

Modicon X80

BMXNOM0200 Serial Link Module

User Manual

Original instructions

09/2020

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

© 2020 Schneider Electric. All rights reserved.

Table of Contents



	Safety Information	7
	About the Book	11
Part I	Hardware Installation for Modbus Serial and Character Mode Communications	15
Chapter 1	Introduction to Serial Communications	17
	Modbus Serial and Character Mode Communications	18
	Presentation of the BMXNOM0200 Module	19
	Dimensions of X80 BMXNOM0200(H) Serial Link Module	25
	Standards and Certifications	26
	Installation of the BMXNOM0200 Module	27
Chapter 2	Serial Communication Architectures	31
	Modbus Line Termination and Polarization (RS485)	32
	Connecting Modbus Devices (RS485)	34
	Connecting Data Terminal Equipment (DTE) (RS232)	36
	Connecting Data Circuit-terminating Equipment (DCE) (RS232)	38
	Cabling	40
Part II	Software Implementation of Modbus Serial and Character Mode Communications	45
Chapter 3	BMXNOM0200 Limitation and Implementation Rules ..	47
	BMXNOM0200 Limitations Rules	48
	BMXNOM0200 Implementation Rules	50
Chapter 4	Modbus Serial Communication	53
4.1	Generalities	54
	About Modbus Serial	55
	Performance	56
	How to Access the Serial Link Parameters	58
4.2	Modbus Serial Communication Configuration	61
	Modbus Serial Communication Configuration Screen	62
	Application-linked Modbus Parameters	64
	Signal and Physical Line Parameters in Modbus	66
	Transmission-linked Modbus Parameters	69
	How to Set the BMXNOM0200 MODBUS Slave Address Without Control Expert?	71

4.3	Modbus Serial Communication Programming	73
	Services Supported by a Modbus Link Master Module	74
	Services Supported by a Modbus Link Slave Module	75
	Detail of Modbus Expert Mode	77
4.4	Debugging Modbus Serial Communication	83
	Modbus Serial Communication Debug Screen	83
Chapter 5	Character Mode Communication	87
5.1	Generalities	88
	About Character Mode Communication	88
5.2	Character Mode Communication Configuration	89
	BMXNOM0200 Character Mode Communication Configuration Screen	90
	Message End Detection Parameters in Character Mode	92
	Transmission Parameters in Character Mode	94
	Signal and Physical Line Parameters in Character Mode	96
5.3	Character Mode Communication Programming	99
	Character Mode Communication Functions	100
	Detail of Character Mode Expert Mode	104
5.4	Debugging Character Mode communication	107
	Character Mode Communication Debug Screen	107
Chapter 6	BMXNOM0200 Module Diagnostics	109
	Detailed Diagnostics by Communication Channel	110
	Diagnostics of a BMXNOM0200 Module	112
Chapter 7	Language Objects of Modbus and Character Mode Communications	115
7.1	Language Objects and IODDTs of Modbus and Character Mode Communications	116
	Introduction to the Language Objects for Modbus and Character Mode Communications	117
	Implicit Exchange Language Objects Associated with the Application-Specific Function	118
	Explicit Exchange Language Objects Associated with the Application-Specific Function	119
	Management of Exchanges and Reports with Explicit Objects	121
7.2	General Language Objects and IODDTs for Communication Protocols	124
	Details of IODDT Implicit Exchange Objects of Type T_COM_STS_GEN	125
	Details of IODDT Explicit Exchange Objects of Type T_COM_STS_GEN	126

7.3	Language Objects and IODDTs Associated with Modbus Communication	128
	Details concerning Explicit Exchange Language Objects for a Modbus Function.	129
	Details of the IODDTs Implicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT	130
	Details of the IODDTs Explicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT	131
	Details of language objects associated with configuration Modbus mode	134
7.4	Language Objects and IODDTs associated with Character Mode Communication	136
	Details concerning Explicit Exchange Language Objects for Communication in Character Mode.	137
	Details of IODDT Implicit Exchange Objects of Type T_COM_CHAR_BMX.	138
	Details of IODDT Explicit Exchange Objects of Type T_COM_CHAR_BMX.	139
	Details of language objects associated with configuration in Character mode	142
7.5	The IODDT Type T_GEN_MOD Applicable to All Modules.	144
	Details of the Language Objects of the IODDT of Type T_GEN_MOD	144
7.6	Language Objects and Device DDTs Associated with Modbus Communication	146
	BMX NOM 0200.x Device DDT	147
	MOD_FLT Byte Description.	150
Chapter 8	Dynamic Protocol Switching	151
	Changing Protocol with the BMXNOM0200 Module	151
Part III	Quick Start: BMXNOM0200 as a Modbus Slave over a Quantum PLC	155
Chapter 9	Overview.	157
	Product Overview	158
	Architecture Overview	159
	Limitations	161
Chapter 10	Configuration in Control Expert.	163
	Module Insertion	164
	Module Configuration Screen	165
Glossary	169
Index	177

Safety Information



Important Information

NOTICE

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



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



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

DANGER

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

WARNING

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

CAUTION

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

NOTICE

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

PLEASE NOTE

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

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

BEFORE YOU BEGIN

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

 WARNING
UNGUARDED EQUIPMENT
<ul style="list-style-type: none">• Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.• Do not reach into machinery during operation.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

START-UP AND TEST

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check be made and that enough time is allowed to perform complete and satisfactory testing.

WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book



At a Glance

Document Scope

This manual describes the principle for hardware and software implementation of character mode and Modbus communication with BMXNOM0200 communication modules.

Validity Note

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

The technical characteristics of the devices described in the present document also appear online. To access the information online:

Step	Action
1	Go to the Schneider Electric home page www.schneider-electric.com .
2	In the Search box type the reference of a product or the name of a product range. <ul style="list-style-type: none">• Do not include blank spaces in the reference or product range.• To get information on grouping similar modules, use asterisks (*).
3	If you entered a reference, go to the Product Datasheets search results and click on the reference that interests you. If you entered the name of a product range, go to the Product Ranges search results and click on the product range that interests you.
4	If more than one reference appears in the Products search results, click on the reference that interests you.
5	Depending on the size of your screen, you may need to scroll down to see the datasheet.
6	To save or print a datasheet as a .pdf file, click Download XXX product datasheet .

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


Related Documents

Title of documentation	Reference number
Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications	EIO0000002726 (English), EIO0000002727 (French), EIO0000002728 (German), EIO0000002730 (Italian), EIO0000002729 (Spanish), EIO0000002731 (Chinese)
Modicon M340, Processors, Setup Manual	35012676 (English), 35012677 (French), 35013351 (German), 35013352 (Italian), 35013353 (Spanish), 35013354 (Chinese)
Modicon M580, Hardware, Reference Manual	EIO0000001578 (English), EIO0000001579 (French), EIO0000001580 (German), EIO0000001582 (Italian), EIO0000001581 (Spanish), EIO0000001583 (Chinese)
EcoStruxure™ Control Expert, Operating Modes	33003101 (English), 33003102 (French), 33003103 (German), 33003104 (Spanish), 33003696 (Italian), 33003697 (Chinese)
EcoStruxure™ Control Expert, Communication, Block Library	33002527 (English), 33002528 (French), 33002529 (German), 33003682 (Italian), 33002530 (Spanish), 33003683 (Chinese)

Title of documentation	Reference number
EcoStruxure™ Control Expert, I/O Management, Block Library	33002531 (English), 33002532 (French), 33002533 (German), 33003684 (Italian), 33002534 (Spanish), 33003685 (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.schneider-electric.com/en/download.

Product Related Information

 WARNING
<p>UNINTENDED EQUIPMENT OPERATION</p> <p>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.</p> <p>Follow all local and national safety codes and standards.</p> <p>Failure to follow these instructions can result in death, serious injury, or equipment damage.</p>

Part I

Hardware Installation for Modbus Serial and Character Mode Communications

In This Part

This part provides an introduction to hardware installation for Modbus serial and Character Mode communications.

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
1	Introduction to Serial Communications	17
2	Serial Communication Architectures	31

Chapter 1

Introduction to Serial Communications

Subject of this Chapter

This chapter introduces the serial communications and give a description of the BMXNOM0200 module.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Modbus Serial and Character Mode Communications	18
Presentation of the BMXNOM0200 Module	19
Dimensions of X80 BMXNOM0200(H) Serial Link Module	25
Standards and Certifications	26
Installation of the BMXNOM0200 Module	27

Modbus Serial and Character Mode Communications

General

The serial links support two communication protocols:

- Modbus Serial
- Character Mode

Modbus Protocol

Modbus is a standard protocol with the following properties:

- Establishes client/server communication between different modules within a bus or serial link. The client is identified by the master and the slave modules represent the servers.
- Is based on a mode of data exchange composed of requests and responses offering services via different function codes.
- Establishes a means of exchanging frames from Modbus-type applications in two types of code:
 - RTU mode
 - ASCII mode

The exchange management procedure is as follows:

- Only one device may send data on the bus.
- Exchanges are managed by the master. Only the master may initiate exchanges. Slaves may not send messages without first being invited to do so.
- In the event of an invalid exchange, the master repeats the request. The slave to which the request is made is declared absent by the master if it does not respond within a given time scale.
- If the slave does not understand or cannot process the request, it sends an exception response to the master. In this case, the master may or may not repeat the request.

Two types of dialogue are possible between master and slave(s):

- The master sends a request to a specific slave number and awaits its response.
- The master sends a request to all the slaves without awaiting a reply (the general broadcast principle).

Character Mode Communication

Character mode is a point-to-point mode of data exchange between two entities. Unlike Modbus Protocol, it does not establish hierarchically structured serial link communications or offer services via function codes.

Character Mode is asynchronous. Each item of textual information is sent or received character by character at irregular time intervals. The time taken by the exchanges can be determined from the following properties:

- One or two end-of-frame characters.
- Timeout.
- Number of characters.

Presentation of the BMXNOM0200 Module

General

The BMXNOM0200 serial link module is a 2-way asynchronous serial line module supporting Modbus Serial (master or slave) and Character Mode communications.

The BMXNOM0200 is a simple-format, dedicated module, which can be installed on a Modicon X80 rack.

