range
BMECXM0100	0...60 °C (32...140 °F)
BMECXM0100H	-25...70 °C (-13...158 °F)

## CAN Characteristics

Characteristics		Description
Protocol supported		CANopen
Connection type		Sub-D 9, male
Standard		CANopen CiA 301 V4.2
Maximum cable length		Refer to chapter <i>Transmission Speed and Cable Length</i> (see CANopen, Hardware Setup Manual).
Isolation between CAN bus and ground		500 Vac RMS, 700 Vdc
CAN bus transmission baud rate (kbd)		20, 50, 125, 250, 500, 1000
CANopen slave devices supported		63 maximum
Services	NMT	NMT master according to DS 301
		Boot-up procedure according to DS 302
	SDO	1 SDO client 1 SDO server
	PDO	256 PDOs IN and 256 PDOs OUT

Characteristics		Description
	SYNC	Producer
	Emergency message	Consumer only
	Health	Heartbeat <ul style="list-style-type: none"><li>• 1 producer</li><li>• 63 consumers</li></ul>
		Node Guarding

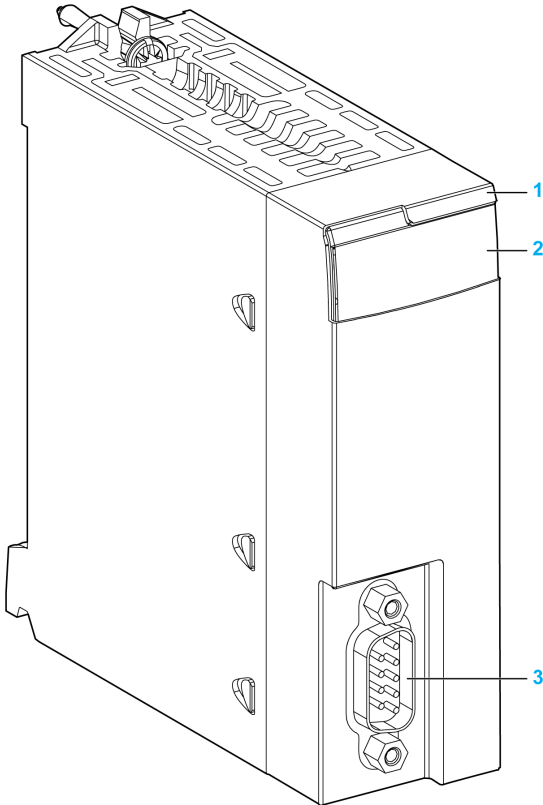
The BMECXM modules comply with the relevant standards and rules for electrical equipment in an industrial automation environment.

For details, refer to chapter *Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications*.



## Physical Description

This figure shows the external features of the module:



Number	Element	Function
1	Module name	BMECXM0100 or BMECXM0100H
2	LED array	Observe the LED display to diagnose the module.
3	SUB-D 9 connector	CANopen port

## LEDs

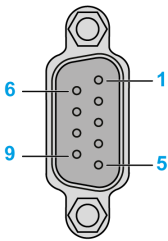
The LED display on the front of the module gives information about module operating status and about CANopen communication status:

LED	Color	Description
<b>RUN</b>	Green	Indicates module operating status.
<b>ERR</b>	Red	Detected error in module operation.
<b>I/O</b>	Red	Indicate exchange status with CANopen devices.
<b>BS (Bus Status)</b>	Red / green	Indicates the EtherNet/IP connection status.
	Yellow	Firmware upgrade in progress
<b>CAN RUN</b>	Green	Indicates the status of the CANopen fieldbus.
<b>CAN ERR</b>	Red	Indicates the status of the CANopen physical layer and indicates detected errors due to missing CAN messages (SYNC, node-guarding or heartbeat)
<b>CAN COM</b>	Yellow	Dedicated to SDO transmission

**NOTE:** Refer to the section **LED diagnostics**, page 119 for information describing how to use the LEDs to diagnose the state of the module and the CANopen operations.

## CANopen Connector

The figure and table below give the CANopen connector pin assignment:



Pin	Signal	Description
1	–	Reserved
2	CAN_L	CAN_L bus line (Low)
3	CAN_GND	CAN ground
4	–	Reserved
5	CAN_SHLD	CAN shield
6	CAN_GND	CAN ground
7	CAN_H	CAN_H bus line (High)

Pin	Signal	Description
8	–	Reserved
9	Reserved	CAN external power supply that is dedicated to the optocouplers power and transmitters-receivers (optional).

## Backplane Connector Connection

The Ethernet bus interface at the back of the BMECXM module connects to the Ethernet backplane connector when you mount the module in the rack, page 25.

The module is powered by the backplane. It is hot swappable, that is, it may be installed and uninstalled without turning off the power supply to the rack.

The X Bus connector of the backplane is not used by the module as it is an Ethernet only module.

The module uses the Ethernet bus on the Ethernet backplane to manage the connectivity to the Ethernet I/O scanner.

The module can be managed by:

- The Ethernet RIO scanner of the CPU.
- The Ethernet DIO scanner of the CPU.

The module communicates with a PC that is connected to the Ethernet network using an asset management, a network manager, or a web browser.

## Communication Profile

### Overview

Two different M580 CPUs exist:

- BMEP58••40 including one RIO scanner and one DIO scanner.
- BMEP58••20 including one DIO scanner

Depending on the level performance requested by the process, the BMECXM module can be scanned by the RIO or the DIO scanner of the M580 CPU. Both scanners use EtherNet/IP to scan the module.

In the same M580 PAC, several BMECXM modules can be connected to the same or different I/O scanner.

For detail on EtherNet/IP connection behavior when the module switched to *FALLBACK* state, refer to chapter *Fallback Strategy*, page 44.

## RIO Scanner

When the BMECXM is scanned by the RIO scanner, the main capabilities are:

- Higher performance constraints
- Highest performance expectation (bandwidth sharing and controlled cycle timing on CANopen and EtherNet/IP).
- Bandwidth controlled (RSTP)
- Timing and cycle synchronized (with either MAST task or FAST task.)
- Support up to 24 kb of IO data.
- RPI is automatically calculated by Control Expert
- Motion function blocs (MFB) are supported

## DIO Scanner

When the BMECXM is scanned by the DIO scanner, the main capabilities are:

- Lower performance constraints
- Lowest performance expectation, no constraint (no bandwidth sharing and no controlled cycle).
- No control and no synchronization.
- Can only support up to 8 kb of IO data.
- No support of motion function blocs (MFB)

## Data Consistency

Whatever the BMECXM is scanned by RIO or DIO scanner, all the variables belonging to the same slave are kept consistent, meaning exchanged in the same EtherNet/IP assembly (one for input, one for outputs).

The same EtherNet/IP assembly can contain the data of several devices, and even the data of all the devices (the complete process image).

**NOTE:** When the BMECXM is scanned by the RIO scanner, all the BMECXM process image is refreshed in the PLC scan. This service is interesting only if the process image has been refreshed on the CANopen fieldbus side during the same period.

## Key Features of BMECXM Modules

### Communication Profile

The module manages the communication with:

- The CANopen slaves according to the CiA 301 V4.2 standard
- The M580 CPU over Ethernet I/O

**NOTE:** The minimum firmware version for the M580 CPU is V2.20.

**NOTE:** Communication with Ethernet I/O scanner modules is not possible.

Two communication profiles can be used:

- Remote (RIO scanner) linked to AUX, MAST, or FAST tasks.
- Distributed (DIO scanner)

These profiles offer a total flexibility depending on the level of performance required by the process, page 38.

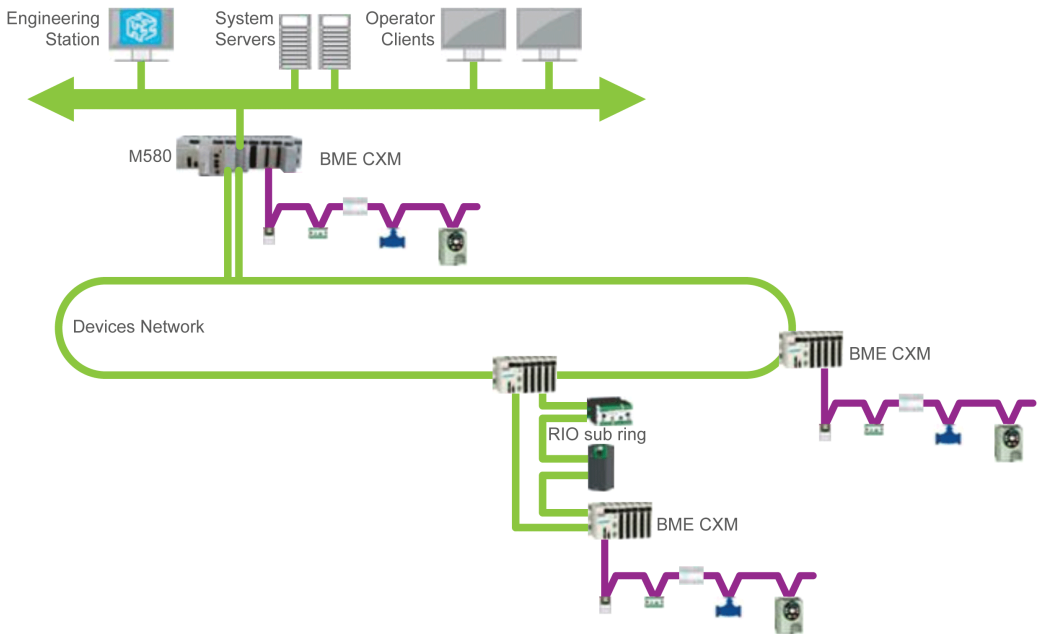
### System Architecture

The BMECXM module can be plugged in any:

- M580 local rack
- Remote drop supporting the M580 Ethernet backplane with a drop end communicator BMECRA31210 module.

**NOTE:** The minimum firmware version for the BMECRA31210 is V2.10.

You can use several CANopen X80 master modules on the M580:



## Services

Operating modes are based on the FDT/DTM technology and are fully integrated in Control Expert.

The BMECXM module can provide these services:

Configuration	
CANopen, page 47	<p>The BMECXM module is natively included in Control Expert <b>Hardware Catalog</b>.</p> <p>Use Control Expert to select and configure the BMECXM module.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>You can configure third-party CANopen slaves by using the Control Expert Hardware Catalog Manager via EDS files.</li> <li>You cannot configure a CANopen slave using its own DTM.</li> </ul>
Ethernet Services, page 84	Use Control Expert DTM to configure the BMECXM module.
Communication	

Language objects, page 101	Use Control Expert application to access CANopen slaves via device DDTs that are automatically created for each device (implicit exchanges through PDOs).
Programming, page 104	Use Control Expert application to access CANopen slaves via SDO function blocks (explicit exchanges) and device DDT (implicit exchanges).
<b>Diagnostics</b>	
LED Indicators, page 119	Use LED indicators to check the status of the BMECXM module and its communications with the network.
Device DDT, page 124	Use the device DDT to diagnose the BMECXM module from the application.
SNMP, page 98	Use SNMP services to gain easy access from an SNMP network to: <ul style="list-style-type: none"> <li>• Diagnostic information for the BMECXM module</li> <li>• Event notification for some services</li> </ul> Configure this service with the BMECXM DTM.
DTM, page 128	Use Control Expert DTM to view communication and status information of: <ul style="list-style-type: none"> <li>• the BMECXM module,</li> <li>• the CPU, and</li> <li>• the CANopen slaves.</li> </ul>
Embedded Webpages, page 134	Use the web browser of your PC to access detailed diagnostics data of the BMECXM module.
<b>Firmware upgrade</b>	
Firmware upgrade, page 145	Use the Automation Device Maintenance or Unity Loader tool to upgrade the firmware of the BMECXM module.

## Cybersecurity

The CANopen X80 master module is in line with the global policy of the Modicon M580 range in terms of cybersecurity.

Access rights between Control Expert and the BMECXM DTM are synchronized.

Webserver is in read-only.

In addition, you can restrict the BMECXM access to:

- FTP, HTTPS, SNMP, NTP, and EIP services
- Authorized IP addresses

For more information, refer to *Modicon Controllers Platform, Cybersecurity, Reference Manual*.

## Module Limits

When you use a BMECXM module on the Modicon M580, observe these limits:

- Each BMECXM module can manage a maximum of 63 CANopen slaves.
- The size of the configuration file (.prm) of each BMECXM module is limited to 64 Kb maximum.
- Each BMECXM module on the Ethernet bus communicates with the CPU. They are in the same network. Transparency is thus provided from the CPU or any PC connected on the same subnet up to the BMECXM module.

**NOTE:** No communication is possible between two BMECXM modules.

## Module Restrictions

These services are not supported:

- Multi-master on CANopen bus
- Complex manufacturer-specific data types
- Scanning from a BMENOC03\*1
- Access to CAN layer 2
- Copy/paste a BMECXM

**NOTE:** The motion function blocs (MFB) are only supported when the BMECXM module is scanned by the RIO scanner.

**NOTE:** The motion function blocs (MFB) are not supported when the BMECXM module is configured on AUX tasks.



# BMECXM Module Installation and Replacement

## What's in This Chapter

BMECXM Module Installation .....	25
BMECXM Module Replacement.....	28

## Introduction

This chapter presents the instructions to install and replace BMECXM modules.

## BMECXM Module Installation

### Introduction

In a M580 architecture, you can mount the BMECXM module on a local rack or a remote drop.

The BMECXM module will only power up on a BMEXBP•••• Ethernet backplane.

**NOTE:** The minimum firmware version for the backplane is V1.0.

For more information on backplane, refer to *Modicon X80 Racks and Power Supplies, Hardware, Reference Manual*.

### Selecting a Backplane

On the BMEXBP•••• backplanes, the BMECXM module can be mounted in any open slot, except the following restrictions:

- When mounted on a local rack, slots 0 and 1 are reserved for the M580 CPU.
- When mounted on a remote drop, slot 0 is reserved for the drop end communicator module.

In addition to the above restrictions on BMEXBP•••• backplanes, some open slots are not allowed for the BMECXM module when mounted on:

BMEXBP1002 backplane:	slots 2 and 8 are not allowed.
BMEXBP1200 backplane:	slots 2, 8, 10, and 11 are not allowed.
BMEXBP1402 backplane:	slots 10, 11, 12, and 13 are not allowed.

## Grounding Considerations

The BMECXM module is equipped with ground connection contacts at the rear for grounding purposes. These contacts connect the grounding bus of the module to the grounding bus of the rack.

To ground the rack, refer to the chapter **Grounding the Rack and Power Supply Module** (see Modicon X80, Racks and Power Supplies, Hardware Reference Manual).

## Installing the Module on the Rack

### DANGER

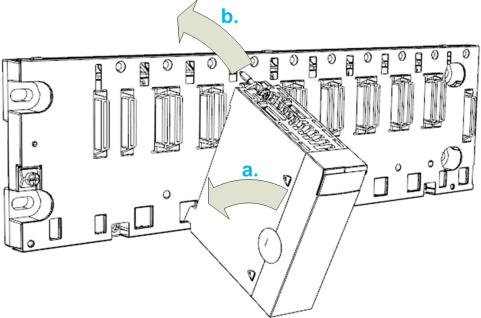
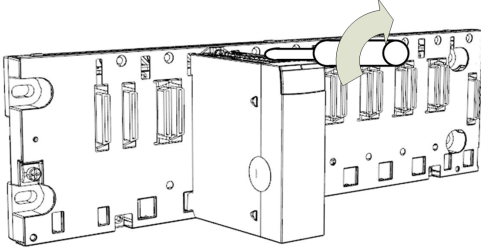
#### ELECTRICAL SHOCK HAZARD

- Switch off the power supply at both ends of the PAC connection, and lock out and tag out both the power sources.
- In case lock out and tag out are not available, ensure that the power sources cannot be inadvertently switched on.
- Use suitable insulation equipment when inserting or removing all or part of this equipment.

**Failure to follow these instructions will result in death or serious injury.**

To mount the module on the backplane, follow these steps:

Step	Action
1	Turn off the power supply to the rack.
2	Remove the protective cover from the module interface on the rack.

Step	Action
3	 <p data-bbox="286 527 1149 552"><b>a.</b> Insert the locating pins on the bottom of the module into the corresponding slots in the rack.</p> <p data-bbox="286 571 1162 617"><b>b.</b> Use the locating pins as a hinge and pivot the module until it is flush with the rack. (The twin connector on the back of the module inserts the connectors on the rack).</p>
4	<p data-bbox="286 662 911 686">Tighten the retaining screw to hold the module in place on the rack:</p>  <p data-bbox="329 974 924 998"><b>NOTE:</b> The tightening torque is 0.4...1.5 N•m (0.30...1.10 lbf-ft).</p>
5	<p data-bbox="286 1019 1045 1044">Connect the CANopen cable to the CANopen connector of the BMECXM module.</p> <p data-bbox="329 1052 1122 1076"><b>NOTE:</b> For details on CANopen network, refer to <i>CANopen Hardware Setup Manual</i>.</p>

A bad module connection can lead to an unexpected behavior of the system.

## ⚠ WARNING

### UNINTENDED EQUIPMENT OPERATION

- Tighten the retaining screw of the module.
- Tighten the retaining screws of the SUB-D 9 CANopen connector.

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

# BMECXM Module Replacement

## Overview

At any time and without turning off the power to the rack, any BMECXM module on the rack can be replaced by another module with compatible firmware.

### **DANGER**

#### **EXPLOSION OR ELECTRIC SHOCK**

- Only perform a hot swap operation in locations known and confirmed to be non-hazardous
- Use only your hands and suitable insulation equipment.
- Do not use any metal tools.

**Failure to follow these instructions will result in death or serious injury.**

### **WARNING**

#### **HAZARD OF ELECTRIC SHOCK**

Before attempting to hot swap a BMECXM module:

- Positively confirm that the rack is connected to the protective earth ground.
- Positively confirm that you have an equipotential grounding system in place.

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

The replacement module obtains its operating parameters over the Ethernet. Assuming that the FDR server is enabled (see Modicon M580, Hardware, Reference Manual), the transfer occurs to the BMECXM module.

**NOTE:** You have to understand and plan for the consequences of hot-swapping a module. Disconnecting a module will interrupt communication to the connected CANopen slave devices.

## ⚠ WARNING

### UNINTENDED EQUIPMENT OPERATION

- Thoroughly identify and understand all implications and consequences of operating mode modifications before attempting them with a new device.
- Take all necessary preventive measures to ensure safe conditions before making operating mode modifications.

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

## Hot Swapping Procedure

To hot-swap the module, follow these steps:

Step	Action
1	Remove the CANopen cable from the module.
2	Remove the module from the backplane.
3	Install the new module to the free slot of backplane.  Tighten the retaining screw to hold the module in place on the rack: <b>NOTE:</b> The tightening torque is 0.4...1.5 N•m (0.30...1.10 lbf-ft).
4	Reconnect the CANopen cable to the CANopen connector of the BMECXM module.

A bad module connection can lead to an unexpected behavior of the system.

## ⚠ WARNING

### UNINTENDED EQUIPMENT OPERATION

- Tighten the retaining screw of the module.
- Tighten the retaining screws of the SUB-D 9 CANopen connector.

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

## Replacing a CANopen Slave

**NOTE:** For CANopen slave device replacement procedure refer to respective device instructions.

After a CANopen slave device replacement, the BMECXM module sends the initial parameters automatically to the new device that restarts automatically. If you have changed the initial parameters, you need to retransfer them explicitly from the application.

---

# CANopen Software Implementation

## What's in This Part

Generalities .....	32
CANopen Configuration .....	47
Ethernet Services Configuration .....	84
Language Objects .....	101
Programming .....	104
Diagnostics .....	118
Firmware Upgrade .....	145

## Subject of This Part

This part describes the various possibilities for software configuration, programming, and diagnostics in a CANopen application.

# Generalities

## What's in This Chapter

Implementation Presentation .....	32
Maximum Configuration .....	34
Device PDO Allocation .....	36
Performance.....	38
Operating Modes .....	40
Fallback Strategy .....	44

## Introduction

This chapter describes CANopen software implementation principles on the Modicon M580 bus.

# Implementation Presentation

## Overview

To implement a CANopen bus, you have to define the physical context of the application in which the bus is integrated: rack, supply, processor, and modules. Then you have to ensure that the necessary software is implemented.

## Implementation Principle

This table shows the different implementation phases:

Phase	Description	Mode
Configuration	Setting of configuration parameters.	Offline
Programming	Programming these specific functions: <ul style="list-style-type: none"> <li>• Implicit bit objects or associated words via the device DDT</li> <li>• Explicit bit objects or associated words via read/write SDO</li> </ul>	Offline or online
Transfer	Transferring the application to the PLC.	Online



Phase	Description	Mode
Debugging and diagnostics	Debugging the application, controlling inputs/outputs, and accessing diagnostic messages with: <ul style="list-style-type: none"> <li>• LED indicators</li> <li>• Device DDTs</li> <li>• DTM diagnosis</li> <li>• Embedded webpages</li> </ul>	Online
Documentation	Printing the various information relating to the configuration of the CANopen master and devices.	Offline or online

**NOTE:**

- For more information on modes (Offline or Online), refer to chapter Project management (see EcoStruxure™ Control Expert, Operating Modes).
- The above order is given for your information. Control Expert software enables you to use editors in the desired order of interactive manner.

## ***NOTICE***

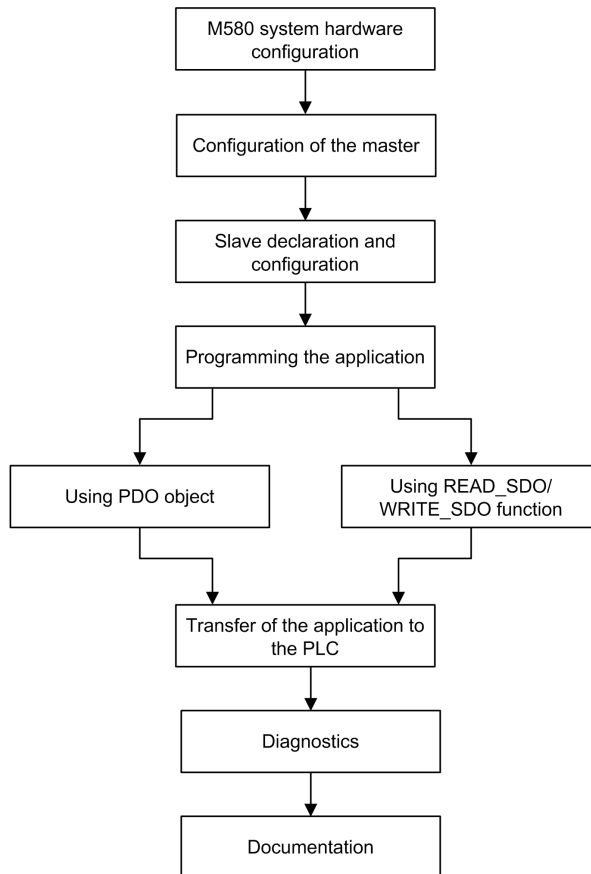
**DIAGNOSTIC DELAY**

- Use diagnosis system information and monitor the response time of the communication.
- In case of disturbed communication, the response time can be 1 to 2s.

**Failure to follow these instructions can result in equipment damage.**

## Implementation Method

This flowchart shows the CANopen port implementation method for BMECXM modules:



## Maximum Configuration

### Overview

The maximum configuration is determined by:

- BMECXM limits
- M580 limits

## BMECXM Limits

If one of these limits is reached, the maximum capacity of the configuration is reached. In that case, you can add another BMECXM module in the architecture.

This table shows the maximum configuration of the BMECXM module:

Parameter	Maximum value
Number of devices supported	63
Number of PDO	<ul style="list-style-type: none"> <li>• 256 IN</li> <li>• 256 OUT</li> </ul> <p><b>NOTE:</b> For more information, refer to device PDO Allocation, page 36.</p>
Process image size	4 Kb IN and 4 Kb OUT
BMECXM configuration file (prm file on FDR server)	64 Kb

## M580 Limits

This table shows how the maximum configuration can depend on other limits:

Parameter	Depends on
Number of Ethernet slots	Rack type
Number of racks	CPU
Scanner capacity	CPU type for: <ul style="list-style-type: none"> <li>• Number of RIO and DIO equipment supported</li> <li>• IN and OUT memory capacity</li> </ul> <p><b>NOTE:</b> All the resources are shared with all the other equipment that is configured on Ethernet I/O.</p>

If these limits are reached, you can distribute the equipment between the RIO or DIO scanner of the CPU or add DIO scanners to the architecture.

# Device PDO Allocation

## Overview

Depending on your configuration, you can check if the maximum number of devices, Tx PDO, or Rx PDO is reached.

You can do the same calculation with third-party devices integrated via the Hardware Catalog Manager.

## Maximum Number of PDOs

This table indicates the maximum number of PDOs used by each device present by default in the Control Expert Catalog Manager:

Family	Device	Tx PDO	Rx PDO
<b>Motor control</b>	APP_1CC00	5	5
	APP_1CC02	5	5
	TeSysT_MMC_L	4	4
	TeSysT_MMC_L_EV40	4	4
	TeSysT_MMC_R	4	4
	TeSysT_MMC_R_EV40	4	4
	TeSysU_C_Ad	4	4
	TeSysU_C_Mu_L	4	4
	TeSysU_C_Mu_R	4	4
	TeSysU_Sc_Ad	4	4
	TeSysU_Sc_Mu_L	4	4
	TeSysU_Sc_Mu_R	4	4
	TeSysU_Sc_St	4	4
<b>Sensors</b>	OsiCoder	2	0
<b>Distributed I/Os</b>	FTB_1CN08E08CM0	2	2
	FTB_1CN08E08SP0	2	2
	FTB_1CN12E04SP0	2	2

Family	Device	Tx PDO	Rx PDO
	FTB_1CN16CM0	2	2
	FTB_1CN16CP0	2	2
	FTB_1CN16EM0	2	2
	FTB_1CN16EP0	2	2
	FTM_1CN10	5	5
	OTB_ISLAND	8	8
	OTB_1C0_DM9LP	8	8
	STB_NCO_1010	32	32
	STB_NCO_2212	32	32
<b>Motion &amp; Drive</b>	ATV312_V5_1	2	2
	ATV31_V1_1	2	2
	ATV31_V1_2	2	2
	ATV31_V1_7	2	2
	ATV31T_V1_3	2	2
	ATV32_MFB	3	3
	ATV61_V1_1	3	3
	ATV71_V1_1	3	3
	lclA_IFA	1	1
	lclA_IFE	1	1
	lclA_IFS	1	1
	LXM05_MFB	4	4
	LXM05_V1_12	4	4
	LXM15LP_V1_45	4	4
	LXM15MH_V6_64	4	4
	SD3_28	4	4
<b>Safety</b>	XPSMC16ZC	4	0
	XPSMC32ZC	4	0
<b>Third party products</b>	CPV_C02	1	1
	CPX_FB14	4	4
	P2M2HBVC11600	1	1

# Performance

## Impact on Task Cycle Time

The impact of the PDO broadcasting on the task cycle time is as follows:

Task	Typical
CANopen inputs	xx $\mu$ s/PDO
CANopen outputs	xx $\mu$ s + xx $\mu$ s/PDO
Diagnostics	xx $\mu$ s

For more information, refer to the Modicon M580 performance characteristics (see Modicon M580, Hardware, Reference Manual).

## RPI Performance

According to your configuration, you can get a message from Control Expert.

In that case, you can check that:

- The data amount and RPI are compatible with the performance objective of the BMECXM modules: 2000 packets/s for all the IN and OUT connections of the BMECXM modules.
- The RPI is compatible with the fieldbus period.

If not, the RPI can be increased by the DTM. If you need determinism, you can either reduce the CANopen configuration, increase the baudrate, or increase the task period.

For more information, refer to RPI values, page 94.

## SDO Performance

There is one SDO at a time for each slave. Up to 63 SDOs can be managed in parallel according to the CPU capacity.

SDO are sent and received in synchronisation with the PLC Mast task. Therefore, the SDO response time is linked to the period of the MAST task, to the CANopen baudrate and to the response time of the slave.

## Bus Start

The CANopen bus start time depends on the number of devices.

The minimum time to start a CANopen bus is 27 seconds.

The time to configure one device is about 0.8 second.

The start time of a CANopen bus with 63 devices is about 1 minute.