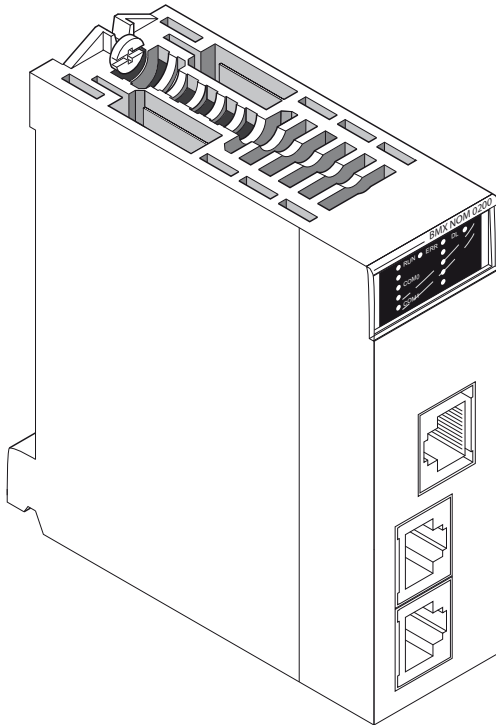
Ruggedized Version

The BMXNOM0200H (hardened) equipment is the ruggedized version of the BMXNOM0200 (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*).

Module introduction

The illustration below shows the physical characteristics of the BMXNOM0200 module:



This BMX NOM 0200 module is composed of the elements in the following table:

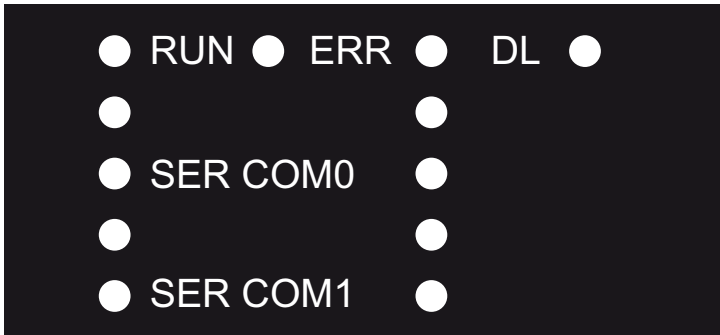
Key	Description
1	Five indicator LEDs on the front of the module: <ul style="list-style-type: none"> ● RUN and ERR show the module's status, ● SER COM0 displays the traffic status on the on the serial port port 0 RS232 or port 0 RS485 (channel 0), ● SER COM1 displays the traffic status on the serial port Port 1 RS485 (channel 1), ● DL shows the firmware download status.
2	Integrated channel 0 dedicated to the serial link with 2 serial ports: <ul style="list-style-type: none"> ● Port 0 RS232 ● Port 0 RS485 Note: Only one serial port can be active at a time.
3	Integrated channel 1 dedicated to the serial link with 1 serial port: <ul style="list-style-type: none"> ● Port 1 RS485

NOTE: In some operating modes, LEDs can indicate more specific information (*see page 20*).

Visual Diagnostics

Five LEDs are located on the front panel of the BMXNOM0200 module. They display information about the module operating status and about the communication status of the built-in serial link.

LED Display:



- RUN = The module is powered and well configured.
- ERR = The module has detected an error and cannot function correctly.
- DL = The firmware is being downloaded.
- SER COM0 = Communication detected on channel 0 (**Port 0 RS232** or **Port 0 RS485**).
- SER COM1 = Communication detected on channel 1 (**Port 1 RS485**).

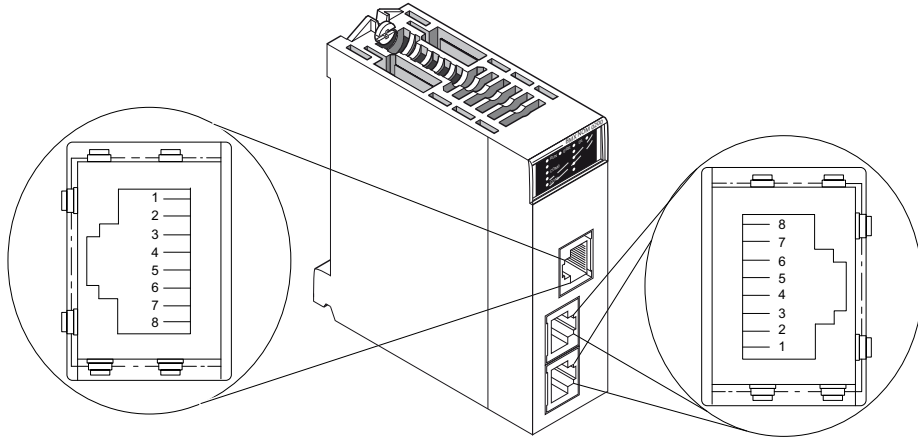
LED meaning:

- Each LED can be in one of these states:
 - 1 = On
 - 0 = Off
 - B = Blinking
- During module startup all LEDs are powered ON and OFF, this verifies that the LEDs are functioning correctly.

RUN	ERR	SER COM0	SER COM1	DL	Diagnose
0	_	_	_	_	The module is not powered or non-operational.
0	B	_	_	_	The module is not configured.
1	1	_	_	_	The module improperly operative.
1	1	1	0	_	The module has detected a problem on the channel 0.
1	1	1	B	_	The module has detected a problem on the channel 0, the channel 1 is exchanging data.
1	1	0	1	_	The module has detected a problem on the channel 1.
1	1	B	1	_	The module has detected a problem on the channel 1, the channel 0 is exchanging data.
1	0	B	_	_	The channel 0 is exchanging data.
1	0	_	B	_	The channel 1 is exchanging data.
B	B	_	_	0	The CPU is absent.
B	B	B	B	_	The module is performing self tests.
_	_	_	_	B	A module firmware is being downloaded.
_	_	_	_	1	The firmware is uploaded; the module must be reset.

Serial Ports Introduction

The illustration below shows the RJ45 serial ports on the BMXNOM0200:



The table below shows the pin assignment for the serial port on the BMXNOM0200:

Pin N°	Serial port RS485	Serial port RS232
1	–	RXD (Receive Data)
2	–	TXD (Transmit Data)
3	–	RTS (Request To Send)
4	D1 (B/B4)	DTR (Data Terminal Ready)
5	D0 (A/A4)	DSR (Data Set Ready)
6	–	CTS (Clear To Send)
7	–	DCD (Data Carrier Detect)
8	Potential serial link grounding (0 V)	Potential serial link grounding (0 V)

NOTE:

- The two RS485 lines are isolated. The isolation voltage between the two serial lines 500 V and between each isolated serial line and the backplane is up to 500V AC.
- The seven-wire RS232 and two-wire RS485 use the same female RJ45 connector. Only the signal cabling is different.

Channels Specifications

The channels of the BMXNOM0200 module include:

- Two isolated RS485 Physical Interfaces,
- One non-isolated RS232 Physical Interface,
- Modbus Serial (ASCII and RTU) and Character Mode communication types.

The link specifications for the two protocols are:

	Modbus Serial / RS485	Modbus Serial / RS232	Character Mode / RS485	Character Mode / RS232
Type	Master/Slave	Master/Slave	Half Duplex	Full Duplex
Flow	19200 bauds. Parameters can be set from 300 bauds to 57600 bauds.	19200 bauds. Parameters can be set from 300 bauds to 115200 bauds.	9600 bauds. Parameters can be set from 300 bauds to 57600 bauds.	9600 bauds. Parameters can be set from 300 bauds to 115200 bauds
Number of devices	32	32	–	–
Authorized slave addresses	1 to 247	1 to 247	–	–
Max. length of Bus without branching	Refer to the table below (15 m with Branching)	15 m	Refer to the table below (15 m with Branching)	15 m
Message Size	Modbus Serial: <ul style="list-style-type: none"> • RTU: 256 bytes (252 bytes of data) • ASCII: 513 bytes (2x252 bytes of data) 	Modbus Serial: <ul style="list-style-type: none"> • RTU: 256 bytes (252 bytes of data) • ASCII: 513 bytes (2x252 bytes of data) 	1024 bytes	1024 bytes
Utilities	Read words/bits. Write words/bits. Diagnostics.	Read words/bits. Write words/bits. Diagnostics.	Send character strings. Receive character strings.	Send character strings. Receive character strings.
Hardware Flow Control	–	Optionally via RTS/CTS signals.	–	Optionally via RTS/CTS signals.

NOTE: By simultaneously using several communication function blocks per channel, the BMXNOM0200 can consume all of the Modbus bandwidth.

The table below shows the maximum RS485 cable length that can be used, according to the baud rate chosen:

Baud Rate choice (bit/s)	Length (m)	Product reference
300	1000	(1)
600	1000	(1)
1200	1000	(1)
2400	1000	(1)
9600	1000	(1)
19200	600	(1)
38400	300	(1) or (2)
57600	200	(1) or (2)

- (1): Cable shielded twisted pair AWG24 gauge (TSX CSA 100, TSX CSA 200, TSX CSA 500)
- (2): Cable category 5 or higher

Altitude Operating Conditions

The characteristics in the table below apply to the modules BMXNOM0200 and BMXNOM0200H 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

Module	Temperature range
BMXNOM0200	0...60 °C (32...140 °F)
BMXNOM0200H	-25...70 °C (-13...158 °F)

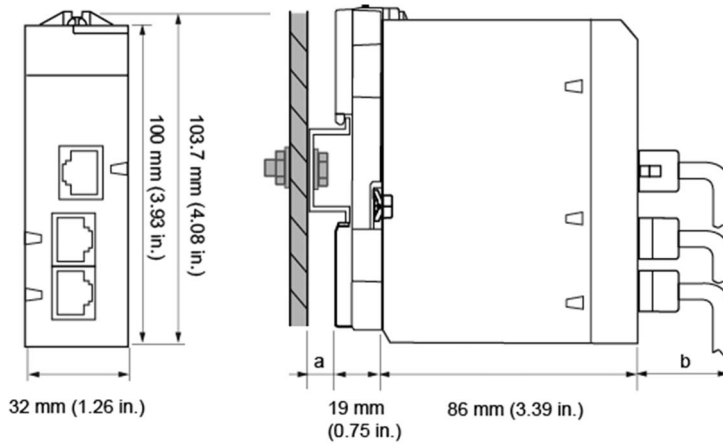
Module Consumption

This table shows the consumption of BMXNOM0200 module:

Voltage	Typical Current	Maximum Current	Typical Power Dissipation	Maximum Power Dissipation
24 VDC	80 mA	130 mA	1.92 W	3.12 W

Dimensions of X80 BMXNOM0200(H) Serial Link Module

General Presentation of X80 BMXNOM0200(H) Serial Link Module



- a** DIN-rail depth: the value depends on the DIN-rail type used in your platform.
b Wiring depth: the value depends on the connector and the wires used in your platform.