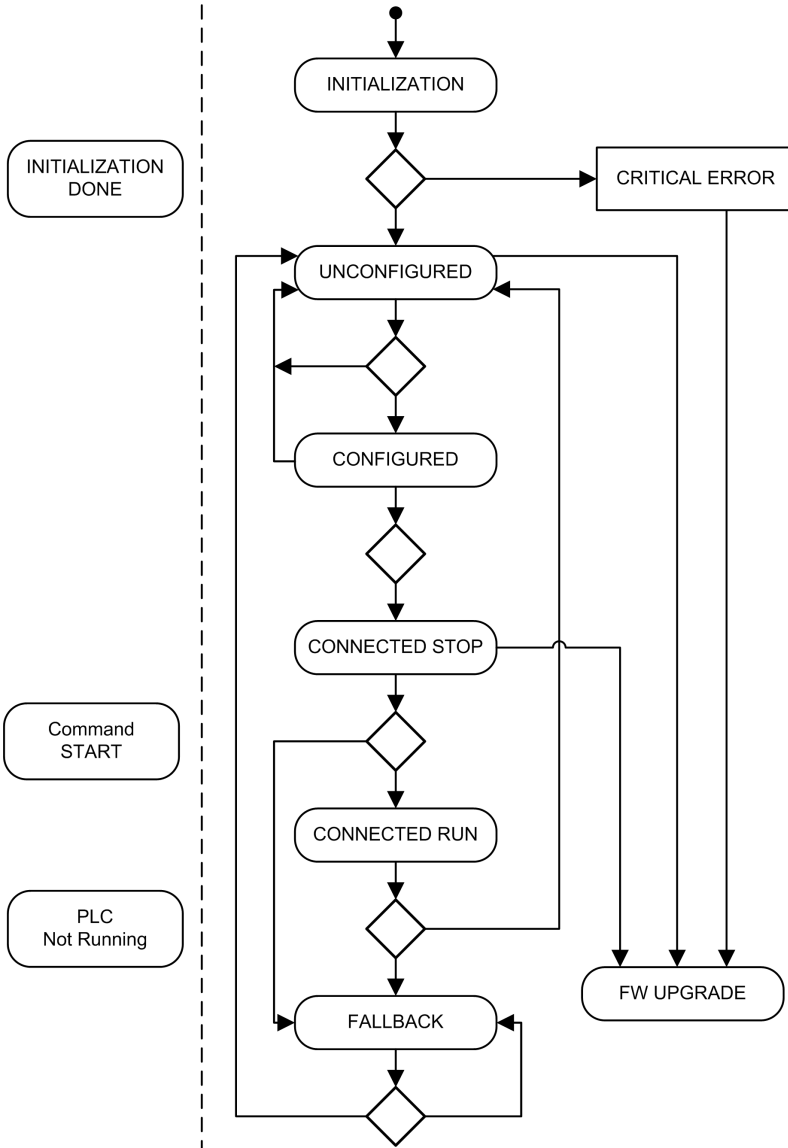
## Device Disconnection

The time to detect the disconnection of a device depends on the protocol configuration, page 70:

Protocol	Description
Node Guarding	The time to detect the disconnection is <b>Guard Time * Life Time Factor</b>
Heartbeat	The time to detect the disconnection is <b>Node Heartbeat Producer Time + Node Heartbeat consumer Time</b>

# Operating Modes

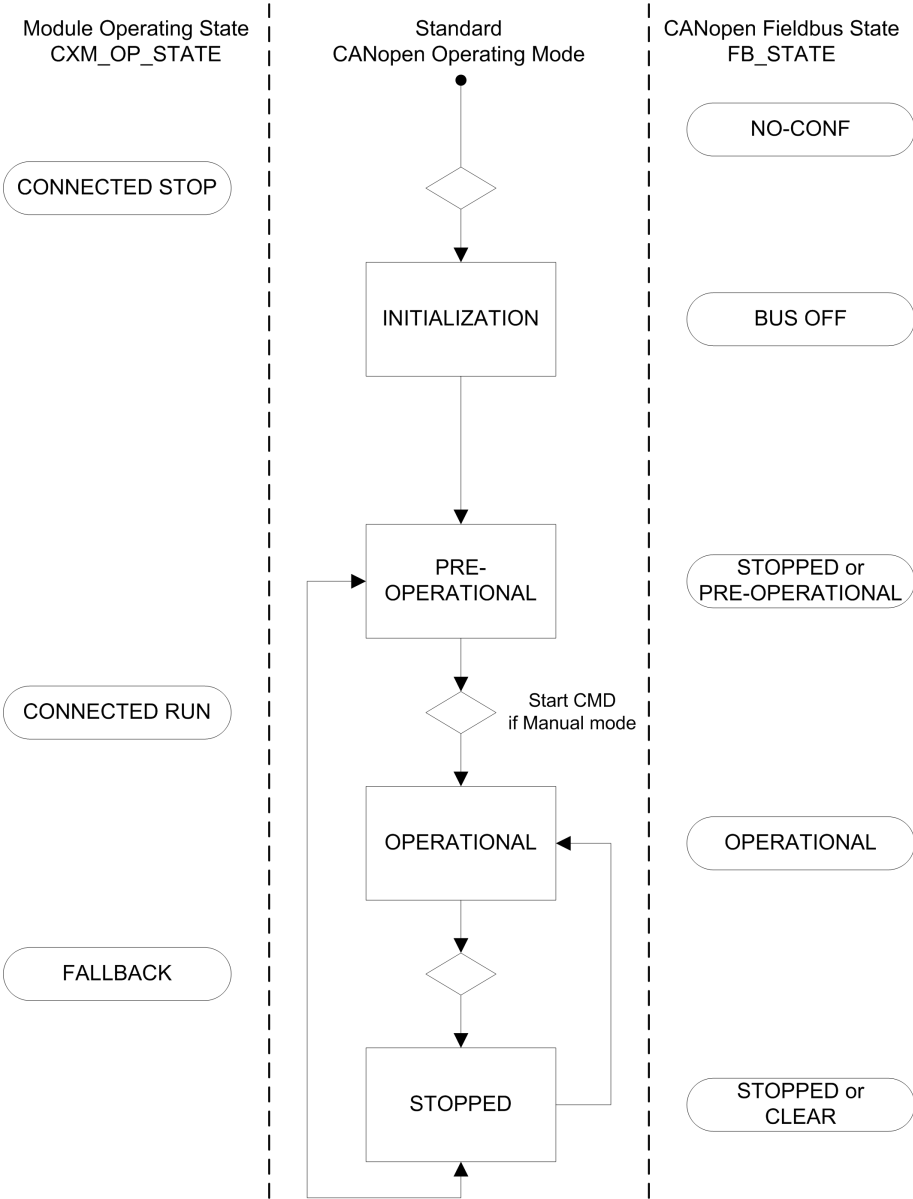
## BMECXM States





BMECXM State	Description
<i>INITIALIZATION</i>	<p>The module enters in this mode on power on or following up a reboot. The power on self-test (POST) is performed during this phase.</p> <p><b>NOTE:</b> When the power on self-test failed, the module switch to <i>CRITICAL ERROR</i> state. Refer to LEDs, page 121 description to diagnose this state.</p>
<i>UNCONFIGURED</i>	<p>The power on self-test is completed and the module proceed to:</p> <ul style="list-style-type: none"> <li>• Backplane initialization,</li> <li>• Get IP address (from DHCP),</li> <li>• Get configuration file (from FDR server).</li> </ul>
<i>CONFIGURED</i>	<p>The module has received its configuration file and is waiting for all expected EtherNet/IP connection with the PLC, as defined in the configuration file (<i>.prm</i> file).</p> <p>After 5 s, if not all expected connections are opened, the BMECXM module restarts.</p>
<i>CONNECTED STOP</i>	<p>The boot slave procedure is started to initialize all CANopen slave devices configured in the <i>.prm</i> file. The status of each device is progressively updated in the <b>Slave Live List</b> tab.</p> <p><b>NOTE:</b> If the BMECXM module is configured in manual mode, the <i>EM_start</i> command is mandatory to reach the <i>CONNECTED RUN</i> state.</p>
<i>CONNECTED RUN</i>	<p>This state is reached when:</p> <ul style="list-style-type: none"> <li>• all EtherNet/IP connection are connected and in <i>RUN</i> state.</li> <li>• The CANopen fieldbus is in <i>OPERATIONAL</i> state.</li> </ul> <p>In <i>CONNECTED RUN</i> state, the BMECXM module exchanges I/O with the CANopen slave devices.</p> <p>From this state, the PLC can drive the BMECXM module state by sending NMT requests.</p> <p>The PLC can also send NMT request, page 104 via explicit message to CANopen slave devices to drive their state individually.</p>
<i>FALLBACK</i>	<p>The BMECXM module switches in <i>FALLBACK</i> state and follow the Fallback strategy, page 44.</p>
<i>FW UPGRADE</i>	<p>The firmware upgrade is in progress. Only the firmware upgrade tool can communicate with the BMECXM module, and EtherNet/IP connections with the PLC are stopped.</p>

# CANopen States



CANopen States	Description
<i>INITIALIZATION</i>	<p>The BMECXM module executes the initialization of the CANopen bus according to the Boot-up procedure. It is done when the BMECXM module is in <i>CONNECTED STOP</i> state.</p> <p>During the CANopen bus initialization, only read accesses to the object dictionary of the CANopen master and slave devices are possible via the SDO command interface.</p>
<i>PRE-OPERATIONAL</i>	<p>The boot-up sequence is now completed. In this state, no command has been received to enter in <i>OPERATIONAL</i> state.</p> <p>The CANopen slave devices respond to SDO and NMT messages but not to PDOs.</p>
<i>OPERATIONAL</i>	<p>This is the main state of the CANopen bus.</p> <p>In this state:</p> <ul style="list-style-type: none"> <li>• The error control service is active.</li> <li>• The detection of the CANopen slave devices is started, according to the assignment in the configuration file.</li> <li>• Emergency and boot-up messages are received.</li> </ul> <p><b>NOTE:</b> If the BMECXM module is configured in manual mode, the <i>EM_start</i> command is mandatory to reach the <i>OPERATIONAL</i> state.</p>
<i>STOPPED</i>	<p>In this state, only the slave monitoring is active. No service is available to read or write SDO.</p>

## EtherNet/IP Connections States

BMECXM Operating State	Fieldbus Operating State	EtherNet/IP State
<i>IDLE</i>	<i>IDLE</i>	There is no connection
<i>UNCONFIGURED</i>	<i>NO-CONF</i>	
<i>CONFIGURED</i>	<i>BUS OFF</i>	
<i>CONNECTED STOP</i>	<i>PRE OPERATIONAL</i>	The connection is running
<i>CONNECTED RUN</i>	<ul style="list-style-type: none"> <li>• <i>OPERATIONAL</i><sup>(1)</sup></li> <li>• <i>STOPPED</i><sup>(1)</sup></li> <li>• <i>PRE OPERATIONAL</i><sup>(1)</sup></li> </ul>	
<i>FALLBACK</i>	<p>Fallback strategy:</p> <ul style="list-style-type: none"> <li>• <i>OPERATIONAL</i><sup>(2)</sup></li> <li>• <i>STOPPED</i><sup>(3)</sup></li> <li>• <i>CLEAR</i><sup>(4)</sup></li> </ul>	<p>Depending on the I/O scanner type:</p> <ul style="list-style-type: none"> <li>• <i>STOP</i></li> <li>• <i>IDLE</i></li> <li>• <i>CLOSE</i></li> </ul>

BMECXM Operating State	Fieldbus Operating State	EtherNet/IP State
<i>FW UPGRADE</i>	Not applicable	There is no connection. The connection is aborted if existing.
<p>(1) Switching from one state to another one is done via an NMT command.</p> <p>(2) The CANopen fieldbus state stays in <i>OPERATIONAL</i> but with the last data received from the PLC.</p> <p>(3) CANopen master and slaves switch to <i>STOPPED</i> state.</p> <p>(4) The CANopen fieldbus state stays in <i>OPERATIONAL</i> but with data set to zero.</p>		

## Communication Objects

The following table presents which communication objects are allowed depending on the CANopen states:

Object	<i>INITIALIZATION</i>	<i>PRE-OPERATIONAL</i>	<i>STOPPED</i>	<i>OPERATIONAL</i>
PDO	–	–	–	Yes
SDO	–	Yes	–	Yes
SYNC	–	Yes	–	Yes
EMCY	–	Yes	–	Yes
Bootup	Yes	–	–	–
NMT	–	Yes	Yes	Yes
– Communication object not allowed.				

## Fallback Strategy

### Overview

The CANopen X80 master module switches in *FALLBACK* state:

- as soon as it detects that the PLC is stopped, or
- after the holdup time, if at least one EtherNet/IP connection of the PLC is closed.

The following table gives an overview of the module behavior following a PLC *STOP*:

		RIO Scanner	DIO Scanner		
		BMEP58-040	BMEP58-040	BMEP58-020	
		Default configuration	Default configuration	Default configuration	User configuration
EtherNet/IP	Connection <sup>(1)</sup>	Open	Closed	Open	Closed
	Run/Idle flag	Run	–	Idle	–
Data (I/O Exchange)		Hold previous	No exchange. Device DDT is cleared	Hold previous	No exchange. Device DDT is cleared
(1) CIP connection between the CPU and the CANopen X80 master module.					

## Output Fallback Strategy

In *FALLBACK* state the module applies the fallback strategy configured in the **Ethernet IO** tab of the DTM:

- **Operational, outputs maintained:** The CANopen fieldbus stays in *OPERATIONAL* state and slaves outputs are maintained with the last values.
- **Operational, outputs set to 0:** The CANopen fieldbus stays in *OPERATIONAL* state and slaves outputs are set to 0 (zero).
- **Stop:** The CANopen fieldbus switches to *STOPPED* state and all devices on the network switched to *STOPPED* state (broadcast NMT).

## Input Fallback Strategy

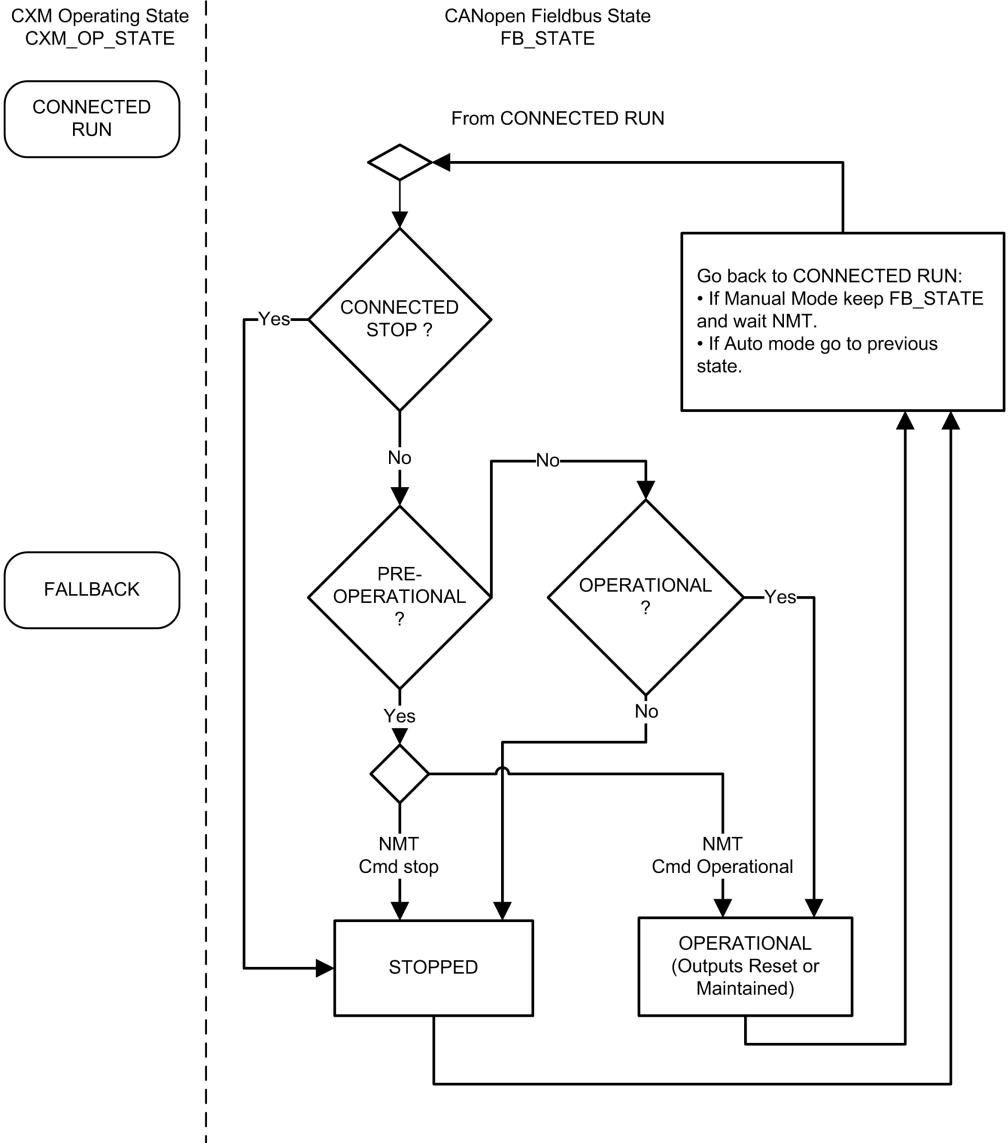
In case of EtherNet/IP connection is lost between the PLC and the BMECXM module, all inputs values of the device DDT are set to 0 (zero), including the *Health* bits.

In case of PLC stopped event, the behavior of inputs differs according to the EtherNet/IP connection state:

- EtherNet/IP connections are closed: input data values are set to 0 (zero).
- EtherNet/IP connections are not closed, Run/Idle flag to Idle: input data are refreshed
- EtherNet/IP connections are not closed, Run/Idle flag to Run, output data contains the CPU state (STOP): input data are refreshed

# States in Fallback Mode Strategy

The following diagram gives the CANopen fieldbus states in fallback mode:



# CANopen Configuration

## What's in This Chapter

Overview .....	47
Adding a CANopen X80 Master BMECXM module .....	50
Bus Configuration .....	52
Device Configuration.....	60
Master Configuration.....	77

## Introduction

This chapter presents the CANopen configuration. It shows how to use Control Expert programming software to select and configure BMECXM modules and CANopen slave devices.

## Overview

### Subject of This Section

This section describes how to use Control Expert software for the CANopen configuration.

## Overview

### Introduction

The CANopen configuration consists in the configuration of the CANopen fieldbus and of the bus master and slaves.

The configuration of CANopen architecture is integrated into Control Expert.

In the same M580 PAC, you can configure several CANopen master BMECXM modules with associated CANopen slave devices.

**⚠ CAUTION****UNEXPECTED EQUIPMENT BEHAVIOR**

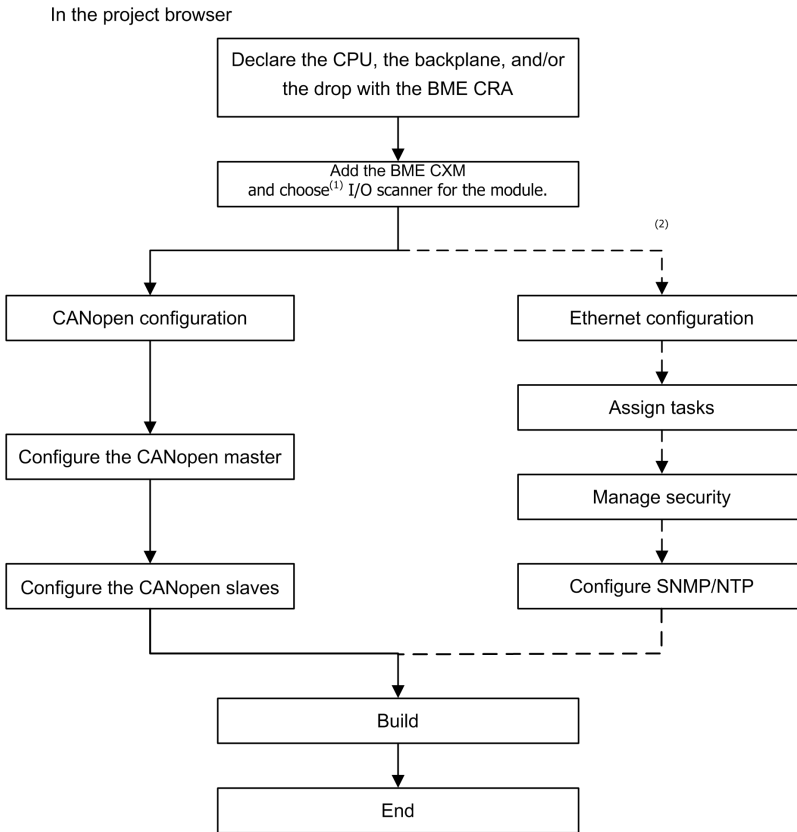
- Configure each CANopen slave device on the correct CANopen master BMECXM module.
- Always verify that the Control Expert configuration is consistent with the hardware installation.

**Failure to follow these instructions can result in injury or equipment damage.**



## Configuration Steps

This chart represents the steps for configuration of the CANopen architecture:



**(1)** After validation of module insertion in the project, if you want to change the scanner association, you have to delete the device from the configuration and then recreate the device with the new scanner association.

**(2)** Optional steps.

**NOTE:** For more information on optional steps, refer to chapter Ethernet Services Configuration, page 84.

# Adding a CANopen X80 Master BMECXM module

## Subject of This Section

This section describes how to use Control Expert software for selecting BMECXM modules.

## Adding a CANopen X80 Master BMECXM Module

### Prerequisite

Before adding the module you have to declare the M580 CPU and if required, the remote drop(s).

### Procedure

To add a BMECXM module to the Control Expert project, follow these steps:

Step	Action
1	From the <b>Project Browser</b> , expand (+) the <b>Configuration</b> directory.
2	Depending on your hardware architecture, double-click: <ul style="list-style-type: none"> <li>• The <b>PLC bus</b> subdirectory for a local rack</li> <li>• The <b>EIO Bus</b> subdirectory for a remote drop</li> </ul>
3	Right-click the open slot in the rack and click <b>New Device....</b> <b>Result:</b> The <b>New Device</b> window is displayed. Expand (+) <b>Communication</b> to select the BMECXM module and click <b>OK</b> . <b>NOTE:</b> You can also click <b>Tools &gt; Hardware Catalog &gt; Modicon M580 local drop &gt; Communication</b> and drag the BMECXM module to an open slot in the rack.
4	In the popup window, select the appropriate combination of <b>I/O-scanner</b> , <b>Protocol</b> , and <b>Profile</b> and click <b>OK</b> . <b>Result:</b> The <b>Properties of device</b> window is displayed. <b>NOTE:</b> All the tabs contain read-only information except the <b>General</b> tab.
5	In the <b>General</b> tab, you can change the name <sup>(1)</sup> of the module in the <b>Name</b> field. In that case, the fields in the <b>Default I/O vision management</b> box are automatically modified to match the new name. By default, the prefix of the name corresponds to the topological address of the module in the configuration. For a BMECXM module on bus EIO, on drop 3, on rack 1, and slot 4, the default name will be EIO2_d3_r1_s4_ECXM0100.

Step	Action
	<b>NOTE:</b> The default naming rule in Control Expert helps prevent modules of the same type getting mixed up.
6	Click <b>OK</b> .  <b>Result:</b> The new device is added.
7	Save the project by clicking <b>File &gt; Save</b> .
(1) In Control Expert, this name is also used as: <ul style="list-style-type: none"> <li>• The module name in the <b>DTM Browser</b> under the <b>Host PC</b></li> <li>• The device DDT name</li> </ul>	

## **WARNING**

### **UNEXPECTED EQUIPMENT BEHAVIOR**

- Always verify that the Control Expert configuration is consistent with the hardware installation.
- If you attempt to change the default names of your devices, manage the naming to prevent from addressing the wrong device.

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

## Control Expert Commands for the Module

In the Control Expert bus editor (local or remote), right-click the BMECXM module to access these commands:

Name	Description
<b>Delete Module</b> <sup>(1)</sup>	Delete the selected module from the rack.
<b>Open Module</b> <sup>(1)</sup>	See a description of the selected communications module.
<b>Move Module</b> <sup>(1)</sup>	Move the selected module to the rack slot that you designate.
<b>Go to DTM</b>	Show the DTM of the selected module in the <b>DTM Browser</b> .

Name	Description
<b>Power Supply Budget...</b> <sup>(2)</sup>	Access the <b>Power supply</b> tab and view: <ul style="list-style-type: none"><li>• The total power</li><li>• The power discharged in the module for each voltage it uses</li></ul> <b>NOTE:</b> Close this window before performing any command in Control Expert.
(1) This command also appears in the <b>Edit</b> menu.	
(2) This command also appears in the <b>Services</b> menu.	

## Bus Configuration

### Subject of this Section

This section presents the configuration of the CANopen bus.

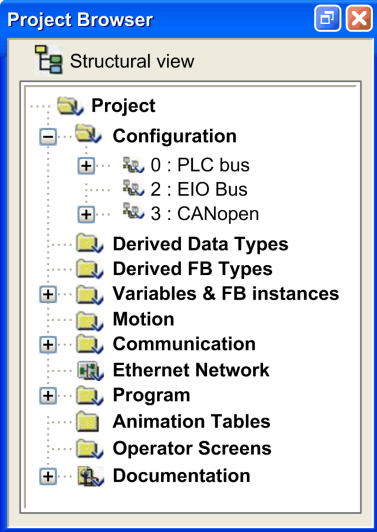
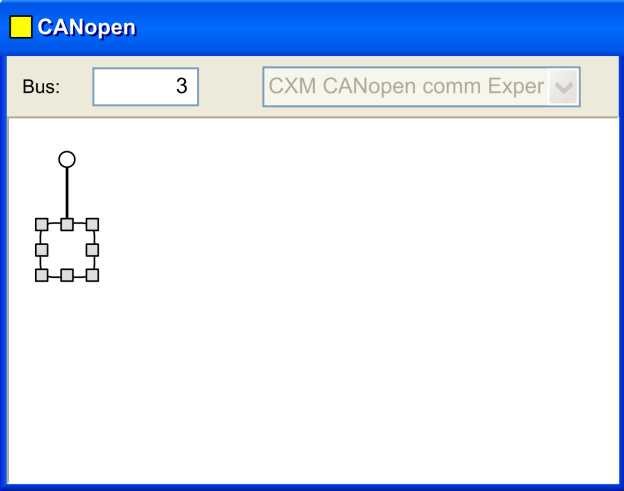
## Access the CANopen Bus Editor

### Overview

As soon as the BMECXM module is configured in the Control Expert project, a node is automatically created in the **Project Browser**.

### Procedure

To access the CANopen bus editor, follow these steps:

Step	Action
1	<p>From the <b>Project Browser</b>, expand (+) the <b>Configuration</b> directory:</p>  <p>The screenshot shows the 'Project Browser' window with a tree view. The 'Configuration' directory is expanded, showing sub-directories: '0 : PLC bus', '2 : EIO Bus', and '3 : CANopen'. Other visible directories include 'Derived Data Types', 'Derived FB Types', 'Variables &amp; FB instances', 'Motion', 'Communication', 'Ethernet Network', 'Program', 'Animation Tables', 'Operator Screens', and 'Documentation'.</p>
2	<p>Double-click the CANopen directory.</p> <p><b>Result:</b> The <b>CANopen</b> window is displayed:</p>  <p>The screenshot shows the 'CANopen' configuration window. It has a title bar with a yellow icon and the text 'CANopen'. Below the title bar, there is a 'Bus:' label, a text input field containing the number '3', and a dropdown menu showing 'CXM CANopen comm Exper'. The main area of the window contains a schematic diagram of a CANopen network topology, consisting of a central node connected to a ring of six other nodes.</p> <p><b>NOTE:</b> You can also select the CANopen subdirectory and click <b>Open</b> in the contextual menu.</p>

# Adding Slave Devices on the CANopen Bus

## Overview

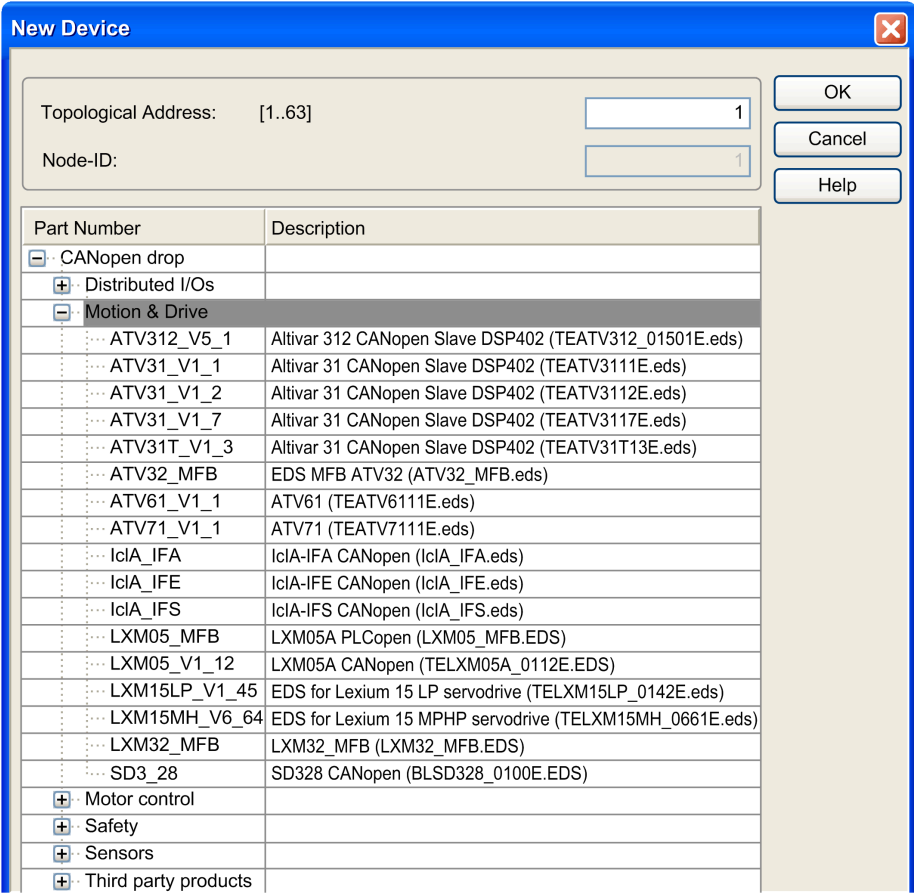
You can launch the bus editor from this node to define the topology of the bus and configure the CANopen elements.


**NOTE:** You cannot modify the configuration of the CANopen bus in connected mode.

## Procedure

To add a slave device follow these steps:

Step	Action
1	Access the CANopen bus editor, page 52.
2	Double-click the connection point, where the module should be connected.

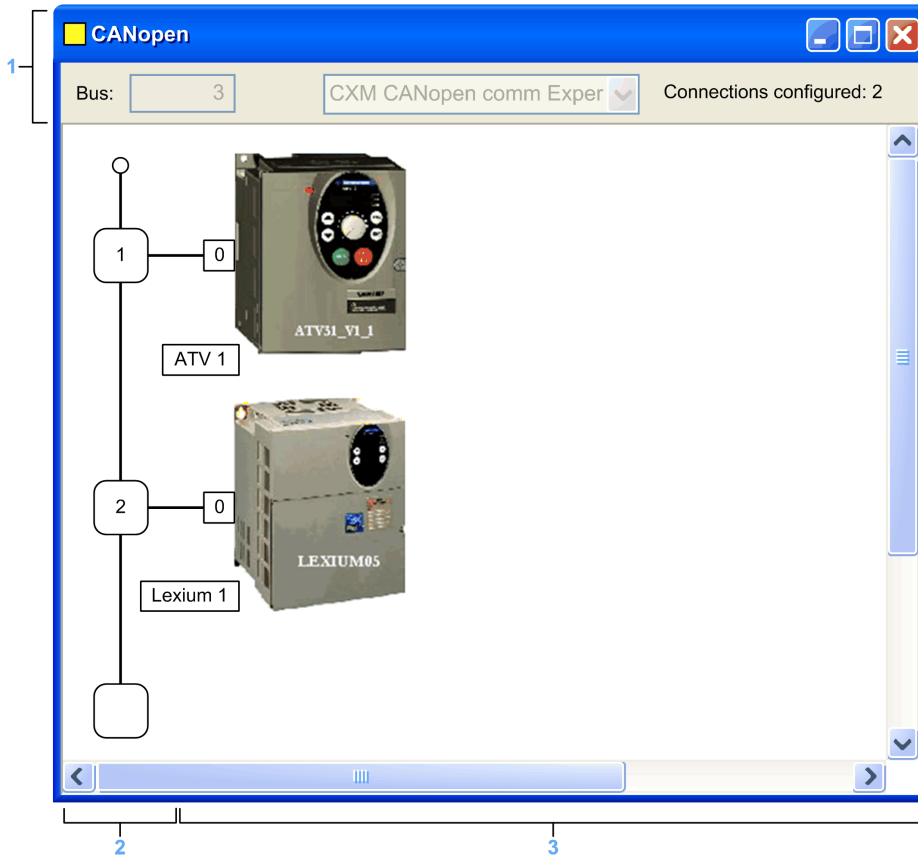
Step	Action
	<p><b>Result:</b> the <b>New Device</b> window is displayed.</p>  <p>The screenshot shows the 'New Device' dialog box with the following details:</p> <ul style="list-style-type: none"> <li><b>Title Bar:</b> New Device (with a close button)</li> <li><b>Topological Address:</b> [1..63] (input field contains 1)</li> <li><b>Node-ID:</b> (input field contains 1)</li> <li><b>Buttons:</b> OK, Cancel, Help</li> <li><b>Tree View:</b> <ul style="list-style-type: none"> <li>- CANopen drop                     <ul style="list-style-type: none"> <li>+ Distributed I/Os</li> <li>- Motion &amp; Drive                             <ul style="list-style-type: none"> <li>... ATV312_V5_1 Altivar 312 CANopen Slave DSP402 (TEATV312_01501E.eds)</li> <li>... ATV31_V1_1 Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</li> <li>... ATV31_V1_2 Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</li> <li>... ATV31_V1_7 Altivar 31 CANopen Slave DSP402 (TEATV3117E.eds)</li> <li>... ATV31T_V1_3 Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</li> <li>... ATV32_MFB EDS MFB ATV32 (ATV32_MFB.eds)</li> <li>... ATV61_V1_1 ATV61 (TEATV6111E.eds)</li> <li>... ATV71_V1_1 ATV71 (TEATV7111E.eds)</li> <li>... IclA_IFA IclA-IFA CANopen (IclA_IFA.eds)</li> <li>... IclA_IFE IclA-IFE CANopen (IclA_IFE.eds)</li> <li>... IclA_IFS IclA-IFS CANopen (IclA_IFS.eds)</li> <li>... LXM05_MFB LXM05A PLCopen (LXM05_MFB.EDS)</li> <li>... LXM05_V1_12 LXM05A CANopen (TELXM05A_0112E.EDS)</li> <li>... LXM15LP_V1_45 EDS for Lexium 15 LP servodrive (TELXM15LP_0142E.eds)</li> <li>... LXM15MH_V6_64 EDS for Lexium 15 MPHP servodrive (TELXM15MH_0661E.eds)</li> <li>... LXM32_MFB LXM32_MFB (LXM32_MFB.EDS)</li> <li>... SD3_28 SD328 CANopen (BLSD328_0100E.EDS)</li> </ul> </li> <li>+ Motor control</li> <li>+ Safety</li> <li>+ Sensors</li> <li>+ Third party products</li> </ul> </li> </ul> </li> </ul>
3	<p>In the <b>Topological Address</b> field, type the number of the connection point corresponding to the address.</p> <p><b>NOTE:</b> By default, the Control Expert software offers the first free consecutive address.</p>

Step	Action
4	From the <b>CANopen drop</b> , expand (+) the family to select your CANopen device.
5	<p>Click <b>OK</b>.</p> <p><b>Result:</b> the module is declared.</p>  <p>The screenshot shows a software window titled "CANopen". At the top, there are fields for "Bus:" with the value "3", "CANopen comm head Expert 0", and "Connections configured" with the value "1". The main area displays a graphical representation of a CANopen network. A vertical line represents the bus, with a circle at the top. A box labeled "1" is connected to the bus, and a box labeled "0" is connected to it. Below the bus line, there is a box with three dots "...". To the right of the bus is a photograph of a physical device labeled "ATV31_V1_1". The device has a circular display and several buttons.</p>



## CANopen Bus Editor

The following graphic describes the different areas of the **CANopen** bus editor:



Number	Element	Function
1	Bus	Bus number
	Connections configured	Indicates the number of connection points configured.
2	Logical address area	Includes the addresses of the devices connected to the bus.
3	Module area	Includes the devices that are connected to the bus.

**NOTE:** Available connection points are indicated by an empty white square.

# Delete/Move/Duplicate a Device on the CANopen Bus

## Deleting a Device

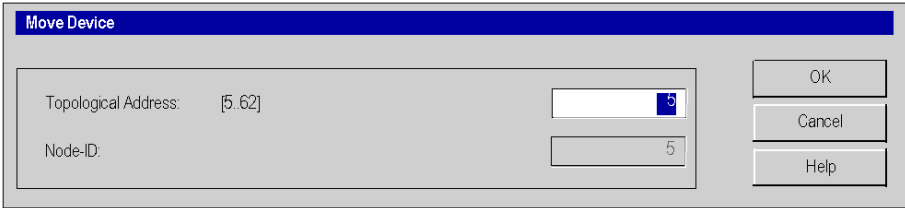
To delete a device follow these steps:

Step	Action
1	Access the CANopen bus editor, page 52.
2	Right-click the connection point of the device to be deleted and click <b>Delete Drop</b> .

## Moving a Device

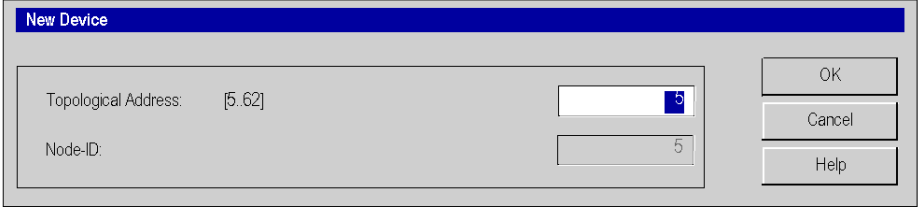
Moving a device does not involve a physical move on the bus, but rather a change in the device topological address. The name of the device DDT instance change except if you have manually modified the default name of your device in the CANopen X80 master module DTM, page 90.

To move a device, follow these steps:

Step	Action
1	Access the CANopen bus editor, page 52.
2	Select the connection point to be moved: a frame surrounds the selected connection point.
3	<p>Drag and drop the connection point to be moved to an empty connection point.</p> <p><b>Result:</b> the <b>Move Device</b> window is displayed:</p> 
4	Type the number of the destination connection point.
5	Click <b>OK</b> to confirm the new connection point.

## Duplicating a Device

To duplicate a device, follow these steps:

Step	Action
1	Access the CANopen bus editor, page 52.
2	Right-click the device to be copied and click <b>Copy</b> .
3	<p>Right-click the destination connection point and click <b>Paste</b>.</p> <p><b>Result:</b> the <b>New Device</b> window is displayed:</p> 
4	Type the number of the destination connection point.
5	Click <b>OK</b> to confirm the new connection point.

## View CANopen Bus in the Project Browser

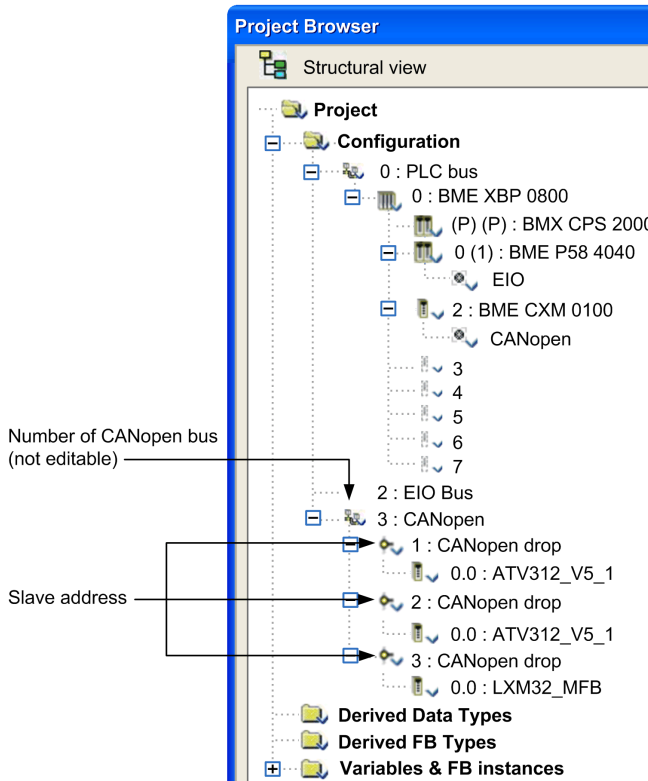
### Overview

The CANopen bus is shown in the configuration directory of the **Project Browser**. The bus number is automatically calculated by Control Expert.

**NOTE:** The value of the bus number cannot be modified.

## Illustration

This illustration shows the CANopen bus and slaves in the **Project Browser**:



**NOTE:** If you right-click **3 : CANopen drop > Go to Bus Master**, the BMECXM module node that corresponds to the bus is automatically selected.

## Device Configuration

### Subject of this Section

This section presents the configuration of the initial parameters of the CANopen devices.

# Presentation of CANopen Devices

## Overview

Devices that you can connect to a CANopen bus and that can be configured in Control Expert are grouped according to their family:

- **Distributed I/Os**
- **Motion & Drive**
- **Motor control**
- **Safety**
- **Sensors**
- **Third-party products**

## CANopen Device Import

Only devices from the **Hardware Catalog** can be used with Control Expert.

To add a CANopen device in the **Hardware Catalog**, you have to:

- Import it into the Hardware Catalog Manager (see EcoStruxure™ Control Expert, Hardware Catalog Manager, Operation Guide).
- Update the **Hardware Catalog** of Control Expert.

**NOTE:** The **Hardware Catalog** in Control Expert, is an overview in read mode only of the Hardware Catalog Manager.

## CANopen Catalog Compatibility Rules

### **NOTE:**

Unity Pro is the former name of Control Expert for version 13.1 or earlier.

The following elements in the catalog are supported on Modicon M580:

- All existing Schneider Electric CANopen devices
- Modules created with Unity Pro  $\geq$  V11.1
- Modules created with Unity Pro  $<$  V11.1
- Specific functions created on old device with Unity Pro  $<$  V11.1
- Specific functions created on existing Schneider Electric devices with Unity Pro  $<$  V11.1

The **Hardware Catalog** can contain devices created with Unity Pro  $<$  V11.1. To update the catalog, you have to:

- Import the \*.cpx file created with the previous Control Expert version in the Hardware Catalog Manager
- Open a \*.sta file created with the previous Control Expert version.

## CANopen Device Configuration

To configure the initial parameters of CANopen devices, you can use:

- Control Expert
- An external tool

Configuration depends on the type of CANopen devices.

**NOTE:** Before configuring a device, select the function when available, page 62.

## Slave Functions

### Overview

To facilitate their configuration, some CANopen devices are represented through functions.

Each function defines:

- Premapped PDOs
- Debugging variables that can be mapped. For more information, refer to the **PDO** tab, page 67.

**NOTE:** The function should be selected before configuring the slave.

### Available Functions

The available functions are as follows:

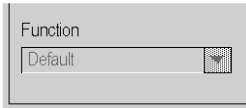
Devices Involved	Function	Description
Altivar	Basic	This function allows a simple control of the speed.
	MFB	This function allows control of the device through PLCOpen Motion function block library.
	Standard	This function allows control of the speed and/or torque. All the parameters that can be mapped are mapped in the supplemental PDOs for: <ul style="list-style-type: none"> <li>• An adjustment of the operating parameters (length of acceleration,...),</li> </ul>

Devices Involved	Function	Description
		<ul style="list-style-type: none"> <li>• Additional surveillance (current value,...),</li> <li>• Additional control (PID, outputs command,...).</li> </ul>
	Advanced	<p>This function allows control of the speed and/or torque.</p> <p>Some parameters can be configured and can also be mapped in the PDOs to allow:</p> <ul style="list-style-type: none"> <li>• An adjustment of the operating parameters (length of acceleration,...),</li> <li>• Additional surveillance (current value,...),</li> <li>• Additional control (PID, outputs command,...).</li> </ul>
STB NCO1010 & NCO2212	Simple	<p>Use this profile if the island does not contain high-resolution analog I/O module or the TeSys U STB modules.</p> <p>This profile contains:</p> <ul style="list-style-type: none"> <li>• NIM diagnostic information (index 4000-index 4006),</li> <li>• 8-bit discrete input information (index 6000),</li> <li>• 16-bit discrete information (index 6100),</li> <li>• 8-bit discrete output information (index 6200),</li> <li>• 16-bit discrete output information (index 6300),</li> <li>• Low-resolution analog input information (index 6401),</li> <li>• Low-resolution analog output information (index 6411).</li> </ul> <p>This profile limits the number of index or subindex entries for any of the above objects to 32. If the island configuration exceeds this limitation, use the large profile.</p>
	Extended	<p>Use this profile if the island contains high-resolution analog I/O module or the TeSys U STB modules.</p> <p>This profile contains:</p> <ul style="list-style-type: none"> <li>• NIM diagnostic information (index 4000-index 4006),</li> <li>• 8-bit discrete input information (index 6000),</li> <li>• 16-bit discrete information (index 6100),</li> <li>• 8-bit discrete output information (index 6200),</li> <li>• 16-bit discrete output information (index 6300),</li> <li>• Low-resolution analog input information (index 6401),</li> <li>• Low-resolution analog output information (index 6411),</li> <li>• High-resolution analog input information or HMI words (index 2200-221F),</li> <li>• High-resolution analog output information or HMI words (index 3200-321F),</li> <li>• TeSys U input information (index 2600-261F),</li> <li>• TeSys U output information (index 3600-361F).</li> </ul> <p>This profile limits the number of index or subindex entries for any of the above objects to 32. If the island configuration exceeds this limitation, use the large profile.</p>
STB NCO2212	Advanced	<p>Use this profile if the island contains enhanced CANopen devices, special features as run-time parameters along with high-resolution analog I/O module or HMI or the TeSys U STB modules.</p>

Devices Involved	Function	Description
		<p>This profile contains:</p> <ul style="list-style-type: none"> <li>• NIM diagnostic information (index 4000-index 4006),</li> <li>• 8-bit discrete input information (index 6000),</li> <li>• 16-bit discrete information (index 6100),</li> <li>• 8-bit discrete output information (index 6200),</li> <li>• 16-bit discrete output information (index 6300),</li> <li>• Low-resolution analog input information (index 6401),</li> <li>• Low-resolution analog output information (index 6411),</li> <li>• High-resolution analog input information or HMI words (index 2200-221F),</li> <li>• High-resolution analog output information or HMI words (index 3200-321F),</li> <li>• TeSys U input information (index 2600-261F),</li> <li>• TeSys U output information (index 3600-361F),</li> <li>• Third-party CANopen devices (index 2000-201F),</li> <li>• RTP information (index 4100 &amp; index 4101).</li> </ul> <p>This profile limits the number of index or subindex entries for any of the above objects to 32. If the island configuration exceeds this limitation, use the large profile.</p>
	Large	Use this profile if the island configuration does not fit any of the above profiles. This profile contains all the objects available for the STB island and hence consumes more memory address location in the CANopen master.
Altivar 61/71	Controller	This function is especially created for CANopen communications with the built-in controller card and all the application cards (pump control,...).
Festo CPV	Advanced	The advanced level is designed to configure the maximum I/Os and the complete parameters set.
	Basic	The basic level is designed to configure the valve terminal without CP extension.
	CP_Extension	This level is designed to configure I/Os including the CP extension.
Festo CPX	Basic_DIO_only	The basic level is designed to configure the CPX with pneumatic valves and digital I/O only.
	Generic_DIO_AIÖ	The generic DS401 level is designed to configure CPX valves and I/Os, including analog I/O modules.
	Advanced	The advanced level is designed to configure the maximum I/Os and the complete parameters set.
All slaves except ATV and LXM05/32_MFB and LXM15L-P_V1_45	Default	This feature is the default function for certain devices. It may not be modified.



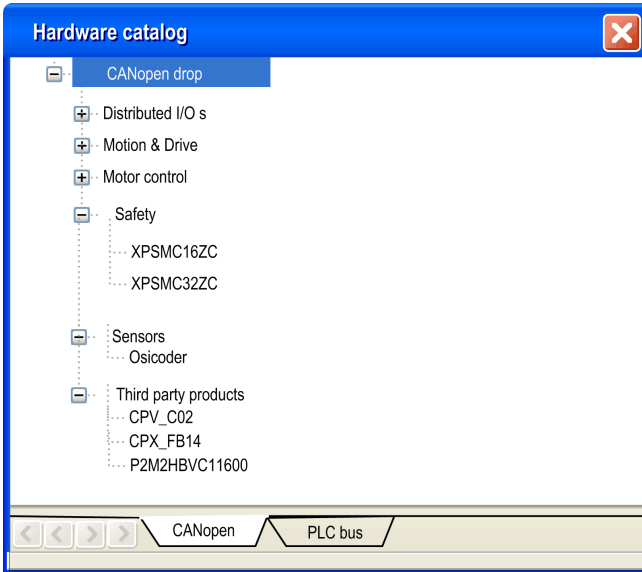
**NOTE:** Some devices can only handle one function. In this case, the function appears grayed out and cannot be modified.



## Configuration Using Control Expert

### Overview

Devices that can be configured using Control Expert are shown in the **Hardware Catalog**:



### Procedure

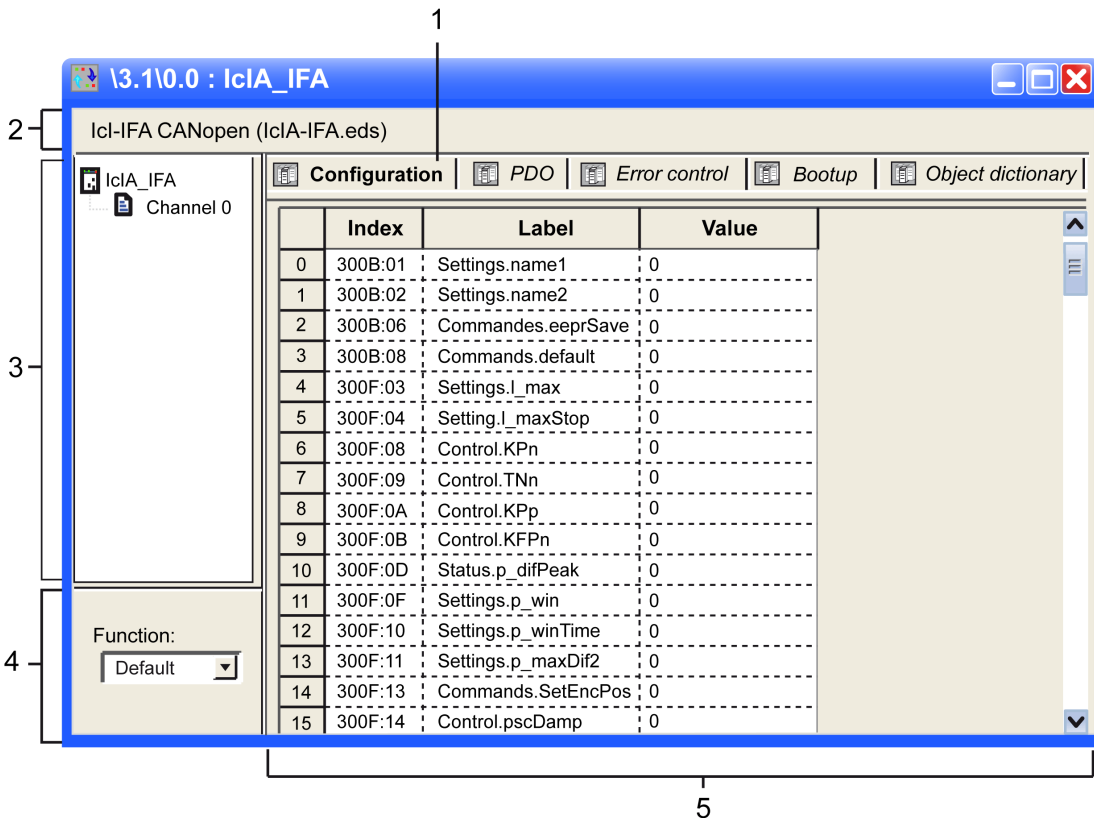
To configure a slave, follow these steps:

Step	Action
1	Access the CANopen bus configuration screen, page 52.
2	Double-click the slave to be configured.

Step	Action
3	Configure the usage function using the <b>Configuration</b> tab.
4	Configure the PDOs using the <b>PDO</b> tab.
5	Select the control of error detected using the <b>Error control</b> tab.
6	Configure the bootup procedure using the <b>Bootup</b> tab.
7	Integrate a third-party product using the <b>Object Dictionary</b> tab.

## Configuration Tab

This figure shows an example of the configuration window of a slave:



This table shows the elements of the configuration window and their functions:

Number	Element	Function
1	Tabs	Indicates the type of window displayed. In this case, it is the configuration window.
2	Module	Indicates the device shortened name.
3	CANopen communication	<p>Allows you to select the device and display the <b>Overview</b> tab. It gives the characteristics of the device.</p> <p>Allows you to select the channel and display these tabs:</p> <ul style="list-style-type: none"> <li>• <b>Configuration</b></li> <li>• <b>PDO</b> (input/output objects)</li> <li>• <b>Error control</b></li> <li>• <b>Bootup</b></li> <li>• <b>Object dictionary</b></li> </ul>
4	General parameters	Allows you to select the slave functions, page 62.
5	Configuration	<p>Allows you to set up the channels of the devices.</p> <p><b>NOTE:</b> Some devices can be configured with an external tool. In this case, the configuration is stored in the device and you cannot enter configuration parameters because this field is empty.</p>

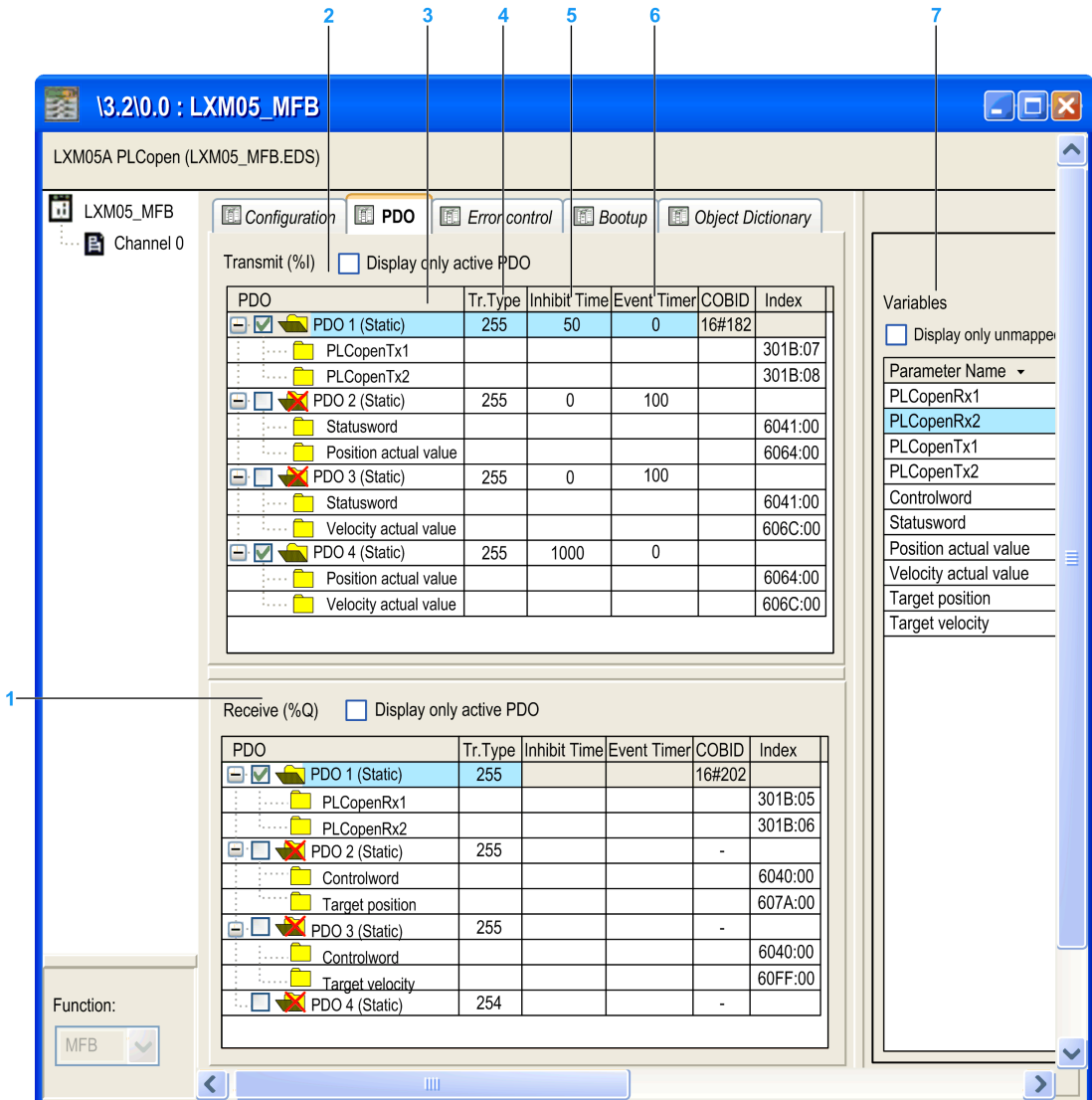
**NOTE:** All parameters are not sent when the device takes its configuration. The CPU sends only parameters that are different from the default values.

For more information on general, configuration, adjustment, and debugging parameters, refer to the documentation of each device.

## PDO Tab

PDOs make it possible to manage the communication flow between the CANopen master and the slaves. A PDO can be enabled or disabled. The **PDO** tab allows you to configure a PDO.

This figure shows an example of the **PDO** tab:



This table shows the elements of the **PDO** tab and their functions:

Number	Element	Function
1	<b>Receive (%Q)</b>	Information received by the slave from the master.
2	<b>Transmit (%I)</b>	Information transmitted by the slave to the master.

Number	Element	Function
3	<b>PDO</b>	According to the EDS file, some PDOs are already mapped. Otherwise, variables can be mapped to the PDOs.
4	<b>Tr.Type</b>	<p>The transmission type can be:</p> <ul style="list-style-type: none"> <li>• <b>Synchronous acyclic (0)</b>: the message is transmitted synchronously with the SYNC message but not periodically according to the value.</li> <li>• <b>Synchronous cyclic (1-240)</b>: the PDO is transmitted synchronously and cyclically. This value indicates the number of SYNC messages between 2 PDO transmissions.</li> <li>• <b>Asynchronous (Manuf. Event) (254)</b>: the PDO is transmitted asynchronously. It depends on the implementation in the device. Used for digital I/O.</li> <li>• <b>Asynchronous (Profile Event) (255)</b>: the PDO is transmitted asynchronously when the value changes.</li> </ul> <p><b>NOTE:</b> Verify that the configured transmission type is supported by the selected device.</p>
5	<b>InhibitTime</b>	Mask the communication during this time.
6	<b>Event Timer</b>	Time to manage an event in order to start a PDO.
7	<b>Variables</b>	<p>Variables can be mapped to the PDOs.</p> <p>To assign a variable to a PDO, drag and drop the variable into the desired PDO. A variable cannot be assigned with a static PDO.</p>

**NOTE:** Double-click the element to edit the value of parameters.

To configure the STB NCO 1010, it is necessary to determine all the objects that are valid for this device and to configure them manually in the PDOs. For more information about the list of the associated objects, refer to the *STB user manual*.

## PDO Multi-Mapping

The BMECXM master allows the PDO multi-mapping. It is possible to configure the same CANopen object in two different PDOs:

If an object is mapped...	Then ...
In more than one RPDO of a node	It is not supported.
In an RPDO and in a TPDO of a node	It is represented by two network variables in both process images.
In more than one TPDO of a node	Each instance is linked to the same network variable.

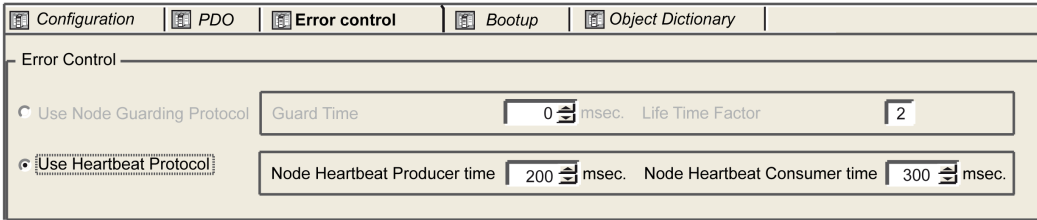
The PDO configuration is checked at build time. If there is an error detected:

- A log is displayed in the **Rebuild All Project** window at the bottom of the screen
- A dialog box with an error detected message appears when validating

## Error Control Tab

Some CANopen slave devices only support either Heartbeat or Node Guarding. But for devices that support both Heartbeat and Node Guarding protocols, the only choice in Control Expert is the Heartbeat mechanism.

The **Error control** tab for CANopen slave modules allows you to configure monitoring:



If the **Node Heartbeat Producer time** value or **Guard Time** value is set to 0 (zero), then according to the CANopen expected behavior, a CANopen slave device disconnected or not present in the configuration will not be diagnosed.

<b>⚠ WARNING</b>
<p><b>UNEXPECTED DIAGNOSTIC BEHAVIOR</b></p> <ul style="list-style-type: none"> <li>• Do not set the <b>Guard Time</b> value to 0 (zero), when using the Node Guarding protocol.</li> <li>• Do not set the <b>Node Heartbeat Producer time</b> value to 0 (zero), when using the Heartbeat protocol.</li> </ul> <p><b>Failure to follow these instructions can result in death, serious injury, or equipment damage.</b></p>

This table shows the elements of the **Error control** tab and their functions:

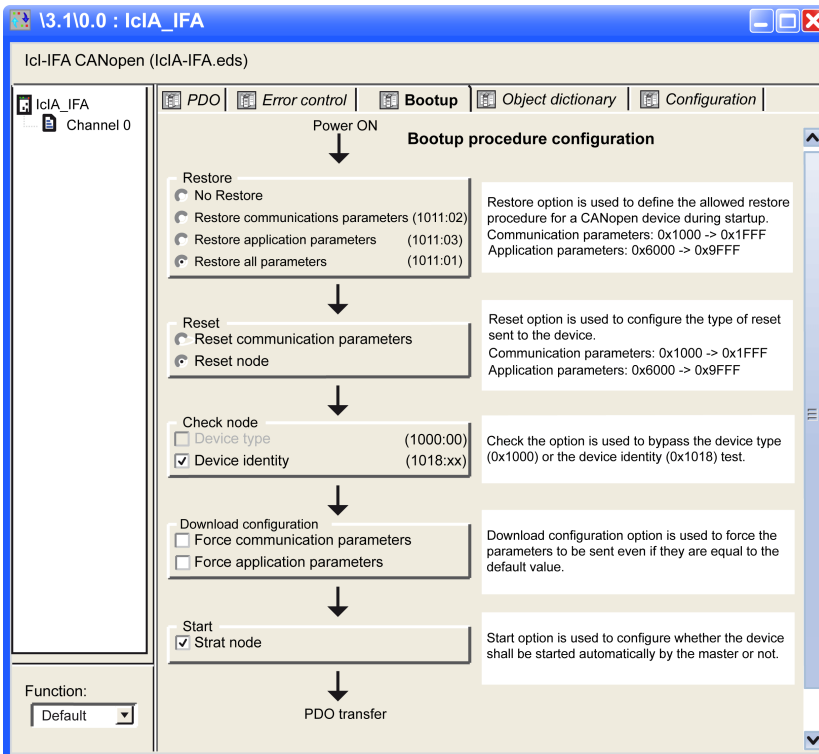
Protocol Used	Function
Node Guarding	<p>Monitoring of network nodes:</p> <ul style="list-style-type: none"> <li>• <b>Guard Time</b>: period when the NMT (Network Management) master sends an RTR (Remote Transmission Request) at regular intervals  <b>NOTE</b>: A value set to 0 deactivates the monitoring of the node.</li> <li>• <b>Life Time Factor</b> (read-only): 2.</li> </ul> <p>The concerned node answers in a given lapse time defined as below: lifetime = <b>Guard Time * Life Time Factor</b></p> <p><b>NOTE</b>: If there is no connection monitoring during the lifetime interval, the CANopen slave device signals an error.</p>
Heartbeat	<p>Mechanism that consists on sending cyclical presence messages generated by a heartbeat producer (CANopen slave) and the Heartbeat consumer (BMECXM) surveys the Heartbeat message reception.</p> <ul style="list-style-type: none"> <li>• <b>Node Heartbeat Producer time</b>: value corresponding to the sending time  <b>NOTE</b>: A value set to 0 deactivates monitoring by a consumer.</li> <li>• <b>Node Heartbeat Consumer time</b> (read-only): value corresponding to the reception time (default value set to 300 ms and cannot be modified).  <b>NOTE</b>: <ul style="list-style-type: none"> <li>• The values for the consumer must not be less than the values for the producer.  By default, <b>Node Heartbeat Consumer time</b> = 1.5 * <b>Node Heartbeat Producer time</b>.</li> <li>• If the BMECXM does not receive a signal within the time set for <b>Node Heartbeat Consumer time</b>, it generates a heartbeat event.</li> </ul> </li> </ul>

## Bootup Tab

<b>⚠ WARNING</b>
<p><b>UNEXPECTED EQUIPMENT OPERATION</b></p> <p>Manually verify all deactivated standard checks on the device before operating the system.</p> <p>Changing the default parameters of the <b>Bootup</b> tab bypasses standard system checks.</p> <p><b>Failure to follow these instructions can result in death, serious injury, or equipment damage.</b></p>

The goal of bootup procedure tab is to bypass the standard bootup procedure for devices that do not comply with CANopen standards.

The **Bootup** tab allows you to configure the bootup procedure:



This table defines the different functionalities of the **Bootup procedure configuration**:

Type		Functionality
<b>Restore</b>	<b>No Restore</b>	–
	<b>Restore communication parameters</b>	Enabled option according to the object 0x1011sub02. If checked, all parameters between 0x1000 to 0x1FFF are restored.
	<b>Restore application parameters</b>	Enabled option according to the object 0x1011sub03. If checked and if the device correctly implements the service, all application parameters are restored.
	<b>Restore all parameters</b>	Enabled option according to the object 0x1011sub01. If checked, all parameters are restored to default value.
<b>Reset</b>	<b>Reset communication parameters</b>	Option always enabled. If checked, all communication parameters are reset.
	<b>Reset node</b>	Option always enabled. If checked, all parameters are reset.