Dimensions of X80 BMXNOM0200(H) Serial Link Module

Module reference	Module dimensions			Installation depth ⁽¹⁾
	Width	Height	Depth	
BMXNOM0200(H)	32 mm (1.26 in.)	103.7 mm (4.08 in.)	86 mm (3.39 in.)	105 mm (4.13 in.) ⁽¹⁾
(1) DIN-rail depth (a) and wiring depth (b) are not included.				

NOTE: Consider the connector dimensions, clearance for cable installation, and spacing around the racks.

Standards and Certifications

Download

Click the link that corresponds to your preferred language to download standards and certifications (PDF format) that apply to the modules in this product line:

Title	Languages
Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications	<ul style="list-style-type: none"><li data-bbox="650 386 920 410">● English: EIO0000002726<li data-bbox="650 414 916 438">● French: EIO0000002727<li data-bbox="650 441 927 466">● German: EIO0000002728<li data-bbox="650 469 906 493">● Italian: EIO0000002730<li data-bbox="650 496 927 521">● Spanish: EIO0000002729<li data-bbox="650 524 927 548">● Chinese: EIO0000002731

Installation of the BMXNOM0200 Module

General

The BMXNOM0200 module is installed in a Modicon X80 rack in any open module slot, except the ones required for the power supply, the processor, the drop end communicator or the rack extender module. This installation must conform to the rack installation instructions.

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 products.

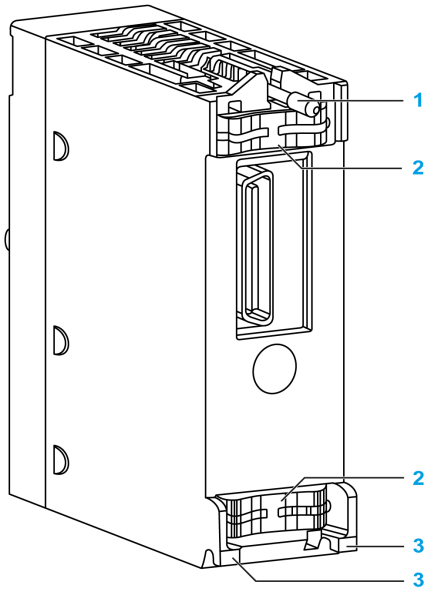
Follow all local and national safety codes and standards.

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

NOTE: The BMXNOM0200 module can be installed in a rack while the application is running on the PLC.

Module Grounding

The BMXNOM0200 module is equipped with contact strips at the rear for grounding purposes:



- 1 Mounting screw
- 2 Contact strips
- 3 Locating pins

When the module is correctly installed on the rack, the contact strips connect the grounding bus of the module to the grounding bus of the rack.

⚡ ⚠ DANGER

HAZARD OF ELECTRIC SHOCK

Check that ground contact strips are available and not bent out of shape.

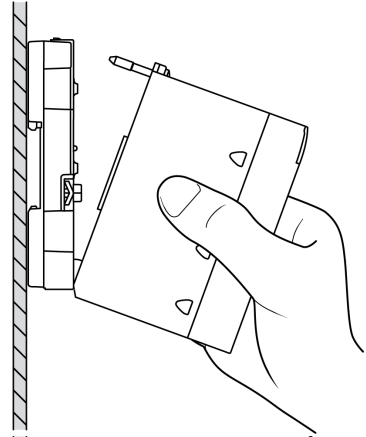
Failure to follow these instructions will result in death or serious injury.

NOTE: If ground contacts are bent or not available, do not use the module and contact your Schneider Electric representative.

Module Installation

Install a BMXNOM0200 module in a rack:

Step	Action
1	Remove the protective cover from the connector of the module slot on the Modicon X80 rack.
2	Position the locating pins situated at the rear of the module (on the bottom part) in the corresponding slot in the rack.
3	Swivel the module towards the top of the rack so that the module sits flush with the rack.
4	Tighten the mounting screw on top of the module to hold in place on the rack. Tightening torque: 0.4...1.5 N•m (0.30...1.10 lbf-ft).



⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Check that the mounting screw is securely tightened to ensure the module is firmly attached to the rack.

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

An RJ45 connector can then be connected to the module according to the targeted network.

Connection/ Disconnection

The BMXNOM0200 module can be connected or disconnected while the power is on.

WARNING

UNINTENDED EQUIPMENT OPERATION

Although you can connect or disconnect the wires on the BMXNOM0200 module while the power to the station is on, doing so can interrupt the application in progress.

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

When the module is disconnected from the rack, its internal memory is erased. The module goes through an initialization phase once it is reconnected to the backplane.

The following situations can create a temporary disruption in the application or communications:

- The RJ45 connector is connected or disconnected when the power is on.
- Modules are re-initialized when the power is switched back on.

Use Case: Extra point of Connection

A BMXNOM0200 (with firmware version ≥ 1.2) module can be inserted into a rack at any free slot without have been configured. This is very useful to connect Control Expert while the CPU is not configured or as an extra point of connection. In this case the BMXNOM0200 is in default configuration.

The BMXNOM0200 default configuration is MODBUS slave at address 248, RTU (delay between frames = 2 ms), 8 bits of data, 1 stop bit, even parity, RS232 at 115200 bit/s on channel 0 and RS485 at 57600 bit/s on channel 1.

The address 248 is the point-to-point address to which any BMXNOM0200 slave module answers. This functionality aims at connecting directly to any slave module whose address is unknown.

Firmware Update

The firmware of the BMXNOM0200 can be updated via the PLC backplane using one of the following tools:

- EcoStruxure™ Automation Device Maintenance
- Unity Loader

Chapter 2

Serial Communication Architectures

Subject of this Chapter

This chapter provides an introduction to architectures that use serial communication, as well as cabling requirements.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Modbus Line Termination and Polarization (RS485)	32
Connecting Modbus Devices (RS485)	34
Connecting Data Terminal Equipment (DTE) (RS232)	36
Connecting Data Circuit-terminating Equipment (DCE) (RS232)	38
Cabling	40

Modbus Line Termination and Polarization (RS485)

Overview

A multi-point Modbus network must have line termination and polarization.

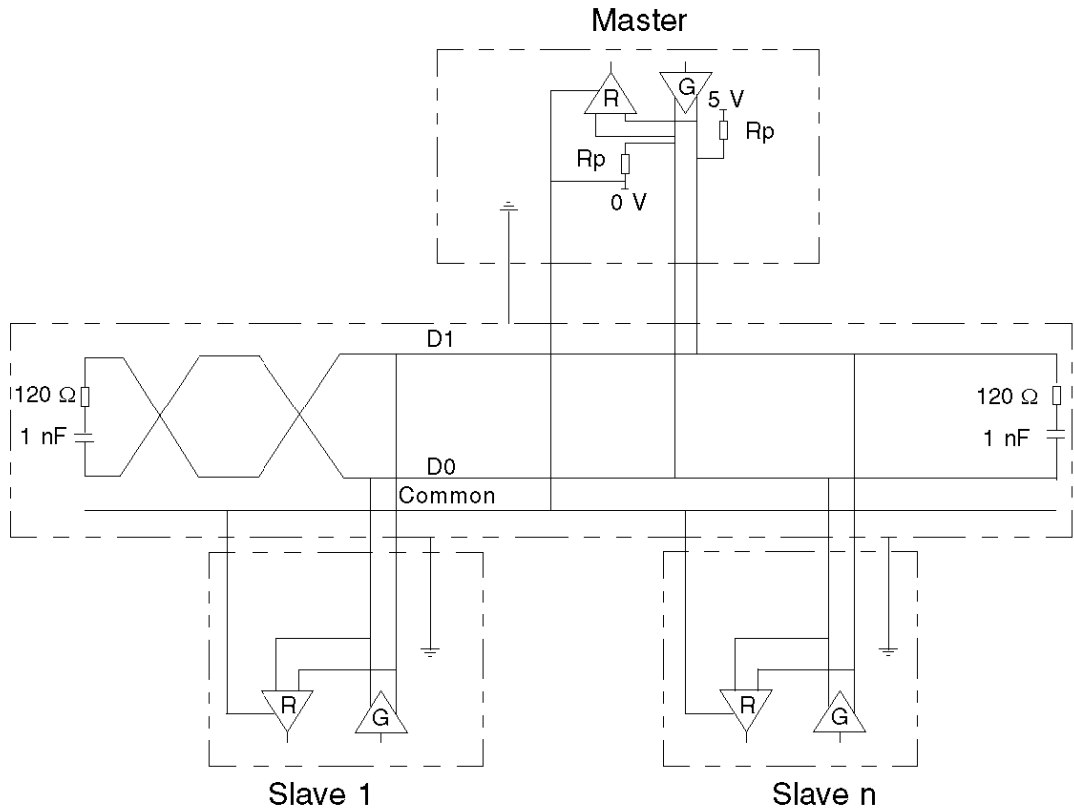
Equipments connectable to this bus are:

- Other PLCs like M340, Premium, Quantum, Twido or Nano
- Schneider Automation devices like Altivar, Security module XPS, SEPAM, XBT or Momentum
- Other Modbus protocol compliant devices
- Modem, Hub

An example of **multi-point Modbus network** (*see page 35*) including a BMXNOM0200 module is presented in this manual.

NOTE: A point to point Modbus network can also be performed.

Electrical schema of line termination and polarization:



Line Termination

Line termination is made externally: it consists of two 120 Ω resistors and 1 nF capacitor placed at each end of the network (VW3 A8 306RC or VW3 A8 306 DRC). Don't place line termination at the end of a derivation cable.

Line Polarization

On a Modbus line, polarization is needed for an RS485 network.

- If the BMXNOM0200 module is used as a master, it is automatically driven by the system so there is no need of external polarization.
- If the BMXNOM0200 module is used as a slave, the polarization must be implemented by two 450 to 650 Ω resistors (R_p) connected on the RS485 balanced pair:
 - a pull-up resistor to a 5 V voltage on the D1 circuit,
 - a pull-down resistor to the common circuit on D0 circuit.

NOTE:

In character mode, the line polarization is configurable under Control Expert. It is possible to choose between:

- low impedance polarization like in Modbus networks (the goal of this kind of polarization is to let the master maintain the default state),
- high polarization impedance (the goal of this kind of polarization is to let each device contribute to maintain the default state),
- no polarization (if an external polarization is used).