Type		Functionality
Check node	Device type	<p>If the device type identification value for the slave in object dictionary 0x1F84 is not 0x0000 ("don't care"), compares it to the actual value.</p> <p><b>NOTE:</b> If unchecked, this option forces the object dictionary 0x1F84 to 0x0000.</p>
	Device identity	<p>If the configured vendor ID in object dictionary 0x1F85 is not 0x0000 ("don't care"), read slave index 0x1018, sub-index 1 and compare it to the actual value.</p> <p>The same comparison is done for <b>ProductCode</b>, <b>RevisionNumber</b>, and <b>SerialNumber</b> according to object 0x1F86-0x1F88.</p> <p><b>NOTE:</b> If unchecked, this option forces the object dictionary 0x1F86-0x1F88 (sub device node ID) to 0x0000.</p>
Download Configuration	Force communication parameters	Forces the download of communication or configuration parameters (unchecked by default).
	Force application parameters	<p>If the option is:</p> <ul style="list-style-type: none"> <li>• Checked, it forces all the corresponding objects to be downloaded.</li> <li>• Unchecked, you have to follow these standard rules: parameters are downloaded if they are different from the default values or if they are forced in the object dictionary. In other cases, parameters are not downloaded.</li> </ul>
Start	Start node	<p>If the option is:</p> <ul style="list-style-type: none"> <li>• Checked (default value), the CANopen master starts the device automatically after the bootup procedure.</li> <li>• Unchecked, the device stays in pre-operational state after bootup procedure. In this case, the device has to be started by the application program.</li> </ul>

## Object Dictionary Tab

### **⚠ WARNING**

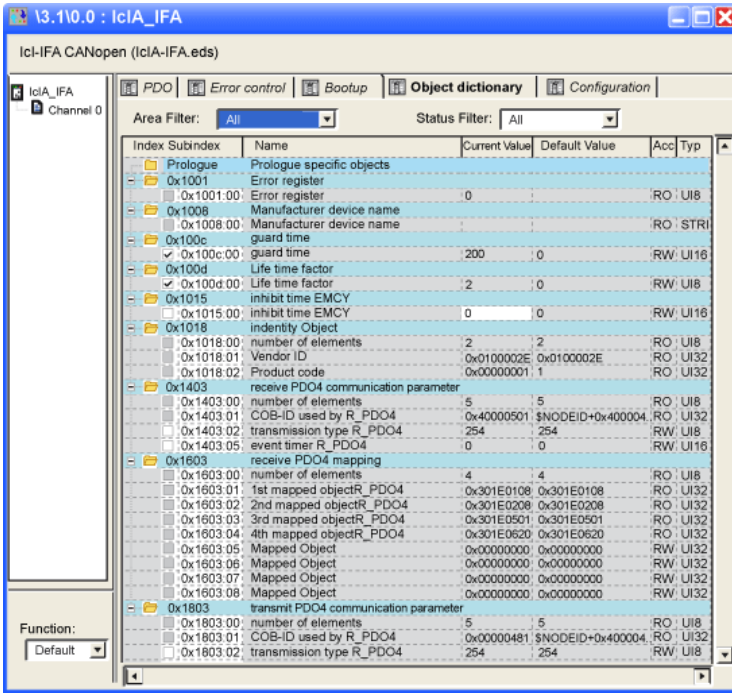
#### **UNEXPECTED EQUIPMENT OPERATION**

Manually verify all Object Dictionary values and mapping.

Changing the default values and mapping of the Object Dictionary table generates non-standard behavior of the equipment.

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

The **Object Dictionary** tab allows you to configure and integrate third-party products:



This table shows the elements of the **Object Dictionary** tab and their functions:

Element	Function
Parameter	If the check box associated to each parameter is: <ul style="list-style-type: none"> <li>Activated: force parameters to be transmitted even if they are unchanged.</li> <li>Deactivated: block parameters that do not need to be sent to the device.</li> </ul> <p><b>NOTE:</b> To prevent programming redundancies or conflicts, parameters that can be modified in <b>Configuration</b>, <b>PDO</b>, or <b>Error Control</b> tabs are grayed out.</p>
Current Value	Modify the current value of an object (except read-only objects) by typing a value in the box. By default, the object is sent if the current value is modified. You can block object sending by deactivating the check box.
Default Value	Set objects to a specific value just before (prolog) or just after (epilog) the standard bootup procedure.

Element	Function
<b>Area Filter</b>	<ul style="list-style-type: none"> <li>All: Show all area.</li> <li>Prolog / Epilog: Show only prolog and epilog projects</li> <li>XXXX...YYYY: Show only objects between XXXX to YYYY.</li> </ul>
<b>Status Filter</b>	<ul style="list-style-type: none"> <li>All: Show all objects.</li> <li>Configured: Show only transmitted objects to the device during bootup.</li> <li>Not configured: Show only not transmitted objects to the device.</li> <li>Modified: Show only objects from which values are different from default values</li> </ul>

You can drag and drop available objects from the index folder to the prolog or epilog section. If not, for example PDOs or read-only, a pop-up appears. Some functions are only available in prolog and epilog section.

**NOTE:** An object that has been put in the prolog or epilog section is always sent.

## Configuration Using an External Tool

### Overview

To configure a Lexium 32/32i, a Lexium ILA, ILE, ILS, a TeSys U, or an ATV device, it is necessary to use an external tool:

- Advantys Configuration Software for the STB
- SoMove software for the ATV32, ATV312, ATV12, ATV61, ATV71, ATV process, and the TeSys U
- SoMove software for the Lexium 32/32i,
- Lexium CT for Lexium ILA, ILE, ILS.
- EasyIcIA V1.104 for ICLA\_IFA, ICLA\_IFE, ICLA\_IFS.

**NOTE:** To facilitate the programming of the motion and drives devices, it is highly recommended to use the software with the Control Expert MFBs.

### File Access

If external tools generate an EDS or DCF file, you can integrate the products via the Hardware Catalog Manager (see EcoStruxure™ Control Expert, Hardware Catalog Manager, Operation Guide).

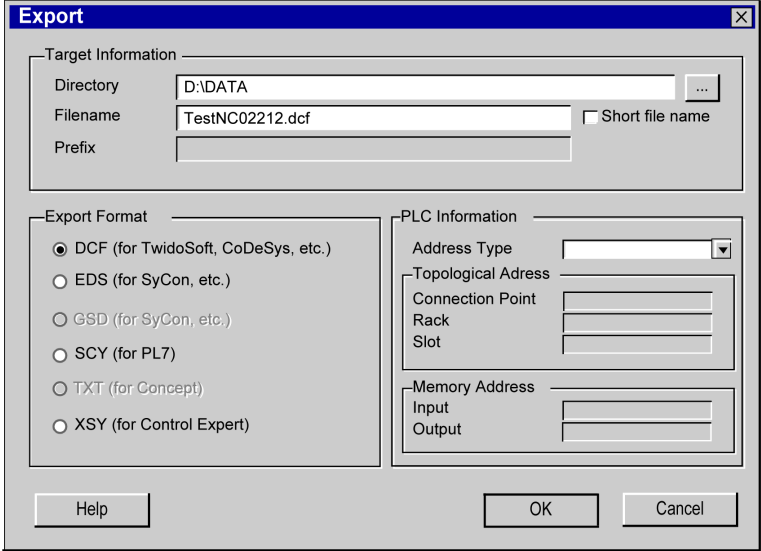
For Advantys STB distributed I/O modules, you can directly access the DCF file from Control Expert as shown below.

## Advantys Configuration Software

Advantys Configuration Software (Version 2.5 or higher) has to be used to configure an STB NCO 2212. The Advantys Configuration Software validates the configuration and creates a DCF file that contains all the objects used in the configuration ordered in the proper sequence. The DCF file can be imported from Control Expert.

**NOTE:** The creation of the DCF file is only possible from the full version of Advantys.

To add an island to a CANopen bus, follow these steps:

Step	Action
1	In Advantys Configuration Software (Version 2.2 or above), create a new Island.
2	Select the STB NCO 2212 network interface module.
3	Select the modules that are used in the application.
4	Configure the island.
5	<p>When the configuration is over, click <b>File/Export</b> to export the island in DCF format.</p> <p><b>Result:</b> The <b>Export</b> window is displayed:</p> 
6	Click <b>OK</b> to confirm.
7	Once the file is exported, launch Control Expert and open the project in which the island is used.
8	Add an STB device to the Bus Editor, page 54.
9	Right-click the STB device and click <b>Open the module</b> .

Step	Action
10	In the <b>PDO</b> tab, click the <b>Import DCF</b> button.
11	Click <b>OK</b> to confirm.  <b>Result:</b> The PDOs are configured automatically.

**NOTE:** Repeat this procedure to modify the topology of an island.

For more information about the STB configuration, refer to the STB user manual.

## Master Configuration

### Subject of this Section

This section presents the master configuration.

## CANopen Master Module Configuration Window

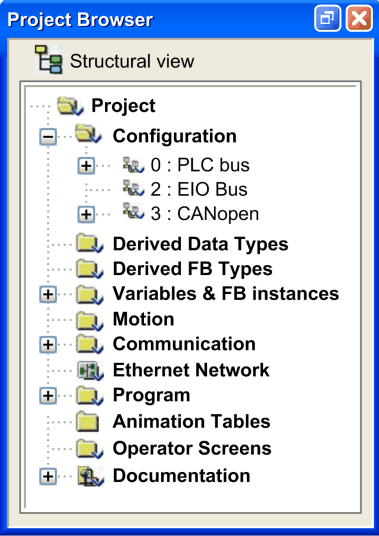
### Overview

The module configuration editor is used to configure the master of the CANopen bus.

**NOTE:** The device configuration procedure is valid when configuring a project with Control Expert Classic. When you configure your device from a system project, some commands are disabled in the Control Expert editor. In this case, you need to configure these parameters at the system level by using the Topology Manager.

### Procedure

To access the module configuration editor follow these steps:

Step	Action
1	<p>From the <b>Structural view</b> of the <b>Project Browser</b>, expand (+) the <b>Configuration</b> directory:</p> 
2	<p>Expand (+) the bus (<b>PLC bus</b> or <b>EIO Bus</b>) where is declared the CANopen X80 master module you want to configure.</p> <p><b>NOTE:</b> You can also right-click the bus CANopen you want to configure the master in the <b>Configuration</b> directory, and click <b>Go to Bus Master</b>.</p>
3	<p>Right-click the BMECXM module, and click <b>Open</b>.</p> <p><b>Result:</b> The module configuration window for the CANopen X80 master module is displayed.</p>

## Description

The module configuration window has three tabs:

- **Overview** (not editable)
- **Web : Main IP** (in online mode only)
- **Ethernet Configuration**

## Ethernet Configuration Tab

This table shows the elements of the **Ethernet Configuration** tab and their functions:

Element	Function
<b>Scanner configuration</b>	Provide the scanner, protocol, and profile of the scanner chosen at module insertion in Control Expert project. This is not editable.  <b>NOTE:</b> To change scanner association (assuming that the choice is available), the module needs to be deleted and inserted again with the new scanner association.
<b>IP/DHCP configuration</b>	Click <b>Update IP/DHCP configuration</b> hyperlink to access an <b>Ethernet Network</b> window where you can edit: <ul style="list-style-type: none"> <li>• The <b>IP Address</b>, <b>Subnet Mask</b>, and <b>Gateway Address</b>.</li> <li>• The <b>Device Name</b> in the <b>Identifier</b> column.</li> </ul>
<b>Other configuration</b>	Click <b>Go to DTM configuration</b> for a direct access to the DTM user interface, page 87.

## Web : Main IP Tab

When online, this tab displays the web pages directly for diagnostics purpose, page 134.

**NOTE:** To go online, refer to DTM connections, page 85.

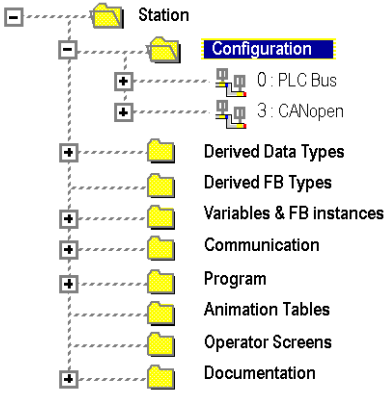
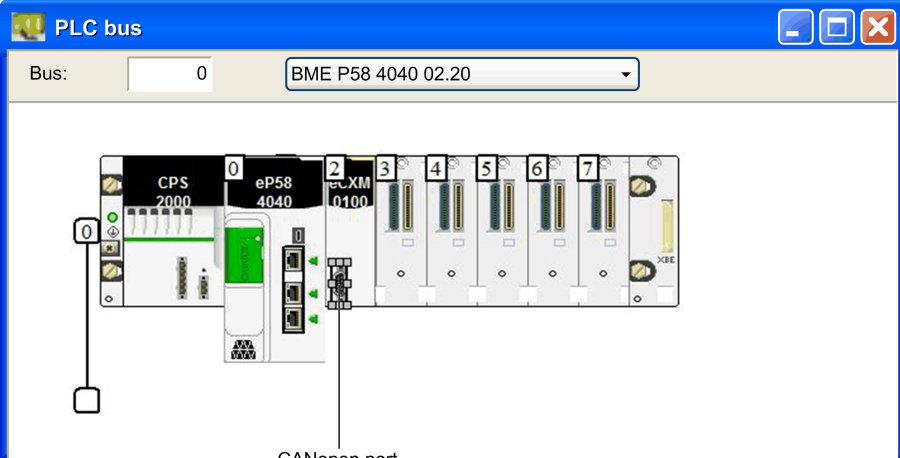
## CANopen Master Port Configuration Screen

### Overview

The CANopen master port configuration screen is used to declare and configure the master of the CANopen network from a Modicon M580 PLC station.

### Procedure

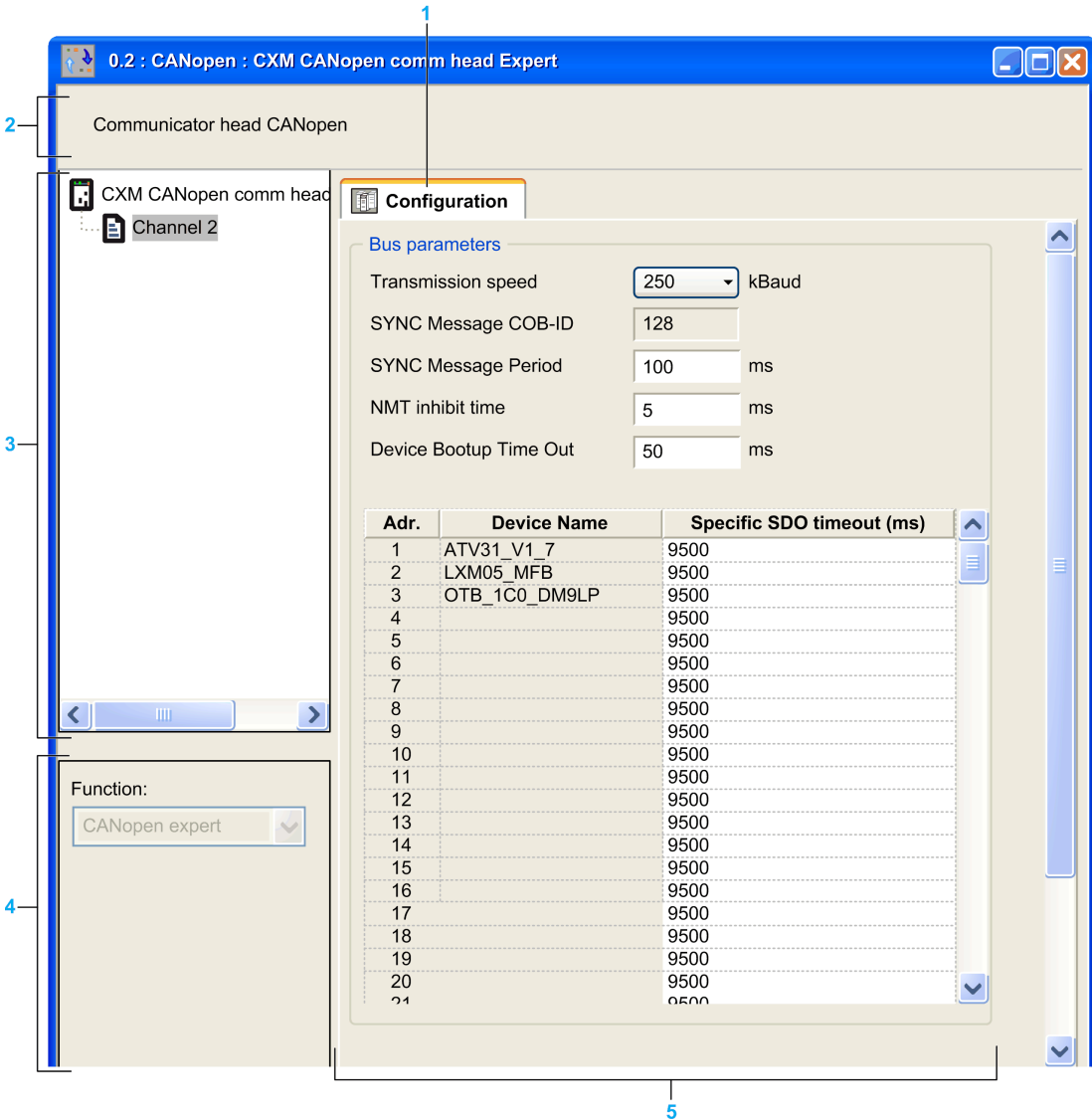
To access the CANopen master port configuration screen of a M580, follow these steps:

Step	Action
1	<p>From the <b>Project Browser</b>, expand (+) the <b>Configuration</b> directory:</p>  <p>The screenshot shows a tree view of a project. The 'Configuration' folder is selected and expanded, showing subfolders: '0 : PLC Bus', '3 : CANopen', 'Derived Data Types', 'Derived FB Types', 'Variables &amp; FB instances', 'Communication', 'Program', 'Animation Tables', 'Operator Screens', and 'Documentation'.</p>
2	<p>Double-click the <b>PLC Bus</b> subdirectory.</p> <p><b>Result:</b> The <b>PLC Bus</b> window is displayed.</p>
3	<p>Double-click the CANopen port of the BMECXM module:</p>  <p>The screenshot shows a window titled 'PLC bus'. It has a 'Bus:' field with '0' and a dropdown menu showing 'BME P58 4040 02.20'. Below is a diagram of a PLC rack with modules: 'CPS 2000', 'eP58 4040', and 'BMECXM 0100'. The 'BMECXM 0100' module has a 'CANopen port' indicated by a line pointing to its connector. The rack also shows slots 3, 4, 5, 6, 7, and XBE.</p> <p><b>Result:</b> The CANopen master configuration window is displayed.</p>



## Description

This figure shows an example of the CANopen master configuration window:



This table shows the elements of the CANopen master configuration window and their functions:

Number	Element	Function
1	Tab	Indicates the type of window displayed. In this case, it is the configuration window.
2	Module	Indicates the abbreviated heading of the BMECXM.
3	CANopen communication	Allows you to select: <ul style="list-style-type: none"> <li>The device and display the <b>Overview</b> tab that gives the characteristics of the BMECXM.</li> <li>The channel and display the <b>Configuration</b> tab that enables you to declare and configure the CANopen master.</li> </ul>
4	General parameters	Indicates the slave functions, page 62.
5	Configuration	Allows you to configure the parameters of the CANopen bus.

## Bus Parameters

The CANopen master port configuration screen allows the configuration of these bus parameters:

Bus parameters	Default value	Comments
<b>Transmission speed</b>	250 kBaud	Enables you to select in a dropdown list: <ul style="list-style-type: none"> <li>1000</li> <li>500</li> <li>250</li> <li>125</li> <li>50</li> <li>20</li> </ul>
<b>SYNC Message COB-ID</b>	128	The BMECXM send SYNC message using COB-ID 0000 0080 hex.
<b>SYNC Message Period</b>	100 ms	Define the interval period between two SYNC messages.
<b>NMT inhibit time</b>	5 ms	During bootup, the CANopen master implements a delay between each NMT message to avoid slave overload. The value has to be given in multiple of 100 $\mu$ s. <b>NOTE:</b> The value 0 disables the inhibit time.

<b>Bus parameters</b>	<b>Default value</b>	<b>Comments</b>
<b>Device Bootup Time Out</b>	50 ms	The global SDO timeout for the master is related to the scanning of the network. During this time, the BMECXM reads the object 1000 hex of each slaves devices to analyze the configuration of the CANopen fieldbus.
<b>Specific SDO timeout (ms)</b>	9500 ms	An individual SDO timeout is necessary for slave devices with long response times that are for accesses to the objects 1010 hex, 1011 hex, 1F50 hex.  The grid displays the node ID, the name, and the SDO timeout value for each present CANopen slave device.

# Ethernet Services Configuration

## What's in This Chapter

DTM Browser .....	84
DTM User Interface.....	87
<b>Ethernet IO</b> Tab .....	92
<b>Security</b> Tab .....	96
<b>SNMP</b> Tab.....	98
<b>NTP</b> Tab.....	100

## Introduction

The configuration of the Ethernet services is done in the BMECXM module DTM.

## DTM Browser

### Overview

Control Expert incorporates the Field Device Tool (FDT)/Device Type Manager (DTM) approach to integrate distributed devices with your process control application.

For details, refer to chapter *FDT Container* (see EcoStruxure™ Control Expert, Operating Modes).

### Procedure

To access the DTM configuration for the BMECXM module in the Control Expert **DTM Browser**, follow these steps:

Step	Action
1	Click <b>Tools &gt; DTM Browser</b> to open the <b>DTM Browser</b> .
2	In the <b>DTM Browser</b> , double-click the name that you assigned to the BMECXM module. <b>Result:</b> The <b>DTM Configuration</b> window is displayed.
3	In the <b>Configuration</b> window, select: <ul style="list-style-type: none"> <li>• <b>General Settings</b> <ul style="list-style-type: none"> <li>◦ IP/DHCP</li> <li>◦ Ethernet IO, page 92</li> <li>◦ Security, page 96</li> <li>◦ SNMP, page 98</li> <li>◦ NTP, page 100</li> </ul> </li> <li>• <b>CANopen Devices</b>, page 90</li> </ul>

## DTM Connections

To connect or disconnect a DTM and the BMECXM module in the **DTM Browser**, follow these steps:

Step	Action
1	Select the DTM that you want to connect to or disconnect from.
2	Right-click and select: <ul style="list-style-type: none"> <li>• <b>Connect</b><sup>(1)</sup> to monitor and diagnose the real-time operation of the device, or</li> <li>• <b>Disconnect</b><sup>(1)</sup></li> </ul> <p><b>NOTE:</b> A connected DTM appears in <b>bold</b> text. The <b>Connect</b> command is available only for disconnected DTMs.</p>
(1) You can also access the <b>Connect</b> and <b>Disconnect</b> commands in the Control Expert <b>Edit</b> menu.	

## DTM Types

The **DTM Browser** displays a hierarchical list of DTM nodes on a connectivity tree. The DTM nodes that appear in the list have been added to your Control Expert project. Each node represents an actual module or device in your Ethernet network.

In the **DTM Browser**, two kinds of DTMs are instantiated automatically:

- Master (communication) DTM: This DTM is both a device DTM and a communication DTM. The master DTM is a pre-installed component of Control Expert.

- **BMECXM DTM:** This is a device DTM that allows the configuration of the CANopen bus.

## DTM Names

Each DTM has a default name when it is inserted into the browser. The name is limited to 26 characters.

This table describes the components of the default name:

Element	Description
<b>Channel</b>	This is the name of the channel communication medium into which the device is plugged. This name is read from the DTM and is set by the device vendor. For example: EtherNet/IP
<b>Address</b>	This is the bus address of the device that defines the connection point on its parent gateway network. For example: the device IP address
<b>DTM name</b>	The default name is determined by the vendor. You can edit the name in the <b>DTM Browser</b> by clicking the DTM node or in the <b>Ethernet Configuration</b> tab of the CANopen master module screen, page 77.

**NOTE:** The DTM name is different from the **Device Name** that is used to get the IP address of the BMECXM0100 module in the **IP/DHCP** tab.

The default name for gateway and device DTMs is in the format `<BusName><BusNumber>_d<DropNumber>_r<RackNumber>_s<SlotNumber>_<PartNumber>_[SubSetName]`. The name of the subset is optional.


As an example on a local rack, a BMECXM0100 module on PLC bus, drop 0, rack 0, and slot 2, has this name: `PLC0_d0_r0_s2_ECXM0100`.

As an example on a remote drop, a BMECXM0100 module on EIO bus, drop 3, rack 1, and slot 4, has this name: `EIO2_d3_r1_s4_ECXM0100`.

## DTM Status

The **DTM Browser** contains graphics to indicate the status of each DTM in the connectivity tree:

Status	Description
Built / Not-built	A blue check mark is superimposed on a device icon to indicate that the node/subnode is not built. This means that some property of the node has changed. Information stored in the physical device is thus no longer consistent with the local project.
Connected /	A connected DTM appears in <b>bold</b> text.

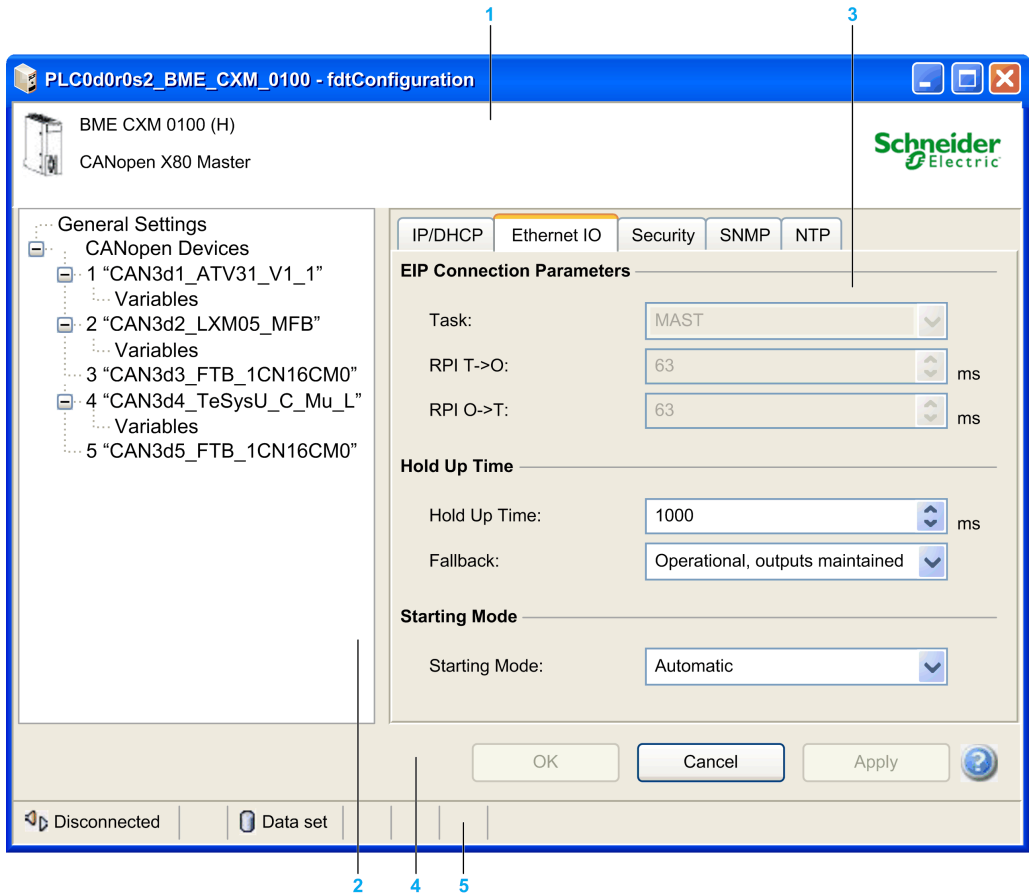
Status	Description
Disconnected	<ul style="list-style-type: none"><li>Connecting a DTM to its physical device automatically connects all higher-level parent nodes up to the root node.</li><li>Disconnecting a DTM from its physical device automatically disconnects all its lower-level child nodes.</li></ul> <p><b>NOTE:</b> Connecting or disconnecting a DTM to or from its device does not also connect or disconnect Control Expert to or from the device. DTMs can be connected/disconnected while Control Expert is either offline or online.</p>
Installed / Not-installed	A red <b>X</b> is superimposed on a device icon to indicate that the DTM for that device is not installed on the PC.
Ethernet ready	A  is superimposed on a device icon to indicate that the device is an Ethernet ready equipment. This means that the equipment provides additional services compared to standard EIP or Modbus equipment. For more information, refer to Ethernet Ready Equipment (see EcoStruxure™ Control Expert, Operating Modes).

## DTM User Interface

### General Layout

You can configure some optional parameters for the BMECXM module through the DTM user interface in Control Expert.

This figure shows the DTM user interface:




- 1 Identification area
- 2 Navigation area
- 3 Application area
- 4 Action area
- 5 Status bar

## Description

This table describes the different areas that make up the DTM user interface:



Element	Function
Identification area	Shows the device type name and product name.
Navigation area	The device tree contains: <ul style="list-style-type: none"> <li>• <b>General Settings</b> node, page 89</li> <li>• <b>CANopen Devices</b> node, page 90 with a device list that includes a subnode for each device</li> </ul>
Application area, page 91	Contains parameters, most of them being editable.
Action area	Contains these buttons: <ul style="list-style-type: none"> <li>• <b>OK</b>: Save your changes and close the page.</li> <li>• <b>Cancel</b>: Cancel changes.</li> <li>• <b>Apply</b>: Save your changes and keep the page open.</li> <li>• : Open an online help page.</li> </ul> <p><b>NOTE</b>: Your changes take effect only when they are successfully downloaded from your PC to the CPU and from the CPU to the BMECXM modules and network devices.</p>
Status bar, page 91	Shows status information with: <ul style="list-style-type: none"> <li>• Connection state icons</li> <li>• Additional icons</li> </ul>

## Navigation Area: General Settings Node

The **General Settings** node contains 5 tabs:

Tab	Function
<b>IP/DHCP</b>	These read-only parameters are displayed: <ul style="list-style-type: none"> <li>• <b>IP Settings</b>: IP address, subnet mask, and default gateway</li> <li>• <b>Rack and Slot Information</b>: rack ID, slot number, and device name</li> </ul> <p><b>NOTE</b>: These parameters can be changed in Control Expert.</p>
<b>Ethernet IO</b> , page 92	These parameters are displayed: <ul style="list-style-type: none"> <li>• <b>EIP Connection Parameters</b>: <ul style="list-style-type: none"> <li>◦ Task: MAST (by default), FAST, or AUX.</li> <li>◦ RPI T-&gt;O: requested packet interval of the consumption connection<sup>(1)</sup>.</li> <li>◦ RPI O-&gt;T: requested packet interval of the production connection<sup>(1)</sup>.</li> </ul> </li> <li>• <b>Hold Up Time</b>: hold-up time and fallback</li> <li>• <b>Starting Mode</b>: automatic (by default) or manual</li> </ul>
<b>Security</b> , page 96	These parameters are displayed:

Tab	Function
	<ul style="list-style-type: none"> <li>• <b>Global Policy:</b> to enforce or unlock security.</li> <li>• <b>Services:</b> FTP, HTTPS, SNMP and EIP to be enabled/disabled.</li> <li>• <b>Access Control:</b> to enable/disable Ethernet access to the EtherNet/IP server.</li> </ul>
SNMP, page 98	These parameters are displayed: <ul style="list-style-type: none"> <li>• <b>IP Address Managers:</b> IP addresses.</li> <li>• <b>Agent:</b> location and contact, and SNMP manager to be enabled/disabled.</li> <li>• <b>Community Names:</b> set, get, and trap.</li> <li>• <b>Security:</b> authentication failure trap to be enabled/disabled.</li> </ul>
NTP, page 100	The <b>NTP Client Configuration</b> parameter is displayed: NTP, IP addresses, and polling period.
(1) The refresh period for this connection is in ms (2...2550).	

## Navigation Area: CANopen Devices Node

When clicking the **CANopen Devices** node, a **Device List** tab shows all slave devices with read-only parameters: **Address**, **Device Name**, **Vendor**, **Type**, and **Version**.

When expanding (+) the **CANopen Devices** node and clicking a slave device, the **I/O** tab is displayed with 2 fields:

Field	Function
<b>IO Structure Name</b>	These parameters are displayed: <ul style="list-style-type: none"> <li>• <b>Structure Name</b> (read-only): name assigned to the device input structure from the imported CANopen configuration</li> <li>• <b>Variable Name</b> (editable)</li> </ul> <p><b>NOTE:</b> The variable name is reinitialized when clicking the <b>Default Name</b> button or if the structure name has been changed.</p>
<b>Variables Management</b>	The <b>Import Mode</b> (read-only): displays the automatic option only.

Each slave device contains a **Variables** subnode.

When expanding (+) a slave device and clicking the **Variables** subnode, these tabs are displayed, when relevant, with read-only parameters:







Tab	Function
<b>Input</b>	Displays <b>Type</b> , <b>Offset</b> , <b>Name</b> , and <b>Comment</b> columns.
<b>Input(bit)</b>	Displays <b>Offset</b> , <b>Position</b> , <b>Name</b> , and <b>Comment</b> columns.

Tab	Function
Output	Displays <b>Type</b> , <b>Offset</b> , <b>Name</b> , and <b>Comment</b> columns.
Output(bit)	Displays <b>Offset</b> , <b>Position</b> , <b>Name</b> , and <b>Comment</b> columns.

**NOTE:** You cannot add, delete, or edit variables.

In these tabs, all the bits or bytes from the I/O modules of the CANopen device are displayed.

At the beginning of each row, the data type is visually indicated:




Data types	BOOL	BYTE	INT, UINT, WORD	DINT, UDINT, DWORD	REAL	String
Visual representation						

**NOTE:** Icons are left blank if bits and bytes are unused.

## Application Area

When you edit a parameter, Control Expert displays an icon next to the field you are editing and in the navigation tree.






These icons refer to the value of the parameter that is being edited:

Icon	Description
	The entered value is not known. The <b>Apply</b> button does not work until a correct value is entered.
	The entered value is not valid. The <b>Apply</b> button does not work until a valid value is entered.
	This parameter has changed. The <b>Apply</b> button does not work until the value is corrected.





**NOTE:** You can move the mouse over the icons to display tooltips.

## Status Bar

This table shows the connection state icons of the status bar and their functions:

Icon	Element	Function
	Disconnected	The DTM is offline.
	Connected	The DTM is online.
	Connecting	The DTM is connecting.
	Disconnecting	The DTM is disconnecting.
	Communication problem	There are communication errors detected, for example wrong IP address.

This table shows the additional icons of the status bar and their functions:

Icon	Element	Function
	Communication in progress	The DTM communicates with the device.
	Data set	Displays the offline data stored in the DTM.
	Device	Displays the offline data stored in the device itself.
	Summary	Displays the summary state of the data when at least one parameter value has been modified (none by default).

## Ethernet IO Tab

### Overview

The **Ethernet IO** tab allows you to configure these parameters in Control Expert:

- **Task**
- **RPI**

- **Hold up time**
- **Fallback**
- **Starting Mode**

## Task Applications

An M580 CPU can execute single-task and multi-task applications. A single-task application only executes the MAST task. A multi-task application defines the priorities of each task.

There are four tasks available:

- MAST
- FAST
- AUX0
- AUX1

Each BMECXM module, is linked to a unique PLC task.

Tasks depend on the profile that is set by Control Expert. For the BMECXM module, there are two profiles:

- Remote (RIO scanner) where you can edit all the tasks
- Distributed (DIO scanner) where you can edit only MAST tasks

For more information on tasks, refer to the chapter on Application Program Structure (see EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual).

## Modicon M580 Task Characteristics

The time model and task period are defined as follows:

Task	Time model	Task period (ms)	
		Range	Default value
MAST <sup>(1)</sup>	Cyclic <sup>(2)</sup> or periodic	1...255	20
FAST	Periodic	1...255	5
AUX0	Periodic	10...2550 by 10	100

Task	Time model	Task period (ms)	
		Range	Default value
AUX1	Periodic	10...2550 by 10	200
<p>(1) MAST task is mandatory.</p> <p>(2) When set to cyclic mode, the minimum cycle time is 8 ms if there is a RIO network and 1 ms if there is no RIO network in the system.</p>			

## RPI Values

The Requested Packet Interval (RPI) depends on the profile that is set by Control Expert at instantiation time. For the BMECXM module, there are two profiles:

- Remote (RIO scanner) where you cannot change the RPIs
- Distributed (DIO scanner) where you can change the RPIs

There are two RPI values:

- T->O for process inputs
- O->T for process outputs

When the BMECXM module is the scanned by the RIO scanner, the RPI values are as follows:

Task Configuration (Period)	RPI			
	Process	Calculated Value...	Minimum <sup>(2)</sup>	Maximum <sup>(2)</sup>
Periodic (>0)	T->O	1/2 of PLC task period (rounded down ms)	2 ms	255 ms
	O->T	1.1 of the PLC task period (rounded up ms)	5 ms	
Cyclic <sup>(1)</sup> (=0)	T->O	1/4 of PLC task watchdog	3 ms	
	O->T	1/4 of PLC task watchdog	5 ms	
<p>(1) For the MAST task only</p> <p>(2) Linked to the task configuration.</p>				

When the BMECXM module is the scanned by the DIO scanner a distributed, the RPI values are as follows:

Mast Task Configuration (Period)	RPI			
	Process	Default Value...	Minimum <sup>(1)</sup>	Maximum <sup>(1)</sup>
Periodic (>0)	T->O	2 times of PLC task period	1/2 of PLC task period (rounded down ms)	1500 ms
	O->T	2 times of PLC task period	1.1 of the PLC task period (rounded up ms)	
Cyclic (=0)	T->O	1/4 of PLC task watchdog	2 ms	255 ms
	O->T	1/4 of PLC task watchdog	5 ms	

(1) Linked to the mast task configuration.

**NOTE:** Outputs that have changed are published at the end of each PLC scan. For the other ones, the default value applies.

## Hold Up Time

The hold up time represents the timeout of input reception to switch to FALLBACK state. By default the **Hold Up Time** = 4 x **Watchdog**.

You can modify the **Hold Up Time** value if you set a value greater than the default value and lower than 5 seconds.

**NOTE:** For the application, the maximum value is 5 s even if the default or set value is greater than 5 s.

## Fallback

The fallback information defines the behaviors of the device:

- **Operational, outputs maintained:** it maintains outputs. The values are kept.
- **Operational, outputs set to 0:** it resets outputs. The values are set to 0.
- **Stop:** the CANopen bus is in STOP state.

**NOTE:** For more information, refer to Fallback Strategy, page 44.

## Starting Mode

### ▲ WARNING

#### UNEXPECTED EQUIPMENT BEHAVIOR

Do not use *RIO\_CTRL* and *DIO\_CTRL* control bits of the M580 CPU device DDT to start or stop the scan of the BMECXM.

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

You can select:

- **Automatic.** The scan is started by the application.
- **Manual.** The BMECXM needs the *EM\_Start* command, page 160 from the PLC to go to the *CONNECTED RUN* state and to set the CANopen field bus to *OPERATIONAL*.

**NOTE:** For more information, refer to the chapter *Operating Modes*, page 40.

## Security Tab

### Overview

The **Security** tab allows you to configure the security level of the services.

The default settings represent the maximum level of security level. The increased security reduces the communication capabilities and the access to communication ports.

**NOTE:** For general information about security, refer to the **Modicon Controllers Platform, Cybersecurity, Reference Manual**.

### Properties

This table describes the properties of the **Security** tab:

Parameter		Description
Global Policy	Enforce Security	Reset all the services to the default settings and implement the highest level of security.
	Unlock Security	Use the lowest level security settings (opposite of default settings).



Parameter		Description
<b>Services</b>	FTP	Enable or disable (default) firmware upgrade.
	HTTPS	Enable or disable (default) the Web access service.
	SNMP	Enable or disable (default) the diagnostic access service.
	EIP	Enable or disable (default) the diagnostic access service for exchanging I/Os and diagnostic information with the CPU.
<b>Access Control</b>	<b>Enabled</b> (by default)	Ethernet services access authorized for the listed addresses.
	<b>Disabled</b>	There is no restriction on which network devices can access EtherNet services. The BMECXM module accepts EtherNet/IP requests from any device.

## Services

For security reasons, all the communication ports of the BMECXM module are disabled by default.

If EIP service is disabled, exchanging with the CPU is not possible. Therefore, you have to enable at least the EIP service in the **Security** tab so that the scanner can access the BMECXM module.

If FTP service is disabled, an FTP upgrade is not possible.

Set the **Security** tab parameters before downloading the application to the CPU.

**NOTE:** Schneider Electric recommends disabling services that are not being used.

## Enabling Access Control

When the **Access Control** is enabled, it restricts access to the BMECXM module services declared in the list.

In the box, you can add the IP addresses of:

- The BMECXM module with **Subnet** set to **Yes** that allows any device in the subnet to communicate with the module using EtherNet/IP.
- Any client device that may send a request to the BMECXM module, which in this case acts as an EtherNet/IP server.
- Your maintenance PC that communicates with the BMECXM module via Control Expert to configure and diagnose your application and view the module web pages.

You must ensure that the corresponding CPU scanner address is filled in the list of authorized addresses.

**NOTE:** Using the BMECXM module in RIO/DIO requires to add the corresponding RIO/DIO scanner IP address in the Access Control list (ACL).

## Adding Devices to the Access Control List

**NOTE:** Before declaring a new address in the list, you must enable the corresponding service in the section [Services](#), page 96.

To add devices, follow these steps:

Step	Action
1	Enable the <b>Access Control</b> .
2	Click <b>Add</b> .
3	<p>Enter the address of the device to access the BMECXM module with either of these methods:</p> <ul style="list-style-type: none"> <li>• <i>Add a single IP address:</i> Enter the IP address of the device and select <b>No</b> in the <b>Subnet</b> column.</li> <li>• <i>Add a subnet:</i> Enter a subnet address in the <b>IP Address</b> column. Select <b>Yes</b> in the <b>Subnet</b> column. Enter a subnet mask in the <b>Subnet Mask</b> column.</li> </ul> <p><b>NOTE:</b> The subnet in the <b>IP Address</b> column can be the subnet itself or any IP address in the subnet. If you enter a subnet without a subnet mask, an on-screen message states that the modification cannot be validated.</p> <p><b>NOTE:</b> A red exclamation point (!) indicates a detected error in the entry. You can save the configuration only after the detected error is addressed.</p>
4	Select one or more of the following methods of access you are granting the device or subnet: FTP, HTTPS, SNMP, EIP.
5	<p>Repeat these steps 2 to 4 for each additional device or subnet to which you want to grant access to the BMECXM module.</p> <p><b>NOTE:</b> You can enter up to 128 authorized IP addresses or subnets.</p>
6	Click <b>Apply</b> .

**NOTE:** You can remove a device by selecting its IP address and clicking **Delete**.

## SNMP Tab

### Overview

The **SNMP** tab allows you to configure the SNMP agent in Control Expert.

## SNMP Agent

The BMECXM module includes an SNMP agent. An SNMP agent is a software component running on the BMECXM module.

It allows access to the diagnostic and management information of the module via the SNMP service. The SNMP agent can communicate with up to 2 SNMP managers as part of an SNMP service. SNMP browsers, network management software, and other tools typically use SNMP to access this data.

In addition, the SNMP agent can be configured with the IP address of up to two devices (typically PCs running network management software) to be the target of event driven trap messages. These trap messages inform the management device of events such as cold start and unauthorized access.

## Properties

This table describes the properties of the **SNMP** tab:

Group/Parameter		Description
<b>IP Address Managers</b>	<b>IP Address Manager 1</b>	The IP address of the first SNMP manager to which the SNMP agent sends trap messages.
	<b>IP Address Manager 2</b>	The IP address of the second SNMP manager to which the SNMP agent sends trap messages.
<b>Agent</b>	<b>Location (SysLocation)</b>	The device location (32 characters maximum)
	<b>Contact (SysContact)</b>	Information describing the person to contact for device maintenance (32 characters maximum).
	<b>Enable SNMP Manager</b>	<ul style="list-style-type: none"> <li>Check the box: you cannot edit the <b>Location</b> and <b>Contact</b> settings on this page. Those settings are managed by the SNMP Manager.</li> <li>Uncheck the box: you can edit the <b>Location</b> and <b>Contact</b> settings on this page.</li> </ul>
<b>Community Names</b>	<b>Set<sup>(1)</sup></b>	Password required by the SNMP agent before executing write commands from an SNMP manager.
	<b>Get<sup>(2)</sup></b>	Password required by the SNMP agent before executing read commands from an SNMP manager.
	<b>Trap<sup>(3)</sup></b>	Password required by the SNMP agent before the manager accepts trap messages from the agent.

Group/Parameter		Description
<b>Security</b>	<b>Enable "Authentication Failure" Trap</b>	Check the box: the SNMP agent sends a trap message to the SNMP manager if an unauthorized manager sends a <b>Get</b> or <b>Set</b> command to the agent.
(1) <b>private</b> by default		
(2) <b>public</b> by default		
(3) <b>alert</b> by default		

## NTP Tab

### Overview

The **NTP** tab allows you to configure the NTP Control Expert.

### Properties

This table describes the properties of the **NTP** tab:

Parameter		Description
<b>NTP Client Configuration</b>	<b>NTP</b>	To be configured when syslog event logging is configured in Control Expert via <b>Tools &gt; Project Settings &gt; General &gt; PLC diagnostics</b> .
	<b>Primary IP address</b>	The IP address of the first NTP server
	<b>Secondary IP address</b>	The IP address of the second NTP server
	<b>Polling Period</b>	The polling period is number of seconds (1...120, default = 20) between updates from the NTP server. A smaller polling period results in better accuracy.

# Language Objects

## What's in This Chapter

Implicit Process Data Exchange .....	101
Device DDT Variables .....	102

## Introduction

This chapter describes implicit messages, which are mapped in the device DDT, and emergency messages that are associated to the BMECXM modules.

# Implicit Process Data Exchange

## Overview

Use implicit messaging to create a communication link between the BMECXM module and the CPU.

## Description

The BMECXM module supports implicit exchange with:

- The CPU through the EtherNet/IP protocol
- The CANopen slaves through PDOs

Implicit EtherNet/IP messages are automatically exchanged at each cycle of the task associated with the module.

Implicit messages concern the status of the slave and BMECXM modules, and the slave process data. Those implicit messages are mapped to the device DDT.

# Device DDT Variables

## Overview

Device derived data type (device DDT) are used to access slave process data and to read/write the data of the BMECXM modules.

There is:

- One device DDT for each CANopen slave with input and output data.  
CANopen slave device DDTs are automatically created when building the application. The device DDTs are built with the list of variables exchanged by the PDOs and with a *HEALTH BYTE* showing the slave status.  
For more information, refer to the **PDO** tab, page 67.
- One device DDT for each BMECXM module. It is automatically created at insertion of the module in the project.

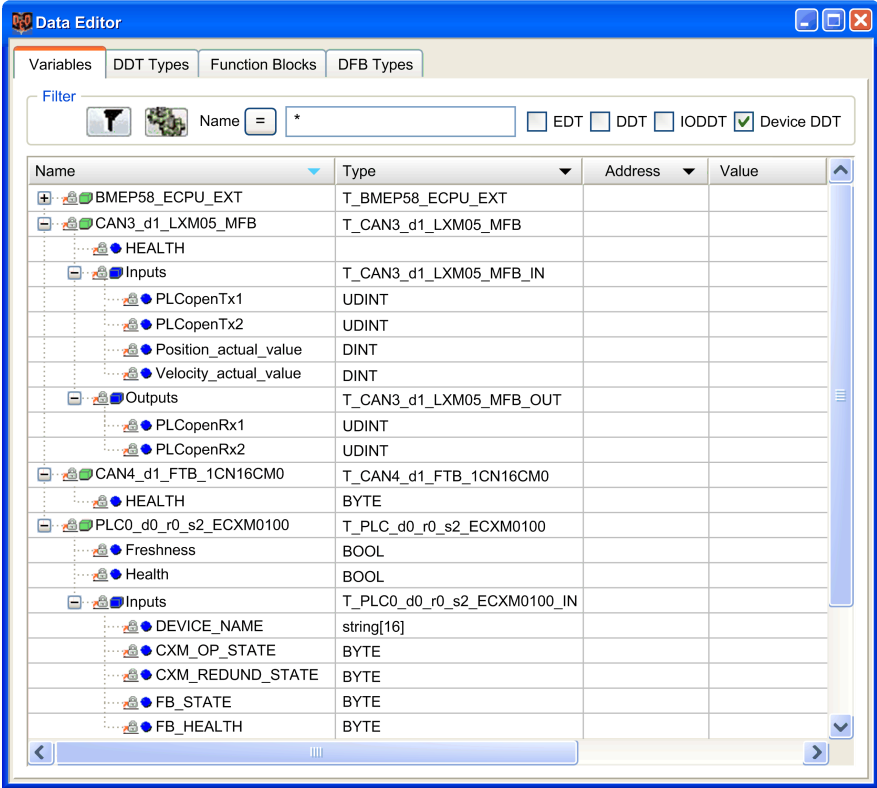
The **Data Editor** displays these variables, page 124.

## Accessing Device DDT

You can access the device DDTs and the corresponding variables in Control Expert. You can add this variable to a user-defined Animation Table (see EcoStruxure™ Control Expert, Operating Modes) to monitor read-only variables and edit read/write variables.

To access the device DDTs, follow these steps:

Step	Action
1	Click <b>Tools &gt; Project Browser</b> to open the Control Expert <b>Project Browser</b> .
2	Expand (+) <b>Variables &amp; FB instances</b> .

Step	Action																																																																																												
3	<p>Double-click <b>Device DDT Variables</b>.</p> <p><b>Result:</b> The <b>Data Editor</b> window is displayed:</p>  <p>The screenshot shows the 'Data Editor' window with the 'Device DDT' checkbox selected. The table below represents the data shown in the screenshot:</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Address</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>BMEP58_ECPU_EXT</td> <td>T_BMEP58_ECPU_EXT</td> <td></td> <td></td> </tr> <tr> <td>CAN3_d1_LXM05_MFB</td> <td>T_CAN3_d1_LXM05_MFB</td> <td></td> <td></td> </tr> <tr> <td>HEALTH</td> <td></td> <td></td> <td></td> </tr> <tr> <td>Inputs</td> <td>T_CAN3_d1_LXM05_MFB_IN</td> <td></td> <td></td> </tr> <tr> <td>  PLCopenTx1</td> <td>UDINT</td> <td></td> <td></td> </tr> <tr> <td>  PLCopenTx2</td> <td>UDINT</td> <td></td> <td></td> </tr> <tr> <td>  Position_actual_value</td> <td>DINT</td> <td></td> <td></td> </tr> <tr> <td>  Velocity_actual_value</td> <td>DINT</td> <td></td> <td></td> </tr> <tr> <td>Outputs</td> <td>T_CAN3_d1_LXM05_MFB_OUT</td> <td></td> <td></td> </tr> <tr> <td>  PLCopenRx1</td> <td>UDINT</td> <td></td> <td></td> </tr> <tr> <td>  PLCopenRx2</td> <td>UDINT</td> <td></td> <td></td> </tr> <tr> <td>CAN4_d1_FTB_1CN16CM0</td> <td>T_CAN4_d1_FTB_1CN16CM0</td> <td></td> <td></td> </tr> <tr> <td>HEALTH</td> <td>BYTE</td> <td></td> <td></td> </tr> <tr> <td>PLC0_d0_r0_s2_ECXM0100</td> <td>T_PLC_d0_r0_s2_ECXM0100</td> <td></td> <td></td> </tr> <tr> <td>  Freshness</td> <td>BOOL</td> <td></td> <td></td> </tr> <tr> <td>  Health</td> <td>BOOL</td> <td></td> <td></td> </tr> <tr> <td>Inputs</td> <td>T_PLC0_d0_r0_s2_ECXM0100_IN</td> <td></td> <td></td> </tr> <tr> <td>  DEVICE_NAME</td> <td>string[16]</td> <td></td> <td></td> </tr> <tr> <td>  CXM_OP_STATE</td> <td>BYTE</td> <td></td> <td></td> </tr> <tr> <td>  CXM_REDUND_STATE</td> <td>BYTE</td> <td></td> <td></td> </tr> <tr> <td>  FB_STATE</td> <td>BYTE</td> <td></td> <td></td> </tr> <tr> <td>  FB_HEALTH</td> <td>BYTE</td> <td></td> <td></td> </tr> </tbody> </table>	Name	Type	Address	Value	BMEP58_ECPU_EXT	T_BMEP58_ECPU_EXT			CAN3_d1_LXM05_MFB	T_CAN3_d1_LXM05_MFB			HEALTH				Inputs	T_CAN3_d1_LXM05_MFB_IN			PLCopenTx1	UDINT			PLCopenTx2	UDINT			Position_actual_value	DINT			Velocity_actual_value	DINT			Outputs	T_CAN3_d1_LXM05_MFB_OUT			PLCopenRx1	UDINT			PLCopenRx2	UDINT			CAN4_d1_FTB_1CN16CM0	T_CAN4_d1_FTB_1CN16CM0			HEALTH	BYTE			PLC0_d0_r0_s2_ECXM0100	T_PLC_d0_r0_s2_ECXM0100			Freshness	BOOL			Health	BOOL			Inputs	T_PLC0_d0_r0_s2_ECXM0100_IN			DEVICE_NAME	string[16]			CXM_OP_STATE	BYTE			CXM_REDUND_STATE	BYTE			FB_STATE	BYTE			FB_HEALTH	BYTE		
Name	Type	Address	Value																																																																																										
BMEP58_ECPU_EXT	T_BMEP58_ECPU_EXT																																																																																												
CAN3_d1_LXM05_MFB	T_CAN3_d1_LXM05_MFB																																																																																												
HEALTH																																																																																													
Inputs	T_CAN3_d1_LXM05_MFB_IN																																																																																												
PLCopenTx1	UDINT																																																																																												
PLCopenTx2	UDINT																																																																																												
Position_actual_value	DINT																																																																																												
Velocity_actual_value	DINT																																																																																												
Outputs	T_CAN3_d1_LXM05_MFB_OUT																																																																																												
PLCopenRx1	UDINT																																																																																												
PLCopenRx2	UDINT																																																																																												
CAN4_d1_FTB_1CN16CM0	T_CAN4_d1_FTB_1CN16CM0																																																																																												
HEALTH	BYTE																																																																																												
PLC0_d0_r0_s2_ECXM0100	T_PLC_d0_r0_s2_ECXM0100																																																																																												
Freshness	BOOL																																																																																												
Health	BOOL																																																																																												
Inputs	T_PLC0_d0_r0_s2_ECXM0100_IN																																																																																												
DEVICE_NAME	string[16]																																																																																												
CXM_OP_STATE	BYTE																																																																																												
CXM_REDUND_STATE	BYTE																																																																																												
FB_STATE	BYTE																																																																																												
FB_HEALTH	BYTE																																																																																												
4	<p>In the <b>Variables</b> tab, expand (+) the name to display inputs, outputs, and other parameters.</p>																																																																																												

**NOTE:** The red arrow and lock icons in the **Device DDT** table indicate that the variable name was auto-generated by Control Expert based on the configuration of the BMECXM module and CANopen slave. The variable name cannot be edited.

# Programming

## What's in This Chapter

Network Management Services .....	104
Exchanges Using SDOs .....	108
READ_SDO: Reading Service Data Object .....	109
WRITE_SDO: Writing Service Data Object .....	112
Function Block Examples .....	116

## Introduction

This chapter describes the programming of a CANopen architecture.

# Network Management Services

## Overview

The network management (NMT) is used to start, stop, reset and initialize CANopen nodes. It process boot-up messages and error control events of the individual CANopen slave devices.

The error control mechanism monitors CANopen slave devices by **Heartbeat** or **Node Guarding**.

The behavior of the CANopen slave devices depends on the configuration of the following objects:

- *NMT startup* object (1F80 hex)  
This object specify the start-up state (*OPERATIONAL*, *PRE-OPERATIONAL*, or *STOPPED*) for every CANopen slave devices on the network.
- *Slave assignment* object (1F81 hex)

The start-up states of the CANopen slave devices can be modified once the BMECXMCM is in *CONNECTED RUN* state by sending NMT commands using explicit messages.

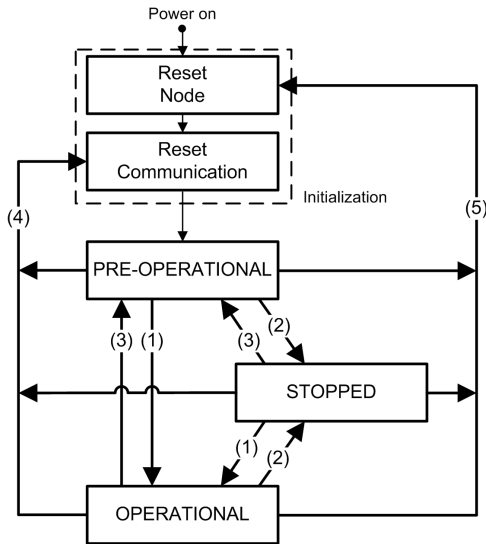
The NMT commands are accessible through a *WRITE\_SDO* or *READ\_SDO* in object 1F82 hex, page 151.

**NOTE:** NMT commands are accepted in both automatic and manual modes.



## NMT State Machine

The following figure gives the NMT service commands to controlling the operating behavior of a CANopen node on the network:



- (1) Start remote node
- (2) Stop remote node
- (3) Enter pre-operational
- (4) Reset communication
- (5) Reset node

## NMT Commands Values

The following table gives the NMT commands available through CANopen SDO commands, page 156:

Value (hex)	READ_SDO command	WRITE_SDO command
00	NMT state unknown	Reserved
01	CANopen device missing	Reserved
02	Reserved	

Value (hex)	READ_SDO command	WRITE_SDO command
03	Reserved	
04	NMT state <i>STOPPED</i>	NMT service <i>Stop remote node</i>
05	NMT state <i>OPERATIONAL</i>	NMT service <i>Start remote node</i>
06	Reserved	
07	Reserved	
08	Reserved	
–	–	
7E	Reserved	
7F	NMT state <i>PRE-OPERATIONAL</i>	NMT service <i>Enter pre-operational</i>
80	Reserved	
–	–	
83	Reserved	
84	NMT state <i>STOPPED</i>	NMT service <i>Stop remote node</i> (excluding NMT master and requesting CANopen device)
85	NMT state <i>OPERATIONAL</i>	NMT service <i>Start remote node</i> (excluding NMT master and requesting CANopen device)
86	Reserved	NMT service <i>Reset node</i> (excluding NMT master and requesting CANopen device)
87	Reserved	NMT service <i>Reset communication</i> (excluding NMT master and requesting CANopen device)
88	Reserved	
–	–	
8E	Reserved	
8F	NMT state <i>PRE-OPERATIONAL</i>	NMT service <i>Enter pre-operational</i> (excluding NMT master and requesting CANopen device)
90	Reserved	
–	–	
FF	Reserved	

## NMT Command Data Examples:

This table shows the command data for a `Start remote node`:

Value	Size	Parameter
0: PLC 1...5: DTM	SINT	Connect ID
127	SINT	Node ID
1F82 hex	INT	Index
Node ID: 1...7F hex: Node ID targeted 80 hex: all nodes (broadcast)	SINT	Subindex
[1]	INT	Length
05 hex 85 hex (excluding master)	SINT	Data

This table shows the command data for a `Stop remote node`:

Value	Size	Parameter
0: PLC 1...5: DTM	SINT	Connect ID
127	SINT	Node ID
1F82 hex	INT	Index
Node ID: 1...7F hex: Node ID targeted 80 hex: all nodes (broadcast)	SINT	Subindex
[1]	INT	Length
04 hex 84 hex (excluding master)	SINT	Data

# Exchanges Using SDOs

## Overview

SDO commands are used to access (read/write) the parameters of CANopen entries and devices in the object dictionary.

For SDO objects sent by the PLC application, the explicit exchanges of messages on a CANopen bus is done by read/write functions.

<b>⚠ WARNING</b>
<p><b>UNINTENDED EQUIPMENT OPERATION</b></p> <p>When modifying a variable, check the consequences of the SDO command in the documentation of the specific target CANopen device.</p> <p><b>Failure to follow these instructions can result in death, serious injury, or equipment damage.</b></p>

## SDO Access

It is possible to access SDOs using the *READ\_SDO* and *WRITE\_SDO* function blocks.

For examples of SDO function blocks, refer to function block example, page 116.

## SDO Timeouts

The following SDO timeouts are configurable in the CANopen bus parameter window, page 82:

Global SDO timeout:	<p>By default the value is set to 50 ms and defined in the object 5FF0 hex.</p> <p>This is the time for the BMECXM module to read object 1000 hex of each CANopen slave devices of the CANopen fieldbus at bootup.</p>
Slave specific SDO timeout:	<p>By default the value is set to 9500 ms for all CANopen slave devices and defined in the object 5FF1 hex.</p> <p>This is the time for the BMECXM module to read objects 1010 hex, 1011 hex, and 1F50 hex of the CANopen slave device at bootup.</p> <p><b>NOTE:</b> This slave specific SDO timeout is necessary for devices with long response time.</p>

In addition to these configurable SDO timeouts, a *READ\_SDO* has a timeout of 1 s and a *WRITE\_SDO* has a timeout of 2 s.

## READ\_SDO: Reading Service Data Object

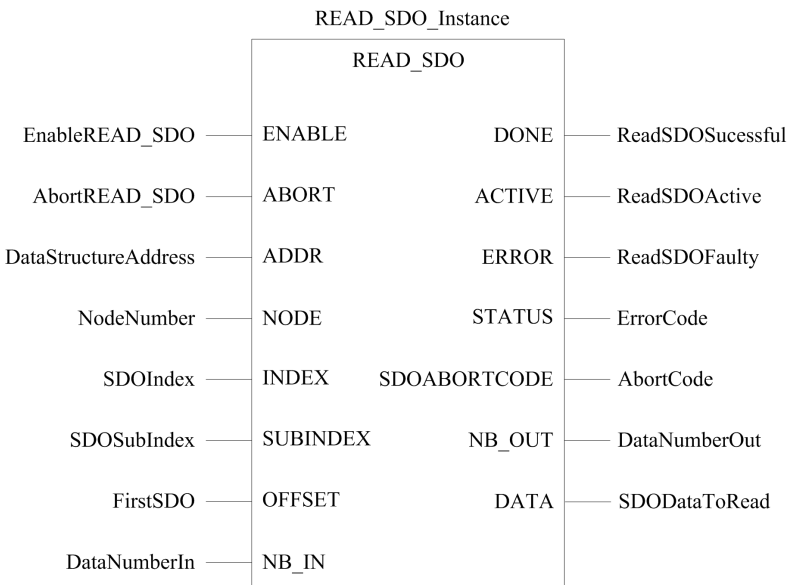
### Function Description

The *READ\_SDO* function block reads (explicit exchange) from the PLC application up to the device (SDO).

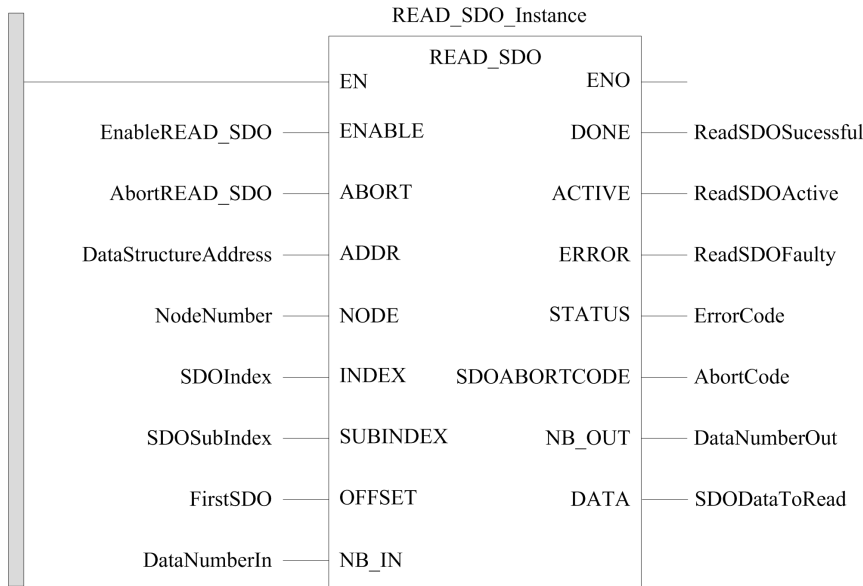
This function block provides access to the abort code when the SDO command is not successful (only if the fieldbus is in RUN mode, and only towards the configured devices).