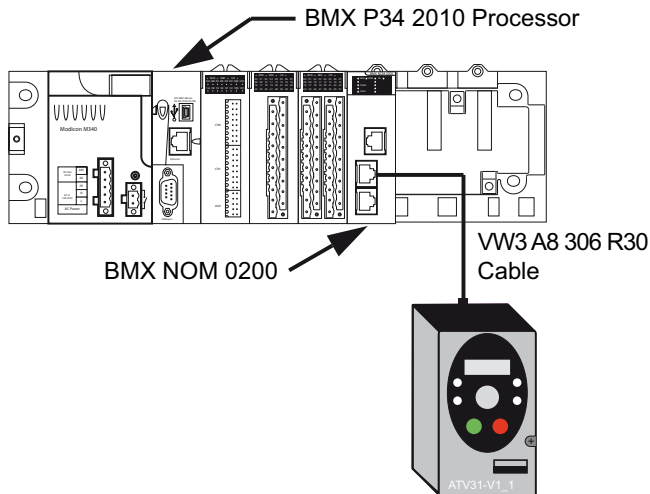
Connecting Modbus Devices (RS485)

General

The following pages present an example of Modbus device connection and a Modbus serial link architecture.

Connecting Modbus devices that are not powered via the Serial Link

The figure below shows a BMXNOM0200 module connected to an ATV31drive:



The devices are configured as follows:

- A BMXP342010 processor,
- A BMXNOM0200 module configured as master,
- An ATV31 drive configured as slave.

The VW3A8306R30 cable has the following properties:

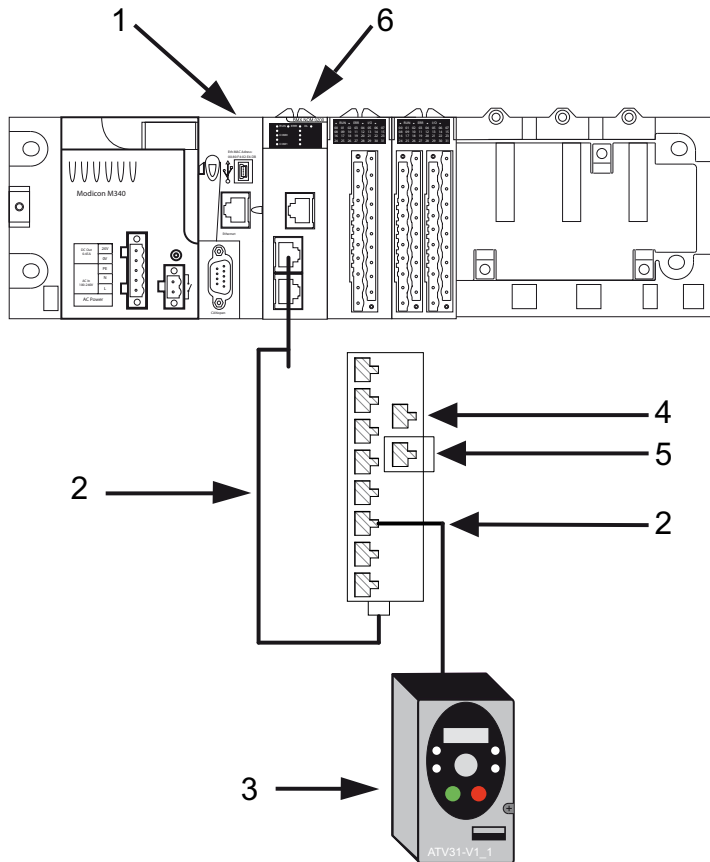
- Connection: 2 male RJ45 connectors
- Wiring: 2 wires for the RS485 physical line

Modbus Serial Link Architecture

The Modbus serial link architecture consists of the following elements:

- A BMXP342010 processor,
- A BMXNOM0200 module configured as master,
- An LU9GC3 splitter block,
- Two ATV31 drives configured as slaves.

The illustration below represents the serial link architecture described above:



- 1 BMXP342010 processor
- 2 VW3A8306R** cable
- 3 ATV31 drive
- 4 LU9GC3 splitter block
- 5 VW3A8306RC Modbus Line Terminator
- 6 BMXNOM0200 module

Connecting Data Terminal Equipment (DTE) (RS232)

General

Data terminal equipment is the term used to describe devices such as:

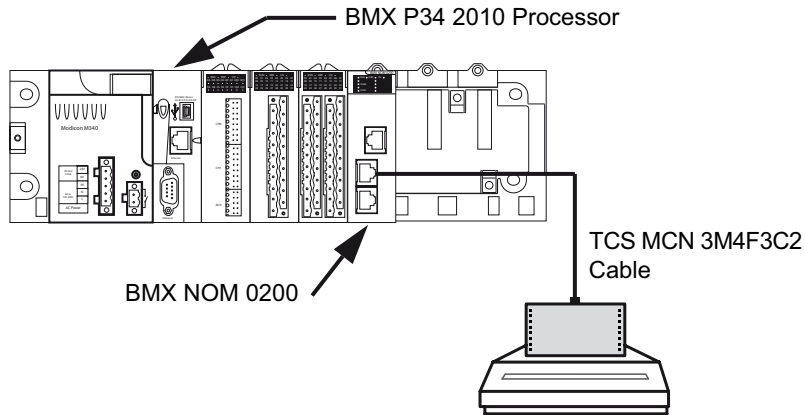
- Common peripherals (printer, keyboard-screen, workshop terminal, etc.)
- Specialized peripherals (barcode readers, etc.)
- PCs

For a DTE type device, the RTS and CTS pins are crossed.

All data terminal equipment is connected to a BMXNOM0200 module by a serial cross cable using the RS232 physical link.

Connecting Data Terminal Equipment

The figure below shows a printer connected to a BMXNOM0200 module:



The communication protocol used is Character Mode.