### FBD Representation

Representation:



## LD Representation



## IL Representation

Representation:

```

CAL READ_SDO_Instance (ENABLE := EnableREAD_SDO, ABORT := AbortREAD_SDO, ADDR := DataSetAddress, NODE := NodeNumber, INDEX := SDOIndex, SUBINDEX := SDOSubIndex, OFFSET := FirstSDO, NB_IN := DataNumberIn, DONE => ReadSDOSuccessful, ACTIVE => ReadSDOActive, ERROR => ReadSDOFaulty, STATUS => ErrorCode, SDOABORTCODE => AbortCode, NB_OUT => DataNumberOut, DATA => SDODataToRead)
    
```

## ST Representation

Representation:

```

READ_SDO_Instance (ENABLE := EnableREAD_SDO, ABORT := AbortREAD_SDO, ADDR := DataSetAddress, NODE := NodeNumber, INDEX := SDOIndex, SUBINDEX := SDOSubIndex, OFFSET := FirstSDO, NB_IN := DataNumberIn, DONE => ReadSDOSuccessful, ACTIVE => ReadSDOActive, ERROR => ReadSDOFaulty,
    
```

STATUS => ErrorCode, SDOABORTCODE => AbortCode, NB\_OUT => DataNumberOut,  
DATA => SDODataToRead)

## Parameter Description

The following table describes the input parameters:

Input parameter	Data type	Description
<i>ENABLE</i>	BOOL	ON: the operation is enabled.
<i>ABORT</i>	BOOL	ON: the currently active operation is aborted.
<i>ADDR</i>	ANY_ARRAY_INT	Array containing the address of the destination entity of the read operation, result of ADDMX function.
<i>NODE</i>	BYTE	Byte used to select a particular NMT slave device on the CANopen network (16#01 to 16#7F).
<i>INDEX</i>	INT	Two bytes used to access a particular object in a CANopen SDO server device.
<i>SUBINDEX</i>	BYTE	Byte used to access a particular subobject in a CANopen SDO server device.
<i>OFFSET</i>	INT	Two bytes indicating the starting offset into the selected object. It can be non-zero when performing segmented SDO transfers. <b>NOTE:</b> Not used when addressing an EtherNet/IP module (address with CIP suffix).
<i>NB_IN</i>	INT	Two bytes providing a count of the desired number of data values to read (in bytes). <b>NOTE:</b> <ul style="list-style-type: none"> <li>If set to 0, the number of data to read is set to the size of the variable associated to the output parameter <i>DATA</i>.</li> <li>When used with the BMECXM0100 module, this input parameter equal 0 whatever the value you set.</li> </ul>

The following table describes the output parameters:

Output parameter	Data type	Description
<i>DONE</i>	BOOL	ON: the operation concludes successfully.
<i>ACTIVE</i>	BOOL	ON: the operation is active.
<i>ERROR</i>	BOOL	ON: the operation is aborted without success.

Output parameter	Data type	Description
<i>STATUS</i>	WORD	Provides the error code (see EcoStruxure™ Control Expert, Communication, Block Library) if an error is detected by the function block.
<i>SDOABORT-CODE</i>	DWORD	SDO abort code, page 159 when <i>STATUS</i> = 16#4007
<i>NB_OUT</i>	INT	Size of data (in BYTES) actually returned in the output parameter <i>DATA</i> .
<i>DATA</i>	ANY_ARRAY_BYTE	Read data.

## WRITE\_SDO: Writing Service Data Object

### Function Description

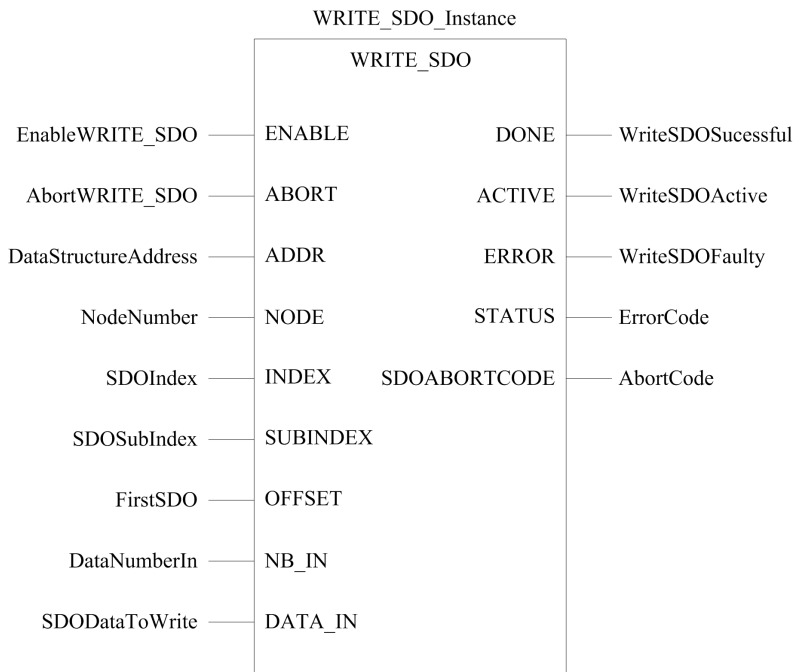
The `WRITE_SDO` function block writes (explicit exchanges) from the PLC application up to the device (SDO).

This function block provides access to the abort code when the SDO command is not successful (only if the fieldbus is in RUN mode, and only towards the configured devices).

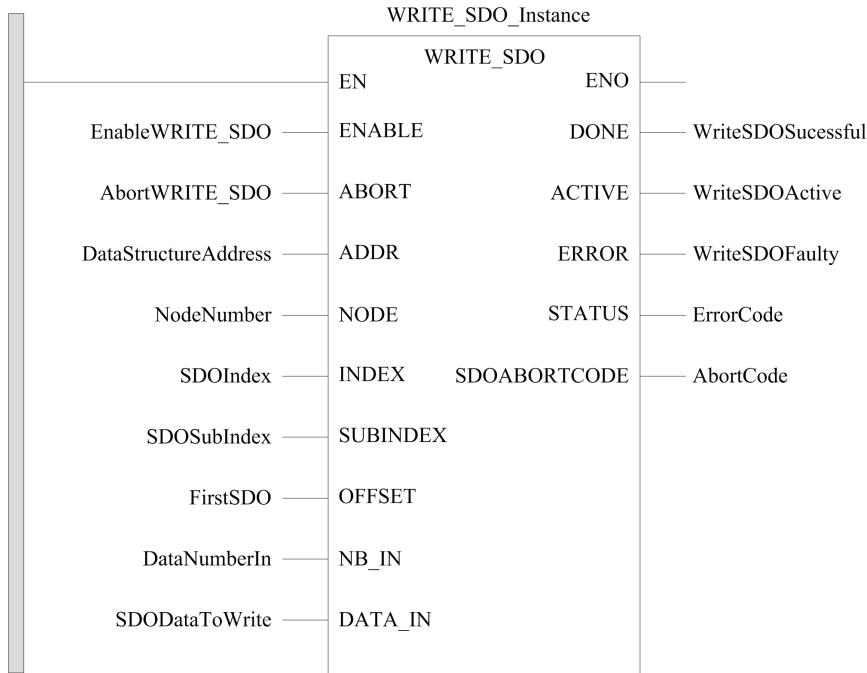


# FBD Representation

Representation:



## LD Representation



## IL Representation

Representation:

```

CAL WRITE_SDO_Instance (ENABLE := EnableWRITE_SDO, ABORT := AbortWRITE_SDO, ADDR := DataSetAddress, NODE := NodeNumber, INDEX := SDOIndex, SUBINDEX := SDOSubIndex, OFFSET := FirstSDO, NB_IN := DataNumberIn, DATA_IN := SDODataToWrite, DONE => WriteSDOSuccessful, ACTIVE => WriteSDOActive, ERROR => WriteSDOFaulty, STATUS => ErrorCode, SDOABORTCODE => AbortCode)
    
```

## ST Representation

Representation:

```

WRITE_SDO_Instance (ENABLE := EnableWRITE_SDO, ABORT := AbortWRITE_SDO, ADDR := DataSetAddress, NODE := NodeNumber, INDEX := SDOIndex,
    
```

```

SUBINDEX := SDOSubIndex, OFFSET := FirstSDO, NB_IN := DataNumberIn,
DATA_IN := SDODataToWrite, DONE => WriteSDOSuccessful, ACTIVE =>
WriteSDOActive, ERROR => WriteSDOFaulty, STATUS => ErrorCode,
SDOABORTCODE => AbortCode)

```

## Parameter Description

The following table describes the input parameters:

Input parameter	Data type	Description
<i>ENABLE</i>	BOOL	ON: the operation is enabled.
<i>ABORT</i>	BOOL	ON: the currently active operation is aborted.
<i>ADDR</i>	ANY_ARRAY_INT	Array containing the address of the destination entity of the read operation, result of ADDMX function.
<i>NODE</i>	BYTE	Byte used to select a particular NMT slave device on the CANopen network (16#01 to 16#7F).
<i>INDEX</i>	INT	Two bytes used to access a particular object in a CANopen SDO server device.
<i>SUBINDEX</i>	BYTE	Byte used to access a particular subobject in a CANopen SDO server device.
<i>OFFSET</i>	INT	Two bytes indicating the starting offset into the selected object. It can be non-zero when performing segmented SDO transfers. <b>NOTE:</b> Not used when addressing an EtherNet/IP module (address with CIP suffix).
<i>NB_IN</i>	INT	Two bytes providing a count of the desired number of data values to be written (in bytes).
<i>DATA_IN</i>	ANY_ARRAY_BYTE	Data to write.

The following table describes the output parameters:

Output parameter	Data type	Description
<i>DONE</i>	BOOL	ON: the operation concludes successfully.
<i>ACTIVE</i>	BOOL	ON: the operation is active.
<i>ERROR</i>	BOOL	ON: the operation is aborted without success.

Output parameter	Data type	Description
<i>STATUS</i>	WORD	Provides the error code (see EcoStruxure™ Control Expert, Communication, Block Library) if an error is detected by the function block.
<i>SDOABORT-CODE</i>	DWORD	SDO abort code, page 159 when <i>STATUS</i> = 16#4007.

## Function Block Examples

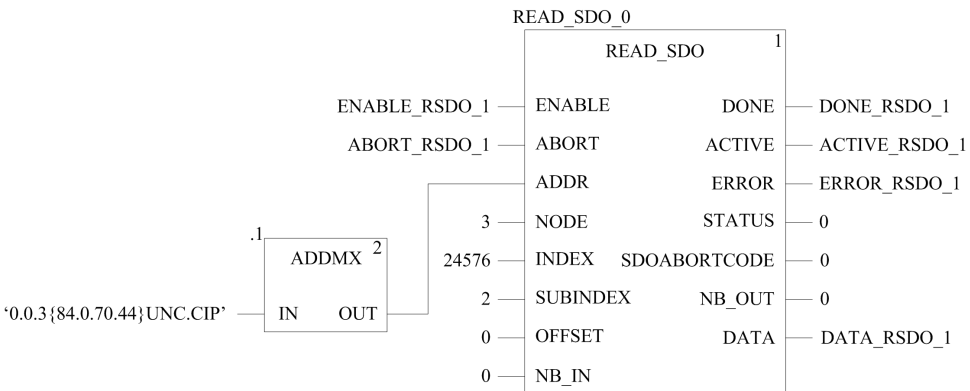
### Overview

In these examples:

- The IP address of the BMECXM module is 84.0.70.44
- The CANopen slave device is an FTB\_1CN16CM0 located on the bus at **Node-ID 3**.

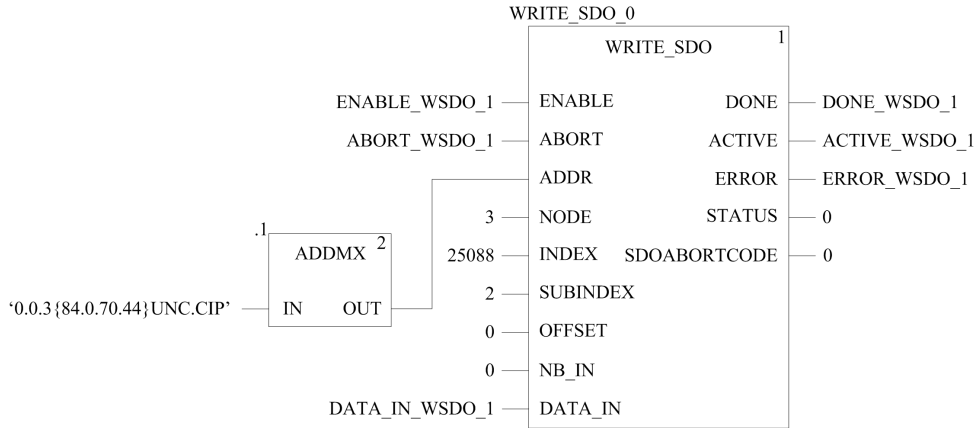
### READ\_SDO Example in FBD

This example is executing a *READ\_SDO* of the object defined at Index 6000 hex and SubIndex 02 hex (Digital Input 8 bits Pin 2):



## WRITE\_SDO Example in FBD

This example is executing a *WRITE\_SDO* of the object defined at Index 6200 hex and SubIndex 02 hex (Write Outputs 9 to 16):



# Diagnostics

## What's in This Chapter

LED Diagnostic.....	119
Device DDT for BMECXM Modules .....	124
Device DDT for CANopen Slave Devices.....	127
BMECXM DTM Diagnosis .....	128
Sending Explicit Messages to the BMECXM Module.....	132
Embedded Web Pages.....	134
Emergency Objects.....	140

## Introduction

This chapter describes the diagnostic means for the BMECXM modules:

- LED indicators on the BMECXM module.
- Control Expert device DDTs:
  - With the device DDT of the M580 CPU, you can perform a first diagnostic.  
If the BMECXM is managed by the RIO scanner, the *RIO\_HEALTH* is activated.  
If the BMECXM is managed by the DIO scanner, the *DIO\_HEALTH* is activated.  
**NOTE:** For more details on M580 CPU device DDT, refer to chapter *Standalone DDT Data Structure for M580 CPUs* (see Modicon M580, Hardware, Reference Manual).
  - The BMECXM device DDT.
  - The CANopen slave device DDT.
- BMECXM DTM diagnosis
- Explicit messages for extended diagnostics:
  - via programming, page 104.
  - via the M580 CPU DTM graphic user interface, using the *DATA\_EXCH*, *READ\_SDO* and CIP objects.
- Embedded web pages with emergency messages.

**NOTE:** The most detailed diagnostic information can be found in the web pages.

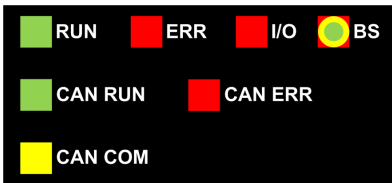
# LED Diagnostic

## Overview

LED indicators report the behavior of the module and its communications with the network. LED indicators appear as words or abbreviations at the top of the module.

## LED Display

This is the LED display located on the front panel of the BMECXM module:





Each LED of the BMECXM module is located with a corresponding positioning letter used in the input parameter `CXM_DISPLAY` of the module device DDT, page 124:

A = RUN	B = ERR	C = I/O	D = BS
E = CAN RUN	–	G = CAN ERR	–
F = CAN COM	–	–	–

## LED State and Flash Rates

This table describes the LED state used in the following tables for diagnosing the module:

LED State	Flash Rate	State Symbol
LED off	Constantly OFF	
LED on	Constantly ON	
LED blinking	Iso-phase: <ul style="list-style-type: none"> <li>• 200 ms ON</li> <li>• 200 ms OFF</li> </ul>	

LED State	Flash Rate	State Symbol
LED flashing	One short single flash: <ul style="list-style-type: none"> <li>• 200 ms ON</li> <li>• 1200 ms OFF</li> </ul>	
One of the possible patterns	–	

## LED Description

This table describes the **RUN**, **ERR**, **I/O**, and **BS** LED states and colors of the BMECXM module:

LED	Color	State	Description
<b>RUN</b>	Green	On	The module is in <i>RUN</i> state.
		Off	<ul style="list-style-type: none"> <li>• There is no power to the module, or</li> <li>• Module configuration failed (see <b>ERR</b> LED for identifying the detected error).</li> </ul>
		Blinking	<ul style="list-style-type: none"> <li>• The power on self-test is running, or</li> <li>• A firmware update is in progress (see <b>BS</b> LED to confirm).</li> </ul>
<b>ERR</b>	Red	On	<ul style="list-style-type: none"> <li>• Critical error detected while running the power-on self-test (<i>INITIALIZATION</i> phase failed), or</li> <li>• Error detected while getting IP address via DHCP (duplicate address).</li> </ul>
		Off	No errors detected.
		Blinking	<ul style="list-style-type: none"> <li>• The power on self-test is running, or</li> <li>• An error detected while retrieving the FDR file.</li> </ul>
<b>I/O</b>	Red	Off	Signification is depending on the module status: <ul style="list-style-type: none"> <li>• If the module is in not in <i>RUN</i> state:                             <ul style="list-style-type: none"> <li>◦ A firmware update is in progress (see <b>BS</b> LED to confirm).</li> <li>◦ Module configuration failed (see <b>ERR</b> LED for identifying the detected error).</li> </ul> </li> <li>• If the module is in <i>RUN</i> state, the combination of <b>I/O</b> and <b>CAN ERR</b> LEDs provides a <i>CANopen</i> diagnostic, page 123.</li> </ul>
		On	The <i>CANopen</i> fieldbus state is <i>NO-CONF</i> or <i>BUS OFF</i> .
		Blinking	Signification is depending on the module status: <ul style="list-style-type: none"> <li>• If the module is in not in <i>RUN</i> state, the power-on self-test is running.</li> <li>• If the module is in <i>RUN</i> state, the combination of <b>I/O</b> and <b>CAN ERR</b> LEDs provides a <i>CANopen</i> diagnostic, page 123.</li> </ul>











































LED	Color	State	Description
BS (bus status)	–	Off	<ul style="list-style-type: none"> <li>Module is not configured, or</li> <li>Module is waiting for IP address from DHCP.</li> </ul>
	Green	On	All EtherNet/IP connections are established.
		Flashing	The module has an IP address, but there is no EtherNet/IP connection.
		Blinking	The power on self-test is in progress.
	Red	On	There is a duplicate IP address.
		Flashing	At least one EtherNet/IP connection is lost. The LED flashes until the connection is re-established or the module is reset.
	Yellow	Blinking	The loading of the firmware is in progress.

This table describes the **CAN RUN**, **CAN ERR**, and **CAN COM** LED states and colors of the CANopen fieldbus:

LED	Color	State	Description
CAN RUN	Green	Off	There is no power to the module.
		On	The CANopen fieldbus state is <i>OPERATIONAL</i> .
		Blinking	The CANopen fieldbus state is <i>PRE-OPERATIONAL</i> .
		Flashing	The CANopen fieldbus state is <i>STOPPED</i> .
CAN ERR	Red	On	<ul style="list-style-type: none"> <li>No CANopen device is configured, or</li> <li>CANopen fieldbus state is <i>BUS OFF</i></li> </ul>
		Off	No CANopen error detected.
		Flashing	At least one of the detected error counters has reached or exceeded the warning level (too many error frames).
CAN COM	Yellow	Flashing	There is an SDO message.

## General Diagnostic

A general diagnostic of the module is possible when you observe the four upper LEDs (**RUN**, **ERR**, **I/O**, and **BS**) in combination:

LEDs				Condition
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Firmware download.
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Power is off. <i>CXM_OP_STATE=IDLE</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Power-on self-test is in progress. <i>CXM_OP_STATE=INITIALIZATION</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	The power on self-test failed. <i>CXM_OP_STATE=INITIALIZATION</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	The power on self-test is completed and the module proceed to: <ul style="list-style-type: none"> <li>• Backplane initialization,</li> <li>• Get IP address (from DHCP),</li> <li>• Get configuration file (from FDR server).</li> </ul> <i>CXM_OP_STATE=UNCONFIGURED</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Error detected while getting IP address via DHCP (duplicate address <sup>(2)</sup> ). <i>CXM_OP_STATE=UNCONFIGURED</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Valid IP address but no EtherNet/IP connection. <i>CXM_OP_STATE=CONFIGURED</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O<sup>(1)</sup></b>	 <b>BS</b>	EtherNet/IP connections established. <i>CXM_OP_STATE=CONNECTED STOP, CONNECTED RUN, or FALLBACK</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O<sup>(1)</sup></b>	 <b>BS</b>	EtherNet/IP connections are closed. <i>CXM_OP_STATE=CONNECTED STOP, CONNECTED RUN, or FALLBACK</i>
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O<sup>(1)</sup></b>	 <b>BS</b>	Detected communication failure. <i>CXM_OP_STATE=CONNECTED STOP, CONNECTED RUN, or FALLBACK</i>






















(1) The status of the **I/O** LED in combination with **CAN ERR** LED gives information on I/O exchange with CANopen devices, page 123.

(2) In case of duplicate IP address, the LED **BS** is blinking at the start-up phase then the module restarts.

## CANopen LED Diagnostic

The module in *RUN* state is a prerequisite to diagnose the I/O exchange with CANopen devices with the LEDs.

The following tables give diagnostic when you observe the **I/O** and **CAN ERR** LEDs in combination:

LEDs				Condition
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	No device configured, or CANopen bus is off (physical wire is bus off).
 <b>CAN RUN</b>		 <b>CAN ERR</b>		
 <b>CAN COM</b>				
LEDs				Condition
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	Configuration detected error, communication detected error on CANopen device or CANopen device absent on bus.
 <b>CAN RUN</b>		 <b>CAN ERR</b>		
 <b>CAN COM</b>				
LEDs				Condition
 <b>RUN</b>	 <b>ERR</b>	 <b>I/O</b>	 <b>BS</b>	No error detected.
 <b>CAN RUN</b>		 <b>CAN ERR</b>		
 <b>CAN COM</b>				

# Device DDT for BMECXM Modules

## Overview

Use the device derived data type (DDDT) for diagnosis. There is one device DDT for each BMECXM module.

The device DDT, contains:

- Input parameters
- Other parameters

**NOTE:** To access these parameters, refer to device DDT Variables, page 102.

## Input Parameters

Parameter	Type	Bit	Description
<b>DEVICE_NAME</b>	STRING [16]		Device name of the CXM module
<b>CXM_OP_STATE</b>	BYTE		Module state operating mode: <b>0</b> = <i>INITIALIZATION</i> <b>1</b> = <i>UNCONFIGURED</i> <b>2</b> = <i>CONFIGURED</i> <b>3</b> = <i>CONNECTED STOP</i> <b>4</b> = <i>CONNECTED RUN</i> <b>5</b> = <i>FALLBACK</i>
<b>CXM_REDUND_STATE</b>	BYTE		Reserved
<b>FB_STATE</b>	BYTE		Fieldbus operating mode: <b>0</b> = <i>IDLE</i> <b>1</b> = <i>NO-CONF</i> <b>2</b> = <i>BUS OFF</i> <b>3</b> = <i>STOPPED</i> <b>4</b> = <i>PRE-OPERATIONAL</i> <b>5</b> = <i>OPERATIONAL</i> <b>6</b> = <i>CLEAR</i>

Parameter	Type	Bit	Description
<b>FB_HEALTH</b>	BYTE		Status information of the network manager for diagnosing fieldbus:  <b>0</b> = Idle <b>1</b> = Fieldbus error detected <b>2</b> = Device fault detected <b>3</b> = Device error detected <b>4</b> = Device fault detected and error detected
<b>SLAVE_PROG_LIST</b>	ARRAY [0...15] OF BYTE		Slave list for programmed state. 1 bit per slave device.
<b>SLAVE_LIVE_LIST</b>	ARRAY [0...15] OF BYTE		Slave list for responding state. 1 bit per slave device.
<b>SLAVE_DIAG_LIST</b>	ARRAY [0...15] OF BYTE		Slave list for error state. 1 bit per slave device.
<b>SLAVE_WAIT_LIST</b>	ARRAY [0...15] OF BYTE		Slave list that indicates if the device is waiting for explicit order to operate or not. 1 bit per slave device.
<b>FB_MAX_SCAN</b>	UDINT		Maximal time period of field devices scanning (by 100 µs resolution).
<b>FB_LAST_SCAN</b>	UDINT		Last time period of field devices scanning (by 100 µs resolution).
<b>FB_MIN_SCAN</b>	UDINT		Minimal time period of field devices scanning (by 100 µs resolution).
<b>CXM_DISPLAY</b>	UINT		8 LEDs A, B, C, D, E, F, G, and H located in 2 bits (high, low):  A = <b>RUN</b> : bits (1, 0) B = <b>ERR</b> : bits (3, 2) C = <b>I/O</b> : bits (5, 4) D = <b>BS</b> : bits (7, 6) E = <b>CAN RUN</b> : bits (9, 8) F = <b>CAN COM</b> : bits (11, 10) G = <b>CAN ERR</b> : bits (13, 12)  H = <b>Not used</b> : bits (15, 14) <b>Off</b> bit high = 0 and bit low = 0  <b>Green</b> bit high = 0 and bit low = 1 <b>Red</b> bit high = 1 and bit low = 0  <b>Yellow</b> bit high = 1 and bit low = 1

Parameter	Type	Bit	Description
<b>ETH_STATUS</b>	BYTE		Status information for Ethernet:
		0	<b>PORT1_LINK:</b> <ul style="list-style-type: none"> <li>0: Link down for Ethernet port 1</li> <li>1: Link up for Ethernet port 1</li> </ul>
		4	<b>RPI_CHANGE:</b> <ul style="list-style-type: none"> <li>0: EtherNet/IP RPI not in progress</li> <li>1: EtherNet/IP RPI in progress</li> </ul>
		5	<b>REDUNDANCY_STATUS:</b> <ul style="list-style-type: none"> <li>0: Backup path not available</li> <li>1: Backup path available</li> </ul>
		6	<b>REDUNDANCY_OWNER:</b> <ul style="list-style-type: none"> <li>0: Redundant owner not present</li> <li>1: Redundant owner present</li> </ul>
	7	<b>GLOBAL_STATUS:</b> <ul style="list-style-type: none"> <li>0: One or more services not operating normally</li> <li>1: All services operating normally</li> </ul>	
<b>SERVICE_STATUS</b>	BYTE		Status information of Ethernet services:
		1	<b>SNTP_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>
		3	<b>SNMP_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>
		4	<b>FDR_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Unable to download PRM file</li> <li>1: Service operating normally or disabled</li> </ul>
		5	<b>FIRMWARE_UPGRADE</b> <ul style="list-style-type: none"> <li>0: Firmware upgrade unauthorized</li> <li>1: Service operating normally</li> </ul>
		6	<b>WEB_PAGE</b> <ul style="list-style-type: none"> <li>0: Webpage not available</li> <li>1: Service operating normally or disabled</li> </ul>
	7	<b>EVENT_LOG_STATUS</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>	

Parameter	Type	Bit	Description
ETH_PORT_1_AND_2_STATUS	BYTE		Not applicable.
ETH_PORT_3_STATUS	BYTE		Not applicable.
SYSLOG_STATUS	BYTE		0: Set to 1 if the syslog client does not receive the acknowledgment of the TCP messages from the syslog server.

## Other Parameters

Parameter	Type	Description
Freshness	BOOL	Global freshness
Health	BOOL	Global I/O health reports a default in the update of the device DDT due to connection issues: <ul style="list-style-type: none"> <li>0: when a detected fault is reported by FB_HEALTH in block 1 or a wrong I/O signature is detected</li> <li>1: OK</li> </ul>

## Device DDT for CANopen Slave Devices

### Overview

Use the device derived data type (DDDT) for diagnosis.

There is one device DDT for each CANopen slave device.

The device DDT, contains:

- *HEALTH* parameter
- *Inputs* parameter
- *Outputs* parameter

**NOTE:** To access these parameters, refer to device DDT Variables, page 102.

### HEALTH Parameter

The *HEALTH* byte provides the status of the CANopen slave device:

Bit 3 Wait	Bit 2 Prog	Bit 1 Live	Bit 0 Diag	General Device Status
0	0	0	0	CANopen node not used
0	1	1	0	<i>OPERATIONAL</i>
1	1	1	0	<i>PRE-OPERATIONAL</i>
0	1	1	1	<i>ERR</i> (Configured with fault)
0	1	0	1	<i>FAULT</i> (Inoperative)
0	1	0	0	<i>DISABLE</i> (Configured)
1	1	0	0	<i>STOPPED</i>

## Inputs and *Outputs* Parameters

The parameter *Inputs* contains the variables mapped in active PDOs to transmit to the CANopen master.

The parameter *Outputs* contains the variable mapped in active PDOs to receive from the CANopen master.

**NOTE:** For detailed information on PDOs, refer to chapter *Device Configuration*, page 60.

## BMECXM DTM Diagnosis

### Overview

The Control Expert DTMs provides communication and status informations that is collected at polling intervals. To diagnose the operation of the CANopen application, use the BMECXM DTM.

**NOTE:** Control Expert DTMs are online diagnosis. To go online, refer to DTM Connections, page 85.



## Prerequisites

### **⚠ CAUTION**

#### **MISINTERPRETATION OF DIAGNOSIS**

Make sure that you are connected to the right BMECXM CANopen master module before diagnosing a CANopen slave device.

**Failure to follow these instructions can result in injury or equipment damage.**

To be able to connect the DTM to the BMECXM module:

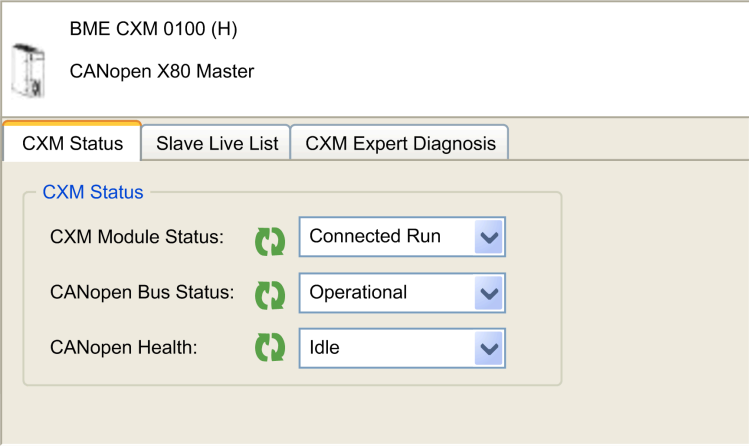
- Enable the access control of the PC that supports the DTM in the **Security** tab, page 96.
- Declare its IP address as **Source IP Address** in the **Channel Properties** of the Ethernet master DTM screen.

## Connect the DTM

Before you can open the diagnostics page, make the connection between the DTM for the target BMECXM module and the physical module:

Step	Action
1	Open the Control Expert <b>DTM Browser</b> ( <b>Tools &gt; DTM Browser</b> ).
2	Find the name that is assigned to the BMECXM module.
3	Right-click on the module name
4	Scroll to <b>Connect</b> .

## Accessing the DTM Diagnosis

Step	Action
1	Right-click the name that is assigned to your BMECXM module in the <b>DTM Browser</b> .
2	<p>Scroll to <b>Device Menu &gt; Diagnosis</b> to view the available diagnostics pages.</p> 

## CXM Status Tab

The **CXM Status** tab shows an overview of the current status.

This table describes the status information for the following parameters:

Parameter	Type	Description
<b>CXM Module Status</b>	BYTE	Indicates the status of the module: <ul style="list-style-type: none"> <li>• <b>Initialization</b></li> <li>• <b>Unconfigured</b></li> <li>• <b>Configured</b></li> <li>• <b>Connected Stop</b></li> <li>• <b>Connected Run</b></li> <li>• <b>Fallback</b></li> </ul>
<b>CANopen Bus Status</b>	BYTE	Indicates the status of the CANopen bus: <ul style="list-style-type: none"> <li>• <b>Idle</b></li> <li>• <b>No-Conf</b></li> <li>• <b>Bus Off</b></li> <li>• <b>Stopped</b></li> <li>• <b>Pre-Operational</b></li> <li>• <b>Operational</b></li> <li>• <b>Clear</b></li> </ul>
<b>CANopen Health</b>	BYTE	Indicates the status of the fieldbus: <ul style="list-style-type: none"> <li>• <b>Idle</b></li> <li>• <b>Fieldbus error detected</b></li> <li>• <b>Device fault detected</b></li> <li>• <b>Device error detected</b></li> <li>• <b>Device fault detected and error detected</b></li> </ul>

## Slave Live List Tab

The **Slave Live List** tab shows the slave diagnosis.

This table describes the color status of the LEDs as shown in the grid of the **Slave Live List** tab:

Color	General Device Status
White	CANopen node not used
Green	<i>OPERATIONAL</i>
Orange	<i>PRE-OPERATIONAL</i>
Red	<i>ERR</i> (Configured with fault)
Half Red/Orange	<i>FAULT</i> (Inoperative)

Color	General Device Status
Half Orange/White	<i>DISABLE</i> (Configured)
Yellow	<i>STOPPED</i>

## CXM Expert Diagnosis Tab

The **CXM Expert Diagnosis** tab displays parameters in hierarchical table, grouped by the following sections:

Group	Displays parameters available in ...
<b>Info</b>	DIAG_FXM_Diagnostic (object 301 hex), page 162
<b>Status</b>	
<b>EIP Parameters</b>	EIP Interface Diagnostic (object 350 hex), page 170
<b>IO Connections</b>	IO Connection Diagnostic (object 352 hex), page 173
<b>Explicit Messaging</b>	EtherNet/IP Explicit Connection Diagnostic (object 353 hex), page 175
<b>Fieldbus info</b>	DIAG_FXM_Diagnostic (object 301 hex), page 162

**NOTE:** When clicking the **Reset** button, all counter parameters are set to 0.

# Sending Explicit Messages to the BMECXM Module

## Overview

Use the EtherNet/IP Explicit Message window in the Control Expert DTM to send an explicit message from Control Expert to the BMECXM module on the network.

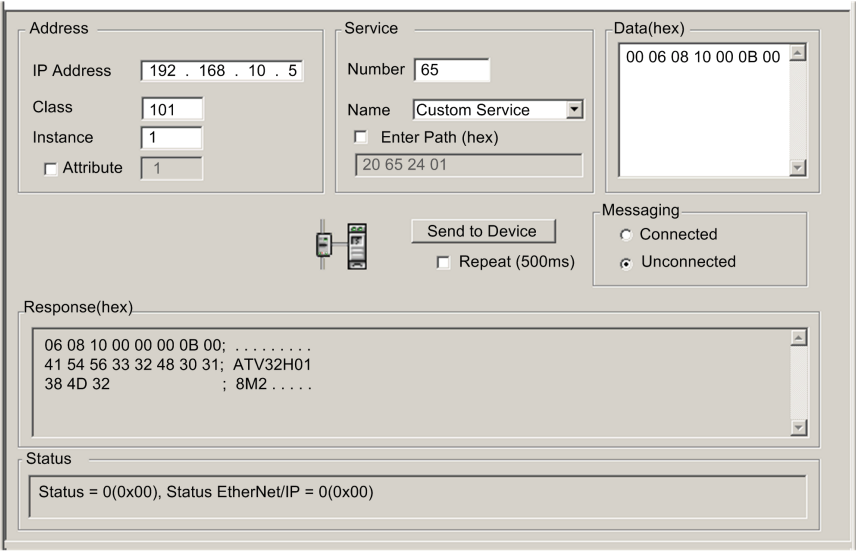
<b>⚠ CAUTION</b>
<p><b>MISINTERPRETATION OF DIAGNOSIS</b></p> <p>Make sure that you are connected to the right BMECXM CANopen master module before diagnosing a CANopen slave device.</p> <p><b>Failure to follow these instructions can result in injury or equipment damage.</b></p>

**NOTE:** For more details on how to configure EtherNet/IP explicit messages, refer to chapter *Explicit Messaging* (see Modicon M580, Hardware, Reference Manual).

## Explicit Message Example

On manual mode, you can read SDO from the M580 CPU master DTM screen.

To read the *Manufacturer Device Name* of a CANopen slave device follow these steps:

Step	Action
1	From the <b>DTM Browser</b> , right-click the DTM master.
2	Select <b>Device menu &gt; Additional functions &gt; EtherNet/IP Explicit Message</b> .
3	<p>In the <b>EtherNet/IP Explicit Message</b> configuration window, enter or select the following information in these fields:</p> <ul style="list-style-type: none"> <li><b>IP Address:</b> The IP address of the BMECXM module</li> <li><b>Class:</b> 101 (this is the decimal value of the object 65 hex for the <i>READ_SDO</i> command.)</li> <li><b>Instance:</b> 1</li> <li><b>Name:</b> Select <b>Custom Service</b> then enter to be able to write the Service Number.</li> <li><b>Number:</b> 65 (this is the decimal value of the service 41 hex for the <i>READ_SDO</i> command.)</li> </ul> 
4	<p>In the <b>Data(hex)</b> field, type the <i>READ_SDO</i> command</p> <p>For example, <i>00 06 08 10 00 0B 00</i>:</p>

Step	Action										
	<table border="1"> <tr> <td>00</td> <td>The request comes from the PLC</td> </tr> <tr> <td>06</td> <td>This is the node ID of the CANopen slave device (target of the request).</td> </tr> <tr> <td>08 10</td> <td>To read the index 1008 hex which corresponds to the Manufacturer Device Name object.</td> </tr> <tr> <td>00</td> <td>This is the subindex 00 hex of the object.</td> </tr> <tr> <td>0B 00</td> <td>This is the length of the data to read.</td> </tr> </table>	00	The request comes from the PLC	06	This is the node ID of the CANopen slave device (target of the request).	08 10	To read the index 1008 hex which corresponds to the Manufacturer Device Name object.	00	This is the subindex 00 hex of the object.	0B 00	This is the length of the data to read.
00	The request comes from the PLC										
06	This is the node ID of the CANopen slave device (target of the request).										
08 10	To read the index 1008 hex which corresponds to the Manufacturer Device Name object.										
00	This is the subindex 00 hex of the object.										
0B 00	This is the length of the data to read.										
5	<p>Click <b>Send to Device</b>.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>In the <b>Response(hex)</b> area, data is sent to the configuration tool by the target in the hexadecimal format.</li> <li>In the <b>Status</b> area, messages indicate whether the explicit message has succeeded or not.</li> </ul>										
6	Click <b>Close</b> .										

**NOTE:** For detailed information on available SDO commands, refer to appendix CANopen SDO commands, page 156.

## Embedded Web Pages

### Overview

The BMECXM modules support a set of web pages.

Embedded web pages provide tools to diagnose the basic functionality of the CANopen module through a web browser. These pages display real-time diagnostic data for both the BMECXM module and the CANopen slaves.

**NOTE:** You have to enable the access control of the PC that is web-connected in the **Security** tab of the BMECXM module, page 96.

### Access

An HTTPS server transmits web pages for monitoring and diagnosing the BMECXM module. The server provides easy access to the BMECXM module from standard Internet browsers.

To access the home page, follow these steps:

Step	Action
1	Open an Internet browser: <ul style="list-style-type: none"> <li>• Google chrome: version 11 or later</li> <li>• Mozilla Firefox: version 4 or later</li> <li>• Internet Explorer: version 8 or later</li> <li>• Safari: version 5.1.7 or later</li> </ul>
2	In the address bar, enter the IP address of the BMECXM module.
3	Press <b>Enter</b> .

You can also access web pages through the **Web : Main IP** tab from the CANopen master module screen, page 77.

**NOTE:** Web pages are automatically updated every 5 seconds.

## Menus

From the **Home** tab, you can access these menus:

Menu	Description
<b>CXM Info/Status</b>	Displays static and status information for the BMECXM module.
<b>EIP Interface</b>	Displays CIP-related diagnostics for the BMECXM module.
<b>IO Connections</b>	Displays diagnostics information on I/O connections between the scanner and the BMECXM module.
<b>CAN Diagnostics</b>	Displays the CAN diagnostics for the BMECXM module.
<b>CANopen Diagnostics</b>	Displays the CANopen diagnostics.
<b>Slave Details</b>	Displays the list and the status of programmed devices.

**NOTE:** When clicking the **Reset Counters** button, which is present on some menu pages, all counters are set to 0.

## CXM Info/Status Menu

Click **CXM Info/Status** to access this information:

Parameter	Description
<b>LED Displayed</b>	Contains LED indicators. The diagnostics information associated with the LED activity is described in LED Indicators, page 119. <b>NOTE:</b> The <b>CPU Rate Available</b> is the percentage of CPU time available.
<b>Name</b>	Gives the device name.
<b>Version</b>	Describes the software and hardware versions that are running on the CANopen module, and the configuration supported.
<b>State</b>	Gives information on: <ul style="list-style-type: none"> <li>• <b>CXMOpState:</b> Module state operating mode                             <ul style="list-style-type: none"> <li>◦ 0: <i>INITIALIZATION</i></li> <li>◦ 1: <i>UNCONFIGURED</i></li> <li>◦ 2: <i>CONFIGURED</i></li> <li>◦ 3: <i>CONNECTED STOP</i></li> <li>◦ 4: <i>CONNECTED RUN</i></li> <li>◦ 5: <i>FALLBACK</i></li> </ul> </li> <li>• <b>CXMFbState:</b> Status of the CANopen fieldbus                             <ul style="list-style-type: none"> <li>◦ 0: <i>IDLE</i></li> <li>◦ 1: <i>NO-CONF</i></li> <li>◦ 2: <i>BUS OFF</i></li> <li>◦ 3: <i>STOPPED</i></li> <li>◦ 4: <i>PRE-OPERATIONAL</i></li> <li>◦ 5: <i>OPERATIONAL</i></li> <li>◦ 6: <i>CLEAR</i></li> </ul> </li> <li>• <b>CXMRedundState:</b> Reserved</li> </ul>
<b>Ethernet</b>	Gives information on: <ul style="list-style-type: none"> <li>• <b>Ethernet Status</b> <ul style="list-style-type: none"> <li>◦ Bit 0: Linkup/down for Ethernet port 1</li> <li>◦ Bit 4: EtherNet/IP RPI in progress</li> <li>◦ Bit 5: Redundancy status/backup path available</li> <li>◦ Bit 6: Redundant owner available</li> <li>◦ Bit 7: Global service status</li> </ul> </li> <li>• <b>Ethernet Service</b> <ul style="list-style-type: none"> <li>◦ Bit 0: Reserved</li> <li>◦ Bit 1: SNTP</li> <li>◦ Bit 2: Reserved for port 502</li> <li>◦ Bit 3: FDR</li> <li>◦ Bit 4...7: Reserved</li> </ul> </li> </ul>
<b>IP Address</b>	Indicates the IP address, subnet mask, default gateway, and MAC address.



## EIP Interface Menu

Click **EIP Interface** to access this information:

Parameter	Description
<b>EIP Interface</b>	Displays the protocol supported and diagnostic information on: <ul style="list-style-type: none"> <li>• CIP and current CIP</li> <li>• CIP errors detected</li> <li>• Counter</li> <li>• Error detected counters</li> <li>• Message counter</li> <li>• Priority rate</li> </ul>

## IO Connections Menu

Click **IO Connections** to access this information:

Parameter	Description
<b>IO Connections</b>	<ul style="list-style-type: none"> <li>• <b>Configured CXM Watchdog:</b> the timeout of input reception to switch in FALLBACK state</li> <li>• Production and consumption connection ID</li> <li>• Production and consumption RPI</li> <li>• Production and consumption API</li> <li>• Production and consumption connection parameters</li> </ul>
<b>Explicit Messaging</b>	<ul style="list-style-type: none"> <li>• <b>Number of explicit connected:</b> maximum instance number of the object</li> <li>• Originator connection ID and originator IP</li> <li>• <b>Msg Send Counter:</b> incremented each time a class 3 CIP message is sent on the connection.</li> <li>• <b>Msg Receive Counter:</b> incremented each time a class 3 CIP message is received on the connection.</li> </ul> <p><b>NOTE:</b> When clicking the <b>Next EM</b> button, it displays the next explicit message.</p>
<b>Field Bus Info</b>	Displays operating modes: <ul style="list-style-type: none"> <li>• <b>CXMFbMaxScan:</b> maximum period in the scanning time of field devices (in ms)</li> <li>• <b>CXMFbLastScan:</b> last period in the scanning time of field devices (in ms)</li> <li>• <b>CXMFbMinScan:</b> minimum period in the scanning time of field devices (in ms)</li> <li>• <b>CXMFbBandwidth:</b> percentage of the fieldbus cycle that is consumed to manage the data exchange</li> <li>• <b>Pending Acyclic request:</b> number of received explicit requests that are not already processed.</li> </ul>

## CAN Diagnostics Menu

Click **CAN Diagnostics** to access this information:

Parameter	Description
<b>Tx</b>	Indicates the number of: <ul style="list-style-type: none"> <li>• Bytes transmitted</li> <li>• Frames transmitted per second</li> </ul>
<b>Rx</b>	Indicates the number of: <ul style="list-style-type: none"> <li>• Bytes received</li> <li>• Frames received per second</li> </ul>
<b>Counter CAN</b>	Gives information on: <ul style="list-style-type: none"> <li>• <b>Overrun</b>: reception buffer overrun counter. It displays the minimum number of frames lost.</li> <li>• <b>Errors</b>: CAN transmission or reception error detected counter</li> <li>• <b>Bus OFF</b>: CAN controller bus off status counter</li> <li>• <b>Baudrate</b>: rate of transmission (in Kbits/s)</li> <li>• <b>Bus load</b>: minimum, current, and maximum network load</li> </ul>

## CANopen Diagnostics Menu

Click **CANopen Diagnostics** to access this information:

Parameter	Description
<b>SYNC ID</b>	Identification number for the synchronization object
<b>SYNC Period</b>	Transmission period of the synchronization object
<b>CXMFbHealth</b>	Gives the network manager status for diagnosing fieldbus: <ul style="list-style-type: none"> <li>• 0: Idle</li> <li>• 1: Fieldbus error detected</li> <li>• 2: Device fault detected</li> <li>• 3: Device error detected</li> <li>• 4: Device fault detected and error detected</li> </ul>
<b>Number of Equipments</b>	Amount of equipments in the configuration.
<b>Total Input Bytes</b>	Number of bytes mapped as input.
<b>Total Output Bytes</b>	Number of bytes mapped as output.

Parameter	Description
<b>Error Emcy_10xx</b>	Generic detected error count: Number of received emergency messages with code 10xx hex.
<b>Error Emcy_50xx</b>	Device hardware detected error count: Number of received emergency messages with code 50xx hex.
<b>Error Emcy_60xx</b>	Device software detected error count: Number of received emergency messages with code 60xx hex.
<b>Error Emcy_81xx</b>	Communication detected error count: Number of received emergency messages with code 81xx hex.
<b>Error Emcy_82xx</b>	Protocol detected error count: Number of received emergency messages with code 82xx hex.
<b>Error Emcy_90xx</b>	External detected error count: Number of received emergency messages with code 90xx hex.
<b>Error Emcy_FFxx</b>	Specific to the device: Number of received emergency messages with code FFxx hex.

## Slave Details Menu

Click **Slave Details** to access the CANopen slave states information. Each device is symbolized by a rectangle colored according to its status.

Color	CANopen Slave Status
White	CANopen node not used (or device not configured)
Green with ✓	<i>OPERATIONAL</i>
Orange	<i>PRE-OPERATIONAL</i>
Red	<i>ERR</i> (Device configured with fault)
Brown	<i>FAULT</i> (Device Inoperative)
Blue	<i>DISABLE</i> (Device configured but disabled)
Yellow	<i>STOPPED</i>
Grey	Webserver is offline

The letter **E** signals an emergency and/or SDO error for the slave device. Click the device to access the detailed information:

Parameter	Description
<b>Device Index</b>	Give the index of the device number among the list of programmed CANopen slave devices.
<b>Emergency Messages Counter</b>	Displays the emergency messages counter for the device.
<b>Emergency</b>	The equipment has emitted an emergency message signaling an event. For more information, refer to <a href="#">Emergency Objects</a> , page 140.  The table gives the 4 most recent messages with code, description and the time when the event occurs.
<b>Event History</b>	Gives the list of error detected during SDO transfer. For more information, refer to <a href="#">CANopen SDO command</a> , page 156.
<p><b>NOTE:</b> When clicking the <b>Reset</b> button, the <b>Emergency Messages Counter</b> is set to 0 and the lists (<b>Emergency</b> and <b>Event History</b>) are cleared. In addition the letter <b>E</b> is removed from the CANopen slave device symbolization.</p>	

## Emergency Objects

### Overview

Emergency objects (EMCY) of CANopen communication have been defined for diagnostic applications. EMCY objects can be accessed explicitly from the application using READ\_SDO.

The COB-ID of these objects contain the identity of the node of the device that produced the emergency message. The COB-ID of EMCY objects are constructed in the following manner:

$$\text{COB-ID}_{\text{EMCY}} = 0x80 + \text{node identity}$$

### Structure

The data field of an EMCY object is composed of 8 bytes containing:

- Emergency detected error code (2 bytes),
- The detected error register (1 byte),
- The factory-specific error information (5 bytes).

This illustration shows the structure of an EMCY object:

COB-ID	Error code		Register error	Error Information manufacturer specific					
	Byte 0	Byte 1		Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
0x80+node-ID									

You can consult the 4 last emergency messages received in chronological order in the **Device list > Event History** menu of the webpages, page 139.

**NOTE:** Regarding Safety considerations, “Emergency objects” and “Fatal error” are mentioned in this manual in conformance with the definition from the DS301 document of the CiA (CAN in Automation).

The contents of the detected error code and error register are specified by CiA.

## Detected Error Code 00xx

This table describes the content of detected error code 00xx:

Detected error code (hex)	Description
00xx	Error reset to zero or no error

## Detected Error Code 10xx

This table describes the content of detected error code 10xx:

Detected error code (hex)	Description
10xx	Generic error

## Detected Error Code 2xxx

This table describes the content of detected error code 2xxx:

Detected error code (hex)	Description
20xx	Current
21xx	Current, input side of the device

Detected error code (hex)	Description
22xx	Internal current to the device
23xx	Current, output side of the device

## Detected Error Code 3xxx

This table describes the content of detected error code 3xxx:

Detected error code (hex)	Description
30xx	Voltage
31xx	Principal voltage
32xx	Internal voltage to the device
33xx	Output voltage

## Detected Error Code 4xxx

This table describes the content of detected error code 4xxx:

Detected error code (hex)	Description
40xx	Temperature
41xx	Ambient temperature
42xx	Device temperature

## Detected Error Code 50xx

This table describes the content of detected error code 50xx:

Detected error code (hex)	Description
50xx	Device hardware

## Detected Error Code 6xxx

This table describes the content of detected error code 6xxx:

Detected error code (hex)	Description
60xx	Device software
61xx	Internal software
62xx	User software
63xx	Data set

## Detected Error Code 70xx

This table describes the content of detected error code 70xx:

Detected error code (hex)	Description
70xx	Additional modules

## Detected Error Code 8xxx

This table describes the content of detected error code 8xxx:

Detected error code (hex)	Description
80xx	Monitoring
81xx	Communication
8110	CAN overflow (objects lost)
8120	CAN in passive error mode
8130	Life Guard error or Heartbeat error
8140	Recovered from bus
8150	Collision during COB-ID transmission
82xx	Protocol error
8210	PDO not processed due to length error
8220	PDO length exceeded

## Detected Error Code 90xx

This table describes the content of detected error code 90xx:

Detected error code (hex)	Description
90xx	External error

## Detected Error Code Fxxx

This table describes the content of detected error code Fxxx:

Detected error code (hex)	Description
F0xx	Additional functions
FFxx	Specific to the device



# Firmware Upgrade

## What's in This Chapter

Firmware Update with Automation Device Maintenance.....	145
Firmware Update with Unity Loader.....	145

## Introduction

This chapter describes the steps for upgrading the firmware for the BMECXM CANopen communications module.

# Firmware Update with Automation Device Maintenance

## Overview

The EcoStruxure™ Automation Device Maintenance is a standalone tool that allows and simplifies the firmware update of devices in a plant (single or multiple).

The tool supports the following features:

- Automatic device discovery
- Manual device identification
- Certificate management
- Firmware update for multiple devices simultaneously

**NOTE:** For a description of the download procedure, refer to the *EcoStruxure™ Automation Device Maintenance, User Guide*.

# Firmware Update with Unity Loader

## Overview

You can update the firmware for the BMECXM module by downloading a new firmware version with Unity Loader. The minimum version for Unity Loader is V11.0.

The firmware download can be performed by connecting to the Ethernet network.  
Refer to the *Unity Loader, User Guide* for a description of the download procedure.

## Password

A firmware password is set in Control Expert and sent to each BMECXM module firmware. To access the Control Expert password in the **Project Browser**, right-click **Project > Properties of Project > Protection**.

To perform the update, check that the password set in Unity Loader matches the password set in Control Expert.

## Preparation

Before performing the firmware update:

- Check that the FTP service, page 96 is enabled.
- Stop the PLC.
- Open Unity Loader on your PC. (**Start > Programs > Schneider Electric > Unity Loader**).

### **▲ WARNING**

#### **UNKNOWN OPERATIONAL STATE OF EQUIPMENT**

Evaluate operational state of equipment before stopping the PLC.

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

**NOTE:** If you do not stop the PLC before trying to transfer firmware, you are informed by Unity Loader that the PLC must be stopped. After confirming this message, Unity Loader stops the PLC automatically.

---

# Appendices

## What's in This Part

CANopen Master Local Object Dictionary Entry..... 148  
CANopen Commands ..... 156  
CIP Objects ..... 162

## Overview

These appendices contain information that should be useful for programming the application.

# CANopen Master Local Object Dictionary Entry

## What’s in This Chapter

Object Dictionary Entries According to Profile DS301..... 148  
 Object Dictionary Entries According to Profile DS302..... 151  
 BMECXM Manufacturer Specific Object Dictionary  
 Entries ..... 153

## Subject of This Chapter

This chapter contains the local object dictionary entry for CANopen Master.

## Object Dictionary Entries According to Profile DS301

### Object Dictionary Entries

The table below presents the object dictionary entries according to profile DS301:

In- dex (hex)	Object Name	Sub- index (hex)	Description	Data Type	Comments
1000	Device type	00	Type of device	Unsigned32	000F 0191 hex
1001	Error register	00	Bit 0 indicates a generic error.	Unsigned8	–
1005	COB-ID SYNC	00	Define COB-ID of the synchronisation object (SYNC)	Unsigned32	–
1006	Communication Cycle Period	00		Unsigned32	–
1007	Synchronous window length	00		Unsigned32	–
1008	Manufacturer device name	00		Visible string (15)	BME CXM 0100

In- dex (hex)	Object Name	Sub- index (hex)	Description	Data Type	Comments
1009	Manufacturer hardware version	00		Visible string (15)	Current hardware revision from V1.2.0.1
100A	Manufacturer software version	00		Visible string (15)	Current software revision from V1.0
1012	COB-ID time stamp message	–	Define COB-ID of the time stamp object (TIME)	Unsigned32	–
1016	Consumer Heartbeat Time	00	Largest Subindex supported Number of entries: 64	Unsigned8	40 hex
		01...-40	The consumer heartbeat time defines the expected heartbeat cycle time and has to be higher than the corresponding producer heartbeat time (multiple of 1 ms).	Unsigned32	Node ID + Heartbeat Time: <ul style="list-style-type: none"> <li>• Bits 31-24: reserved</li> <li>• Bits 23-16: Node ID</li> <li>• Bits 15-0: Heartbeat Time</li> </ul>
1017	Producer Heartbeat Time	00	The producer heartbeat time defines the cycle time of the heartbeat (multiple of 1 ms).	Unsigned16	–
1018	Identity object	00	Number of entries	Unsigned8	04 hex
		01	Vendor ID	Unsigned32	0600 005A hex
		02	Product code	Unsigned32	081C xxxx hex
		03	Revision number	Unsigned32	0001 xxxx hex
		04	Serial number	Unsigned32	–
1020	Verify configuration	00	Number of entries	Unsigned8	02 hex
		01	Configuration date	Unsigned32	–
		02	Configuration time	Unsigned32	–
102A	NMT inhibit time	00		Unsigned16	–
1200	Server SDO Parameter	00	Number of entries	Unsigned8	02 hex
		01	COB-ID client -> server (Rx)	Unsigned32	600 hex + Node-ID
		02	COB-ID server -> client (Tx)	Unsigned32	580 hex + Node-ID
1280 ...	Client SDO Parameter 1...3	00	Number of entries	Unsigned8	–
		01	COB-ID client -> Server (Rx)	Unsigned32	

In- dex (hex)	Object Name	Sub- index (hex)	Description	Data Type	Comments
1282		02	COB-ID server -> Client (Tx)	Unsigned32	
		03	Node-ID of the SDO server	Unsigned8	
1400 ... 14FF	Receive PDO parameter 1...256	00	Largest subindex supported	Unsigned8	–
		01	COB-ID used by PDO	Unsigned32	
		02	Transmission type	Unsigned8	
		03	–	Unsigned16	
		04	–	Unsigned8	
		05	Event timer	Unsigned16	
1600 ... 16FF	Receive PDO mapping 1...256	00	Number of mapped application objects in PDO	Unsigned8	Depends on PDO mapping of the application
		01	PDO mapping for the first application object to be mapped	Unsigned32	Index (16 bit)   subindex (8 bit)   length (8 bit)
		02	PDO mapping for the second application object	Unsigned32	–
		.....	–	–	–
		08	PDO mapping for the eighth application object	Unsigned32	–
1800 ... 18FF	Transmit PDO parameter 1...256	00	Largest subindex supported	Unsigned8	–
		01	COB-ID used by PDO	Unsigned32	
		02	Transmission type	Unsigned8	
		03	Inhibit time	Unsigned16	
		04	Reserved	Unsigned8	
		05	Event timer	Unsigned16	
1A00 ... 1AFF	Transmit PDO mapping 1...256	0	Number of mapped application objects in PDO	Unsigned8	Depends on PDO mapping of the application
		1	PDO mapping for the first application object to be mapped	Unsigned32	Index (16 bit)   subindex (8 bit)   length (8 bit)
		2	PDO mapping for the second application object	Unsigned32	–

In- dex (hex)	Object Name	Sub- index (hex)	Description	Data Type	Comments
		.....	–	–	–
		8	PDO mapping for the eighth application object	Unsigned32	–

## Object Dictionary Entries According to Profile DS302

### Object Dictionary Entries

The table below presents the object dictionary entries according to profile DS302.

Index (Hex)	Subindex	Description	Object type	Data type
1F22	–	Concise DCF	ARRAY	–
	0	Number of entries	VAR	Unsigned8
	1	Device with Node-ID 1	VAR	DOMAIN
	...	–	–	–
	127	Device with Node-ID 127	–	DOMAIN
1F26	–	Expected configuration date	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F27	–	Expected configuration time	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F80	–	NMT startup	VAR	Unsigned32
1F81	...	Slave assignment	ARRAY	–

Index (Hex)	Subindex	Description	Object type	Data type
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F82	...	Request NMT	ARRAY	–
	0	Number of entries		Unsigned8
	1	Request NMT for Node-ID 1		Unsigned8
	...	–		–
	128	Request NMT for all nodes		Unsigned8
1F84	...	Device type identification	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F85	...	Vendor identification	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F86	...	Product code	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F87	...	Revision number	ARRAY	–
	0	Number of entries		Unsigned8
	1	Device with Node-ID 1		Unsigned32
	...	–		–
	127	Device with Node-ID 127		Unsigned32
1F8A	–	Restore configuration	ARRAY	–



Index (Hex)	Subindex	Description	Object type	Data type
	0	Number of entries		Unsigned8
	1	Restore for Node-ID 1		Unsigned8
	...	–		–
	64	Restore for Node-ID 64		Unsigned8

## BMECXM Manufacturer Specific Object Dictionary Entries

### Boot Slave Control Reset

The table below presents the object entry 4210:

Index (Hex)	Subindex	Description	Object type	Data type
4210	–	Boot slave control reset	ARRAY	–
	0	Number of entries		Unsigned8
	1	Reset for Node-ID 1 <sup>(1)</sup>		Unsigned8
	...	–		–
	64	Reset for Node-ID 64 <sup>(1)</sup>		Unsigned8
<p><b>(1)</b></p> <ul style="list-style-type: none"> <li>• Data = 0: No reset</li> <li>• Data = 1: Reset all parameters (by default)</li> <li>• Data = 2: Reset communication parameters only (1000 hex-1FFF hex)</li> <li>• Data &gt;2: Not used</li> </ul>				

### Boot Slave Control Start

The table below presents the object entry 4211:

Index (Hex)	Subindex	Description	Object type	Data type
4211	–	Boot slave control start	ARRAY	–

Index (Hex)	Subindex	Description	Object type	Data type
	0	Number of entries		Unsigned8
	1	Start for Node-ID 1 <sup>(1)</sup>		Unsigned8
	...	–		–
	64	Start for Node-ID 64 <sup>(1)</sup>		Unsigned8
<b>(1)</b> <ul style="list-style-type: none"> <li>• Data = 0: No start</li> <li>• Data = 1: Start all parameters (by default)</li> <li>• Data &gt;1: Not used</li> </ul>				

## Force Download

The table below presents the object entry 4212:

Index (Hex)	Subindex	Description	Object type	Data type
4212	–	Force download	ARRAY	–
	0	Number of entries		Unsigned8
	1	Type of force download for node 1 <sup>(1)</sup>		Unsigned8
	...	–		–
	64	Type of force download for node 64 <sup>(1)</sup>		Unsigned8
<b>(1)</b> <ul style="list-style-type: none"> <li>• Data = 0: No forcing (by default)</li> <li>• Data = 1: Force download of communication parameters</li> <li>• Data = 2: Force download of application parameters</li> <li>• Data &gt;2: Not used</li> </ul>				

## Global SDO Timeout

The table below presents the object entry 5FF0:

Index (Hex)	Subindex	Description	Object type	Data type
5FF0	–	Global SDO timeout	VAR	Unsigned16

## Slave Specific SDO Timeout

The table below presents the object entry 5FF1:

Index (Hex)	Subindex	Description	Object type	Data type
5FF1	–	Slave specific SDO timeout	ARRAY	–
	0	Number of entries		Unsigned8
	1	Timeout for Node-ID 1		Unsigned16
	...	–		–
	64	Timeout for Node-ID 64		Unsigned16

# CANopen Commands

## What's in This Chapter

CANopen SDO Commands .....	156
CANopen SDO Abort Code .....	158
CANopen Start Command .....	160
CANopen Slave Enabling Command .....	161

## Subject of This Chapter

This chapter defines CANopen commands that are specific to BMECXM modules.

# CANopen SDO Commands

## Overview

*WRITE\_SDO* and *READ\_SDO* objects are used to send NMT commands. The SDO abort code is used when the SDO command is not successful.

## *WRITE\_SDO* Command

This table shows the header of the *WRITE\_SDO* command:

Value (hex)	CIP generic message parameter
65	Class ID
40	Service Number
1	Instance
x	Length

This table shows the command data and the response data for the *WRITE\_SDO*:

Value	Size	Parameter
Command data		
0: PLC	SINT	Connect ID

Value	Size	Parameter
1...4: DTM		
[1...127]	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
[1...255]	INT	Length
User defined	SINT[...]	Data
Positive response data		
[1...127]	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
0	INT	Status
Negative response data		
[1...127]	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
≠0 (see EcoStruxure™ Control Expert, Communication, Block Library)	SINT	Status
See table, page 159	SINT[4]	SDO abort code

## READ\_SDO Command

This table shows the header of the *READ\_SDO* command:

Value (hex)	CIP generic message parameter
65	Class ID
41	Service Number
1	Instance
x	Length

This table shows the command data and the response data for the *READ\_SDO*:

Value	Size	Parameter
Command data		
0: PLC 1...4: DTM	SINT	Connect ID
[1...127]	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
[1...255]	INT	Length
Positive response data		
[1...127]	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
0	INT	Status
User defined	INT	Length
Value requested	SINT[...]	Object value
Negative response data		
0x2B	SINT	Node ID
User defined	INT	Index
User defined	SINT	Subindex
≠0 (see EcoStruxure™ Control Expert, Communication, Block Library)	SINT	Status
See table, page 159	SINT[4]	SDO abort code

## CANopen SDO Abort Code

### Overview

The SDO abort code is used when the SDO command is not successful.

## SDO Abort Code

SDO Abort Code Value (hex)	Parameter
0503 0000	Toggle bit not alternated.
0504 0000	SDO protocol timed out.
0504 0001	Client/server command specifier not valid or unknown.
0504 0002	Invalid block size (block mode only).
0504 0003	Invalid sequence number (block mode only).
0504 0004	CRC error (block mode only).
0504 0005	Out of memory.
0601 0000	Unsupported access to an object.
0601 0001	Attempt to read a write only object.
0601 0002	Attempt to write a read only object.
0602 0002	Object does not exist in the object dictionary.
0604 0041	Object cannot be mapped to the PDO.
0604 0042	The number and length of the objects to be mapped would exceed PDO length.
0604 0043	General parameter incompatibility reason.
0604 0047	General internal incompatibility in the device.
0606 0000	Access failed due to an hardware error
0607 0010	Data type does not match, length of service parameter does not match
0607 0012	Data type does not match, length of service parameter too high
0607 0013	Data type does not match, length of service parameter too low
0609 0011	Sub-index does not exist.
0609 0030	Value range of parameter exceeded (only for write access).
0609 0031	Value of parameter written too high.
0609 0032	Value of parameter written too low.
0609 0036	Maximum value is less than minimum value.
0800 0000	General error
0800 0020	Data cannot be transferred or stored to the application.
0800 0021	Data cannot be transferred or stored to the application because of local control.

SDO Abort Code Value (hex)	Parameter
0800 0022	Data cannot be transferred or stored to the application because of the present device state.
0800 0023	Object dictionary dynamic generation fails or no object dictionary is present (for example, object dictionary is generated from file and generation fails because of a file error).

## CANopen Start Command

### Overview

The *EM\_Start* command is used to synchronize the start of the BMECXM module.

**NOTE:** This command is valid in manual mode only. For more information, refer to starting mode, page 96.

### *EM\_Start* Command

This table shows the header of the *EM\_Start* command:

Value (hex)	CIP generic message parameter
66	Class ID
40	Service Number
1	Instance
x	Length

This table shows the command data and the response data for the *EM\_Start*:

Value	Size	Parameter
Command data		
–	–	Not applicable
Response data		
[0...1]	SINT	Status: <ul style="list-style-type: none"> <li>0: accepted</li> <li>1: refused</li> </ul>



# CANopen Slave Enabling Command

## Overview

The *Slave Enable / Disable* command is used to disable a device which is configured or enable a device which as been already disabled by calling this function before.

## Slave Enable / Disable Command

This table shows the header of the *Slave Enable / Disable* command:

Value (hex)	CIP generic message parameter
67	Class ID
40	Service Number
1	Instance
x	Length

This table shows the command data for the *Slave Enable / Disable*:

Value	Size	Parameter
Command data		
0...2	SINT	For node ID 1: <ul style="list-style-type: none"> <li>• 0: Do not change state</li> <li>• 1: Enable slave</li> <li>• 2: Disable slave</li> </ul>
...	...	For node n
[0...2]	SINT	For node ID 126: <ul style="list-style-type: none"> <li>• 0: Do not change state</li> <li>• 1: Enable slave</li> <li>• 2: Disable slave</li> </ul>

# CIP Objects

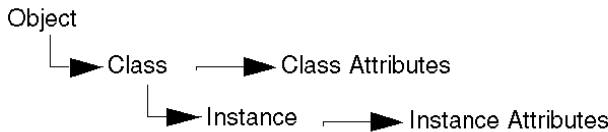
## What's in This Chapter

<i>DIAG_FXM_Diagnostic</i> Object.....	162
<i>DIAG_CXM</i> Object.....	167
EIP Interface Diagnostic Object.....	170
I/O Connection Diagnostics Object.....	173
EtherNet/IP Explicit Connection Diagnostic Object .....	175

## Subject of This Chapter

Modicon M580 applications use CIP within a producer/consumer model to provide communication services in an industrial environment. The M580 CPU can access CIP data and services located in connected devices.

CIP object data and content are exposed-and accessed- hierarchically in the following nested levels:



**NOTE:** You can use explicit messaging to access these items:

- Access a collection of instance attributes by including only the class and instance values for the object in the explicit message.
- Access a single attribute by adding a specific attribute value to the explicit message with the class and instance values for the object.

This chapter describes the available CIP objects you can use to diagnose the BMECXM module.

## *DIAG\_FXM\_Diagnostic* Object

### Overview

Basic diagnostic of the CANopen X80 master module can be done via explicit message using the *DIAG\_FXM\_Diagnostic* object.

The diagnostic object presents the instances, attributes, and services described below.

## Class ID

301 hex

## Instance IDs

The diagnostic object presents two instance values:

- 0: class
- 1: instance

## Attributes

The diagnostic object presents the following attributes.

Instance ID = 0 (class attributes):

Attribute ID (hex)	Type	Description
01	WORD	Version high
02	WORD	Version low
03	WORD	Number of instance

Instance ID = 1 (instance attributes):

Attribute ID (hex)	Parameter	Type	Bit	Description
01	<b>MAC Address</b>	DWORD		Ethernet MAC address of the module <b>NOTE:</b> only the 4 Least Significant Byte (LSB), for completing the MAC address add 00-80 on the Most Significant Byte (MSB).)
02	PBA version	DWORD		4 bytes for major, minor, intermediate, and release (reserved)
03	Micro FW1 Version	WORD		2 bytes for major, and minor
04	Micro FW2 Version	WORD		2 bytes for major, and minor

Attribute ID (hex)	Parameter	Type	Bit	Description
05	Firmware Version	DWORD		4 bytes for major, minor, intermediate, and release (reserved)
06	Configuration Supported	DWORD		2 bytes for major, and minor
07	Device Name(16)	DWORD		Device name of the module
08	IP address	DWORD		Current Ethernet IPV4 address (format is xxx.xxx.xxx.xxx)
09	Subnet Mask	DWORD		Current Ethernet IPV4 subnet (format is xxx.xxx.xxx.xxx)
0A	Default Gateway	DWORD		Current Ethernet IPV4 Gateway address (format is xxx.xxx.xxx.xxx)
0B	CPU Rate Available	DUINT		Percentage (%) of CPU time available
0C	FxmOpState	BYTE		0: <i>INITIALIZATION</i> 1: <i>UNCONFIGURED</i> 2: <i>CONFIGURED</i> 3: <i>CONNECTED RUN</i> 4: <i>CONNECTED STOP</i> 5: <i>FALLBACK</i>
0D	FxmRedundState	BYTE		Reserved
0E	FxmDisplay	WORD		2 bits (high, low) per LED: <b>RUN</b> : bits (1, 0) <b>ERR</b> : bits (3, 2) <b>I/O</b> : bits (5, 4) <b>BS</b> : bits (7, 6) <b>CAN RUN</b> : bits (9, 8) <b>CAN COM</b> : bits (11, 10) <b>CAN ERR</b> : bits (13, 12) <b>Off</b> bit high = 0 and bit low = 0 <b>Green</b> bit high = 0 and bit low = 1 <b>Red</b> bit high = 1 and bit low = 0 <b>Yellow</b> bit high = 1 and bit low = 1
0F	Ethernet Status	BYTE		Main Ethernet status:
			0	<b>PORT1_LINK</b> :

Attribute ID (hex)	Parameter	Type	Bit	Description
				<ul style="list-style-type: none"> <li>0: Link down for Ethernet port 1</li> <li>1: Link up for Ethernet port 1</li> </ul>
			4	<b>RPI_CHANGE:</b> <ul style="list-style-type: none"> <li>0: EtherNet/IP RPI not in progress</li> <li>1: EtherNet/IP RPI in progress</li> </ul>
			5	<b>REDUNDANCY_STATUS:</b> <ul style="list-style-type: none"> <li>0: Backup path not available</li> <li>1: Backup path available</li> </ul>
			6	<b>REDUNDANCY_OWNER:</b> <ul style="list-style-type: none"> <li>0: Redundant owner not present</li> <li>1: Redundant owner present</li> </ul>
			7	<b>GLOBAL_STATUS:</b> <ul style="list-style-type: none"> <li>0: One or more services not operating normally</li> <li>1: All services operating normally</li> </ul>
10	Ethernet Services	BYTE		Detailed Ethernet status:
			1	<b>SNTP_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>
			3	<b>SNMP_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>
			4	<b>FDR_SERVICE:</b> <ul style="list-style-type: none"> <li>0: Unable to download PRM file</li> <li>1: Service operating normally or disabled</li> </ul>
			5	<b>FIRMWARE_UPGRADE</b> <ul style="list-style-type: none"> <li>0: Firmware upgrade unauthorized</li> <li>1: Service operating normally</li> </ul>
			6	<b>WEB_PAGE</b> <ul style="list-style-type: none"> <li>0: Webpage not available</li> <li>1: Service operating normally or disabled</li> </ul>
			7	<b>EVENT_LOG_STATUS</b> <ul style="list-style-type: none"> <li>0: Service not operating normally</li> <li>1: Service operating normally or disabled</li> </ul>

Attribute ID (hex)	Parameter	Type	Bit	Description
11	Syslog_Status	BYTE		
			0	Set to 1 if the syslog client does not receive the acknowledgment of the TCP messages from the syslog server.
12	Syslog_Buffer_Free	DUINT		Free place in % of the events buffer.
13	Syslog_Lost_Events	DUINT		Number of events lost since the last restart.
14	FxmFBState	BYTE		Fieldbus operating mode: 0 = <i>IDLE</i> 1 = <i>NO-CONF</i> 2 = <i>BUS OFF</i> 3 = <i>STOPPED</i> 4 = <i>PRE-OPERATIONAL</i> 5 = <i>OPERATIONAL</i> 6 = <i>CLEAR</i>
15	FxmFBHealth	BYTE		Status information of the network manager for diagnosing fieldbus: 0 = Idle 1 = Fieldbus error detected 2 = Device fault detected 3 = Device error detected 4 = Device fault detected and error detected
16	SlavesProgList	BOOL [128]		Slave list for programmed state. 1 bit per slave device: <ul style="list-style-type: none"> <li>• 0: Programmed (in configuration file)</li> <li>• 1: Not expected (not configured or disabled)</li> </ul>
17	SlavesLiveList	BOOL [128]		Slave list for responding state. 1 bit per slave device: <ul style="list-style-type: none"> <li>• 0: Responding</li> <li>• 1: Not responding or disabled</li> </ul>
18	SlavesDiagList	BOOL [128]		Slave list for error state. 1 bit per slave device: <ul style="list-style-type: none"> <li>• 0: Detected error or fault on expected slave</li> </ul>

Attribute ID (hex)	Parameter	Type	Bit	Description
				<ul style="list-style-type: none"> <li>1: No reported error</li> </ul>
19	SlavesWaitList	BOOL [128]		Slave list that indicates if the device is waiting for explicit order to operate or not.  1 bit per slave device: <ul style="list-style-type: none"> <li>0: Slave device is waiting for explicit message to operate</li> <li>1: No action required</li> </ul>
1A	FxmFBMaxScan	UDINT		Maximal time period of field devices scanning (by 100 $\mu$ s resolution).
1B	FxmFBLastScan	UDINT		Last time period of field devices scanning (by 100 $\mu$ s resolution).
1C	FxmFBMinScan	UDINT		Minimal time period of field devices scanning (by 100 $\mu$ s resolution).
1D	Pending Acyclic request	UINT		Number of pending explicit requests.
1F	FxmFB Bandwidth	UINT		Current Fieldbus bus load in %

## Service Supported

The DIAG\_FXM\_Diagnostic object performs the following service:

Service ID (hex)	Service Name	Class	Instance
01	<i>Get_Attributes_All</i>	X	X

## DIAG\_CXM Object

### Overview

Diagnostic of the CANopen bus activity can be done via explicit message using the CXM\_DIAG object.

The diagnostic object presents the instances, attributes, and services described below.

## Class ID

302 hex

## Instance IDs

The diagnostic object presents two instance values:

- 0: class
- 1: instance

## Attributes

The diagnostic object presents the following attributes.

Instance ID = 0 (class attributes):

Attribute ID (hex)	Type	Description
01	WORD	Version high
02	WORD	Version low
03	WORD	Number of instance

Instance ID = 1 (instance attributes):

Attribute ID (hex)	Type	Description
01	DWORD	Total number of bytes received.
02	DWORD	Number of frames received since the beginning.
03	DWORD	Total number of bytes Transmitted.
04	DWORD	Number of Frames Transmitted since the beginning.
05	DWORD	Reception buffer overrun counter: minimum number of frames lost
06	DWORD	CAN transmission or reception detected error counter. (include all detected error described in error flag of the CAN2.0B protocol)
07	DWORD	Minimum bus load in %
08	DWORD	Current bus load in %



Attribute ID (hex)	Type	Description
09	DWORD	Maximum bus load in %
0A	DWORD	Rate of transmission (Kbits/s).
0B	DWORD	CAN controller Bus Off status counter.
0C	DWORD	0 is no bus off 1 is Bus off
0D	DWORD	Identification number for the SYNC synchronization object.
0E	DWORD	Period Sync ID object.
0F	DWORD	Current number of found error frames in % for last 10000 exchanged frames
10	DWORD	Maximum number of found error frames in %
11	DWORD	Minimum number of error found frames in %
12	DWORD	Generic detected error count: Number of received emergency messages with code 10xx hex
13	DWORD	Device hardware detected error count: Number of received emergency messages with code 50xx hex
14	DWORD	Device software detected error count: Number of received emergency messages with code 60xx hex
15	DWORD	Communication detected error count: Number of received emergency messages with code 81xx hex
16	DWORD	Protocol detected error count: Number of received emergency messages with code 82xx hex
17	DWORD	External detected error count: Number of received emergency messages with code 90xx hex
18	DWORD	Device-specific: Number of received emergency messages with code FFxx hex
19	DWORD	Maximum number of TPDOs to transmit during one cycle
1A	DWORD	Highest used Node ID
1B	DWORD	Number of used RxPDOs
1C	DWORD	Number of used TxPDOs
1D	DWORD	Total number of variables in the input process image.
1F	DWORD	Total number of variables in the output process image.

## Service Supported

The CXM\_DIAG object performs the following services upon the listed object types:

Service ID (hex)	Service Name	Class	Instance
01	<i>Get_Attributes_All</i>	X	X
05	<i>RESET</i>	–	X

## EIP Interface Diagnostic Object

### Overview

The diagnostic object presents the instances, attributes, and services described below.

### Class ID

350 hex

### Instance IDs

The diagnostic object presents two instance values:

- 0: class
- 1: instance

### Attributes

The diagnostic object presents the following attributes.

Instance ID = 0 (class attributes):

Attribute ID (hex)	Type	Description
01	WORD	Version high
02	WORD	Version low
03	WORD	Number of instance

Instance ID = 1 (instance attributes):

Attribute ID (hex)	Parameter	Type	Description
01	Protocol Supported	UINT	Protocol supported
02	Connection DIAG	Structure of:	
	Max CIP IO Cnx Opened	UINT	Maximum number of CIP IO connections opened
	Current CIP IO Cnx	UINT	Number of CIP IO connections currently opened
	Max CIP Explicit Cnx Opened	UINT	Maximum number of CIP explicit connections opened
	Current CIP Explicit Cnx	UINT	Number of CIP explicit connections currently opened
	CIP Cnx Explicit opening Errors	UINT	Incremented at each attempt to open a CIP connection that fails
	CIP Cnx Timeout Errors	UINT	Incremented when a CIP connection is timed out
	Max EIP TCP Cnx Opened	UINT	Maximum number of TCP connection opened and used for EIP communication
03	Current EIP TCP Cnx Opened	UINT	Number of TCP connections currently opened and used for EIP communication
	IO Messaging DIAG	Structure of:	
	IO Prod 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 prod send Errors Counter	UINT	Incremented each Time a Class 0/1 message is not sent
04	IO consumption Receive Errors Counter	UINT	Incremented each time a consumption is received with an error
	Explicit Messaging DIAG	Structure of:	
	Class3 Msg Send counter	UDINT	Incremented each time a Class 3 CIP message is sent
	Class3 Msg Rec counter	UDINT	Incremented each time a Class 3 CIP message is received
	UCMM Msg Send counter	UDINT	Incremented each time an UCMM message is sent

Attribute ID (hex)	Parameter	Type	Description
	UCMM Msg Receive counter	UDINT	Incremented each time an UCMM message is received
05	COM Capacity	Structure of:	
	Capacity Max CIP Cnx	UINT	Max supported CIP connections
	Capacity Max TCP Cnx	UINT	Max supported TCP connections
	Capacity Max Urgent priority rate	UINT	Max CIP transport class 0/1 urgent priority messages packets/s
	Capacity Max Scheduled priority rate	UINT	Max CIP transport class 0/1 scheduled priority messages packets/s
	Capacity Max High priority rate	UINT	Max CIP transport class 0/1 high priority messages packets/s
	Capacity Max Low priority rate	UINT	Max CIP transport class 0/1 low priority messages packets/s
	Capacity Max Explicit rate	UINT	Max CIP transport class 2/3 or other EIP messages packets/s
06	Bandwidth Diag	Structure of:	
	Current sending Urgent priority rate	UINT	CIP transport class 0/1 urgent priority messages packets/s sent
	Current recept Urgent priority rate	UINT	CIP transport class 0/1 urgent priority messages packets/s received
	Current sending Scheduled priority rate	UINT	CIP transport class 0/1 scheduled priority messages packets/s sent
	Current recept Scheduled priority rate	UINT	CIP transport class 0/1 scheduled priority messages packets/s received
	Current sending High priority rate	UINT	CIP transport class 0/1 high priority messages packets/s sent
	Current recept High priority rate	UINT	CIP transport class 0/1 high priority messages packets/s received
	Current sending Low priority rate	UINT	CIP transport class 0/1 low priority messages packets/s sent
	Current recept Low priority rate	UINT	CIP transport class 0/1 Low priority messages packets/s received
	Current sending Explicit rate	UINT	CIP transport class 2/3 or other EIP messages packets sent
	Current reception Explicit rate	UINT	CIP transport class 2/3 or other EIP messages packets received

## Service Supported

The object performs the following services upon the listed object types:

Service ID (hex)	Service Name	Class	Instance
01	<i>Get_Attributes_All</i>	X	X
05	<i>RESET</i>	–	X

## I/O Connection Diagnostics Object

### Overview

The diagnostic object presents the instances, attributes, and services described below.

### Class ID

352 hex

### Instance IDs

The diagnostic object presents two instance values:

- 0: class
- 1...256: instances

### Attributes

The diagnostic object presents the following attributes.

Instance ID = 0 (class attributes):

Attribute ID (hex)	Type	Description
01	UINT	Revision
02	UINT	Maximum instance

Instance ID = 1 to 256 (instance attributes):

Attribute ID (hex)	Parameter	Type	Description
01	IO connections	Structure of:	
	IO product counter	UDINT	Incremented at each production
	IO consumption counter	UDINT	Incremented at each consumption
	IO product send error	UINT	Incremented each time a production is not sent
	IO Consumption Receive error	UINT	Incremented each time a consumption is received with an error
	CIP Connection TimeOut errors	UINT	Incremented when a connection is timed out
	CIP Connection Opening errors	UINT	Incremented at each attempt to open a connection that fails
	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 comm status	UINT	Communication Status of the Outputs
02	Connection Diag	Structure of:	
	Production Connection ID	UDINT	Connection ID for Production
	Consumption Connection ID	UDINT	Connection ID for Consumption
	Production RPI (conf)	UDINT	RPI for production
	Production API {current}	UDINT	API for production
	Consumption RPI (conf)	UDINT	RPI for consumption
	Consumption API {current}	UDINT	API for consumption
	Production Connection parameters	UDINT	Connection parameters for production
	Consumption Connection parameters	UDINT	Connection parameters for consumption
	Local IP	UDINT	Description from TI 82 , CIP diag
	Local UDP port	UINT	Description from TI 82 , CIP diag
	Remote IP	UDINT	Description from TI 82 , CIP diag
	Remote UDP port	UINT	Description from TI 82 , CIP diag

Attribute ID (hex)	Parameter	Type	Description
	Production Multicast IP	UDINT	Multicast IP used for production
	Consumption Multicast IP	UDINT	Multicast IP used for consumption
	Protocol supported	UINT	Protocol(s) supported on the connection

## Service Supported

The object performs the following services upon the listed object types:

Service ID (hex)	Service Name	Class	Instance
01	<i>Get_Attributes_All</i>	X	X
0E	<i>Get_Attribute_Single</i>	–	X
4C	<i>Get_and_Clear</i>	–	X
X supported			
– Not supported			

# EtherNet/IP Explicit Connection Diagnostic Object

## Overview

The EtherNet/IP explicit diagnostic object presents the instances, attributes, and services described below.

## Class ID

353 hex

## Instance IDs

The diagnostic object presents two instance values:

- 0: class
- 1...N: instance (N = maximum concurrent number of explicit connections)

## Attributes

The diagnostic object presents the following attributes.

Instance ID = 0 (class attributes):

Attribute ID (hex)	Type	Description
01	UINT	Revision
02	UINT	Maximum instance

Instance ID = 1 (instance attributes):

Attribute ID (hex)	Parameter	Type	Description
01	Originator connection ID	UDINT	O to T Connection ID
02	Originator IP	UINT	–
03	Originator TCP port	UDINT	–
04	Target Connection ID	UDINT	T to O Connection ID
05	Target IP	UDINT	–
06	Target TCP port	UDINT	–
07	Msg Send Counter	UDINT	Incremented each time a Class 3 CIP message is sent on the connection.
08	Msg Receive Counter	UDINT	Incremented each time a Class 3 CIP message is received on the connection.

## Service Supported

The object performs the following services upon the listed object types:

Service ID (hex)	Service Name	Class	Instance
01	<i>Get_Attributes_All</i>	X	X



# Glossary

## A

### ARRAY:

Table containing elements of a single type. This is the syntax: ARRAY [<limits>] OF <Type>. For example:

- ARRAY [1..2] OF BOOL is a one-dimensional table with two elements of type BOOL.
- ARRAY [1..10, 1..20] OF INT is a two-dimensional table with 10x20 elements of type INT.

### asset management:

A software application that can configure, monitor, and manage devices used as part of an industrial automation system..

### AUX:

(*auxiliary*) Optional, periodic processor task that is run through its programming software. The AUX task is used to execute a part of the application requiring a low priority. This task is executed only if the MAST and FAST tasks have nothing to execute. The AUX task has two sections:

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

## B

### BOOL:

(*boolean type*) This is the basic data type used in computing. A BOOL variable can have either of these values: 0 (FALSE) or 1 (TRUE). A bit extracted from a word is of type BOOL, for example: %MW10.4.

## C

### CAN :

(*Controller area network*) Fieldbus originally developed for automobile applications and now used in many sectors.

### CiA :

(*CAN in Automation*) International organization of users and manufacturers of CAN devices.

**CIP™:**

*(common industrial protocol)* A comprehensive suite of messages and services for the collection of manufacturing automation applications (control, safety, synchronization, motion, configuration, and information). CIP allows you to integrate these manufacturing applications with enterprise-level Ethernet networks and the Internet. CIP is the core protocol of EtherNet/IP.

**class 3 connection:**

A CIP transport class 3 connection used for explicit messaging between EtherNet/IP devices.

**COB-ID:**

*(communication object identifier)* Unique identifier of a COB on a CANopen network. The identifier determines the priority of a COB.

**CPU:**

*(central processing unit)* Brain of an industrial manufacturing process, also known as the processor or controller. It automates a process as opposed to relay control systems. CPUs are computers suited to survive the harsh conditions of the industrial environment.

**D**

**device DDT (DDDT):**

*(Device derived data type)* Predefined by the manufacturer and not editable. It contains the I/O language elements of an I/O module.

**DHCP:**

*(dynamic host configuration protocol)* An extension of the BOOTP communications protocol that provides for the automatic assignment of IP addressing settings, including IP address, subnet mask, gateway IP address, and DNS server names. DHCP does not require the maintenance of a table identifying each network device. The client identifies itself to the DHCP server using either its MAC address, or a uniquely assigned device identifier. The DHCP service utilizes UDP ports 67 and 68.

**DIO network:**

A network containing distributed equipment, in which I/O scanning is performed by a CPU with DIO scanner service on the local rack. DIO network traffic is delivered after RIO traffic, which takes priority in an RIO network.

**DTM:**

(*Device type manager*) Device driver running on the host PC. It provides a unified structure for accessing device parameters, configuring and operating the devices, and troubleshooting devices. DTMs can range from a simple graphical user interface (GUI) for setting device parameters to a highly sophisticated application capable of performing complex real-time calculations for diagnosis and maintenance purposes. In the context of a DTM, a device can be a communications module or a remote device on the network.

Refer to FDT.

**E****EDS:**

(*electronic data sheet*) Simple text file that describes the configuration capabilities of a device. EDS files are generated and maintained by the manufacturer of the device.

**EMCY:**

(*emergency*) A trigger event, generated by an internal error/fault. This object is transmitted with each new error, since error codes are independent mechanisms.

**EtherNet/IP™:**

A network communication protocol for industrial automation applications. It combines the standard Internet transmission protocols of TCP/IP and UDP with the application layer common industrial protocol (CIP) to support both high-speed data exchange and industrial control. EtherNet/IP employs electronic data sheets (EDS) to classify each network device and its functionality.

**Ethernet:**

A 10 Mb/s, 100 Mb/s, or 1 Gb/s, CSMA/CD, frame-based LAN that can run over copper twisted pair or fiber optic cable, or wireless. The IEEE standard 802.3 defines the rules for configuring a wired Ethernet network; the IEEE standard 802.11 defines the rules for configuring a wireless Ethernet network. Common forms include 10BASE-T, 100BASE-TX, and 1000BASE-T, which can utilize category 5e copper twisted-pair cables and RJ45 modular connectors.

**explicit messaging:**

TCP/IP-based messaging for Modbus TCP and EtherNet/IP. It is used for point-to-point, client/server messages that include both data, typically unscheduled information between a client and a server, and routing information. In EtherNet/IP, explicit messaging is considered class 3 type messaging, and can be connection-based or connectionless.

## F

### **FAST:**

An event-triggered (FAST) task is an optional, periodic processor task that identifies high priority, multiple scan requests, which is run through its programming software. A FAST task can schedule selected I/O modules to have their logic solved more than once per scan. The FAST task has two sections:

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

### **FDR:**

*(fast device replacement)* A service that uses configuration software to replace an inoperable product.

### **FDT:**

*(Field device tool)* The technology that harmonizes communication between field devices and the system host.

### **FTP:**

*(file transfer protocol)* A protocol that copies a file from one host to another over a TCP/IP-based network, such as the Internet. FTP uses a client/server architecture as well as separate control and data connections between the client and server.

## G

### **gateway:**

A gateway device interconnects two different networks, sometimes through different network protocols. When it connects networks based on different protocols, a gateway converts a datagram from one protocol stack into the other. When used to connect two IP-based networks, a gateway (also called a router) has two separate IP addresses, one on each network.

## H

### **health bit:**

Variable that indicates the communication state of the channels.

### **HMI:**

*(Human machine interface)* System that allows interaction between a human and a machine.

**HTTPS:**

(*hypertext transfer protocol secure*) A networking protocol for distributed and collaborative information systems. HTTPS is the basis of data communication for the Web.

**I****implicit messaging:**

UDP/IP-based class 1 connected messaging for EtherNet/IP. Implicit messaging maintains an open connection for the scheduled transfer of control data between a producer and consumer. Because an open connection is maintained, each message contains data primarily, without the overhead of object information, plus a connection identifier.

**IP address:**

The 32-bit identifier, consisting of both a network address and a host address assigned to a device connected to a TCP/IP network.

**L****local rack:**

An M580 rack containing the CPU and a power supply. A local rack consists of one or two racks: the main rack and the extended rack, which belongs to the same family as the main rack. The extended rack is optional.

**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.

**mapping :**

Transformation of data consigned in a special and different format.

**MAST:**

(*master*) Deterministic processor task that is run through its programming software. The MAST task schedules the RIO module logic to be solved in every I/O scan. The MAST task has two 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.

## N

### **NIM:**

(*Network interface module*) A NIM resides in the first position on an STB island (leftmost on the physical setup). The NIM provides the interface between the I/O modules and the fieldbus master. It is the only module on the island that is fieldbus-dependent, a different NIM is available for each fieldbus.

### **NMT:**

(*Network management*) CANopen protocols that provide services for network initialization, detected error control, and device status control.

### **NTP:**

(*Network time protocol*) Protocol for synchronizing computer system clocks. The protocol uses a jitter buffer to resist the effects of variable latency.

## O

### **O->T:**

(*originator to target*) Refer to originator and target.

### **originator:**

In EtherNet/IP, a device is considered the originator when it initiates a CIP connection for implicit or explicit messaging communications or when it initiates a message request for unconnected explicit messaging.

## P

### **PDO:**

(*process data object*) An unconfirmed broadcast message 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.

## R

### **RIO network:**

An Ethernet-based network that contains three types of RIO devices: a local rack, an RIO drop, and a ConneXium extended dual-ring switch (DRS). Distributed equipment may also participate in a RIO network via connection to DRSs.

**RPDO:**

*(received process data object)* Refer to PDO.

**RPI:**

*(requested packet interval)* The time period between cyclic data transmissions 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 at each RPI.

**S****SDO:**

*(service data object)* A message used by the fieldbus 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).

**SNMP:**

*(simple network management protocol)* Protocol used in network management systems to monitor network-attached devices. The protocol is part of the Internet protocol suite (IP) as defined by the Internet engineering task force (IETF). It consists of network management guidelines, including an application layer protocol, a database schema, and a set of data objects.

**SNTP:**

*(simple network time protocol)* Refer to NTP.

**string:**

A variable that is a series of ASCII characters.

**subnet mask:**

The 32-bit value used to hide (or mask) the network portion of the IP address and reveal the host address of a device on a network using the IP protocol.

**T****T->O:**

*(target to originator)* Refer to originator and target.

**task:**

A group of sections and subroutines, executed cyclically or periodically for the MAST task and 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.

**TPDO:**

*(transmit process data object)* Refer to PDO.

**trap:**

A trap is an event directed by an SNMP agent that indicates one of these events:

- A change has occurred in the status of an agent.
- An unauthorized SNMP manager device has attempted to get data from (or change data on) an SNMP agent.

**V**

**variable:**

Memory unit that is addressed and modified by a program.



# Index

<b>A</b>	
authorized devices	
cybersecurity .....	97
AUX task .....	93
<b>B</b>	
backplane .....	25
BMECXM0100	
description .....	14
bootup tab .....	71
bus start .....	39
<b>C</b>	
CANopen	
bus parameters .....	82
connector .....	18
device configuration .....	62
device import .....	61
CIP object	
301 hex .....	162
302 hex .....	167
350 hex .....	170
352 hex .....	173
353 hex .....	175
communication - instructions	
READ_SDO .....	109
WRITE_SDO .....	112
communication profile .....	21
configuration tab .....	66
configuring the devices	
STB .....	75
Tsys U .....	75
configuring the servodrives	
ATV .....	75
Lexium 05 .....	75
CXM expert diagnosis tab .....	132
CXM status tab .....	130
cyber security	
authorized devices .....	97
cybersecurity .....	23
<b>D</b>	
device DDT .....	102
device disconnection .....	39
diagnostics .....	118
DTM	
application area .....	91
description .....	88
general layout .....	87
navigation area .....	89–90
status bar .....	91
<b>E</b>	
error control	
heartbeat .....	70
node guarding .....	70
tab .....	70
Ethernet configuration tab .....	78
event timer .....	67
explicit messages .....	132
<b>F</b>	
fallback .....	95
FAST task .....	93
firmware	
update .....	145
upgrade .....	145
firmware update .....	145
<b>H</b>	
hold up time .....	95
<b>I</b>	
inhibit time .....	67
<b>L</b>	
limits .....	24
CXM .....	35
M580 .....	35

**M**

MAST task ..... 93–94  
 motion function bloc ..... 24

**N**

NMT (network management) ..... 70  
 NTP ..... 100

**O**

object dictionary ..... 148  
 object dictionary tab ..... 73

**P**

PDO multi-mapping ..... 69  
 PDO tab ..... 67

**R**

READ\_SDO ..... 108–109  
     example ..... 116  
 replacing  
     CANopen module ..... 28  
     CANopen slave ..... 30  
 request packet interval ..... 94  
 restrictions ..... 24  
 RPI  
     performance ..... 38  
     values ..... 94

**S**

SDO  
     performance ..... 38  
 SDO timeout ..... 108  
 slave live list tab ..... 131  
 SNMP ..... 98  
 starting mode ..... 96  
 system architecture ..... 21

**T**

task  
     AUX ..... 93  
     characteristics ..... 93  
     cycle time ..... 38  
     FAST ..... 93  
     MAST ..... 93–94  
 transmission type ..... 67

**U**

Unity Loader  
     firmware update ..... 145  
 update  
     firmware ..... 145  
 update firmware ..... 145  
 upgrade  
     firmware ..... 145

**V**

variables  
     device DDT ..... 102

**W**

web pages  
     diagnostics ..... 79, 134  
 WRITE\_SDO ..... 108, 112  
     example ..... 116



Schneider Electric  
35 rue Joseph Monier  
92500 Rueil Malmaison  
France

+ 33 (0) 1 41 29 70 00

[www.se.com](http://www.se.com)

As standards, specifications, and design change from time to time, please ask for confirmation of the information given in this publication.

© 2023 Schneider Electric. All rights reserved.

EIO0000002129.06