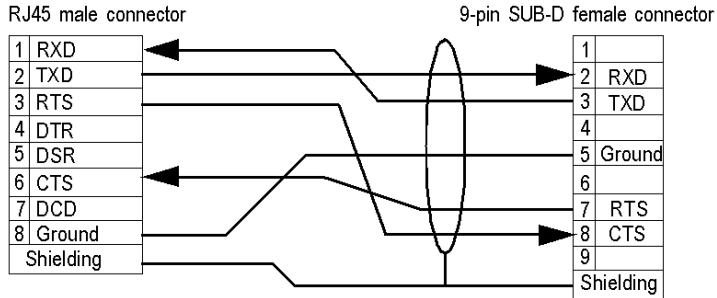
NOTE: Only one item of data terminal equipment may be connected to the BMXNOM0200 module.

RS 232 Serial Cross Cable

The TCSMCN3M4F3C2 serial cross cable has two connectors:

- RJ45 male,
- 9-pin SUB-D female.

The figure below shows the pin assignment for a TCSMCN3M4F3C2 serial cross cable:



Connecting Cables and Accessories

The table below shows the product references of the cables and adapters to be used according to the serial connector used by the data terminal equipment:

Serial Connector for Data Terminal Equipment	Wiring
9-pin SUB-D male connector	TCSMCN3M4F3C2 cable
25-pin SUB-D male connector	<ul style="list-style-type: none"> ● TCSMCN3M4F3C2 cable ● TSXCTC07 adapter
25-pin SUB-D female connector	<ul style="list-style-type: none"> ● TCSMCN3M4F3C2 cable ● TSXCTC10 adapter

Connecting Data Circuit-terminating Equipment (DCE) (RS232)

General

Data Circuit-terminating Equipment (DCE) is the term used to describe devices such as modems. For a DCE type device, the RTS and CTS pins are connected directly (not crossed).

All data circuit-terminating equipments are connected to a BMXNOM0200 module by a serial direct cable using an RS232 physical link.

NOTE: The differences between DCE and DTE connections are largely in the plugs and the signal direction of the pins (input or output). For example, a desktop PC is termed as a DTE device while a modem is termed as a DCE device.

Modem Characteristics

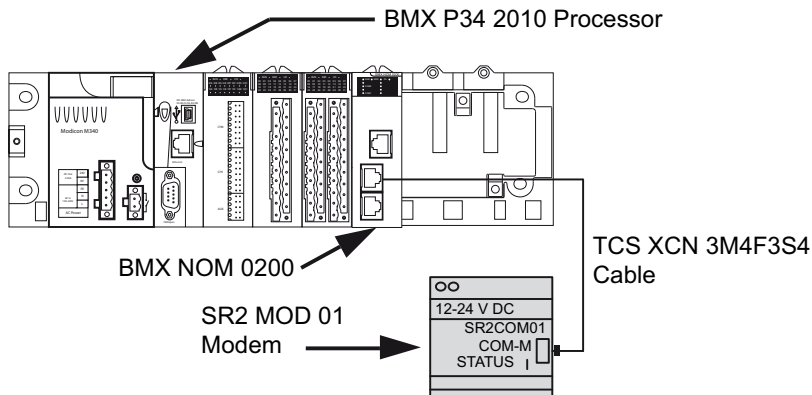
The BMXNOM0200 module works with most modems on the market. To connect a modem to the serial port of a BMXNOM0200 module, the modem must have the following characteristics:

- Support 10 or 11 bits per character if the terminal port is used in Modbus Serial:
 - 7 or 8 data bits
 - 1 or 2 stop bits
 - Odd, even or no parity
- Operate without a data carrier check.

CTS, DTR, DSR and DCD signals can be managed by the application.

Connecting Data Circuit-terminating Equipment

The figure below shows a modem connected to a BMXNOM0200 module:



The modem connection needs specific modem cable to work.

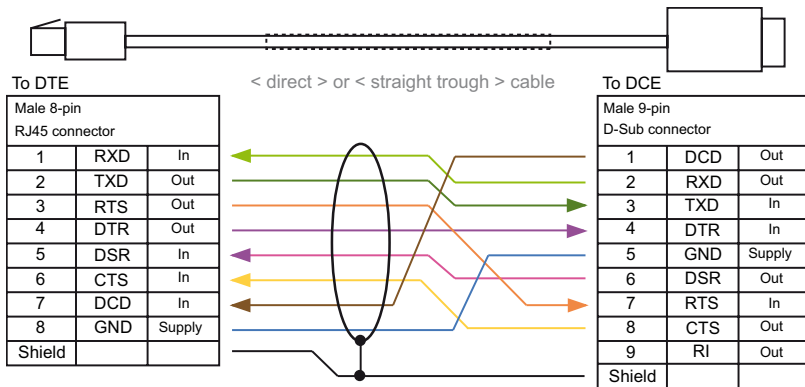
RS 232 Serial Direct Cable

Example of the TCSXCN3M4F3S4 Cable:

The TCSXCN3M4F3S4 serial direct cable is an 8 wires version and has two connectors:

- RJ45 male,
- 9-pin SUB-D male.

The illustration below shows the pin assignment for a TCSXCN3M4F3S4 serial direct cable:



Connecting Cables and Accessories

The table below shows the product references of the cables and adapters to be used according to the serial connector used by the data circuit-terminating equipment:

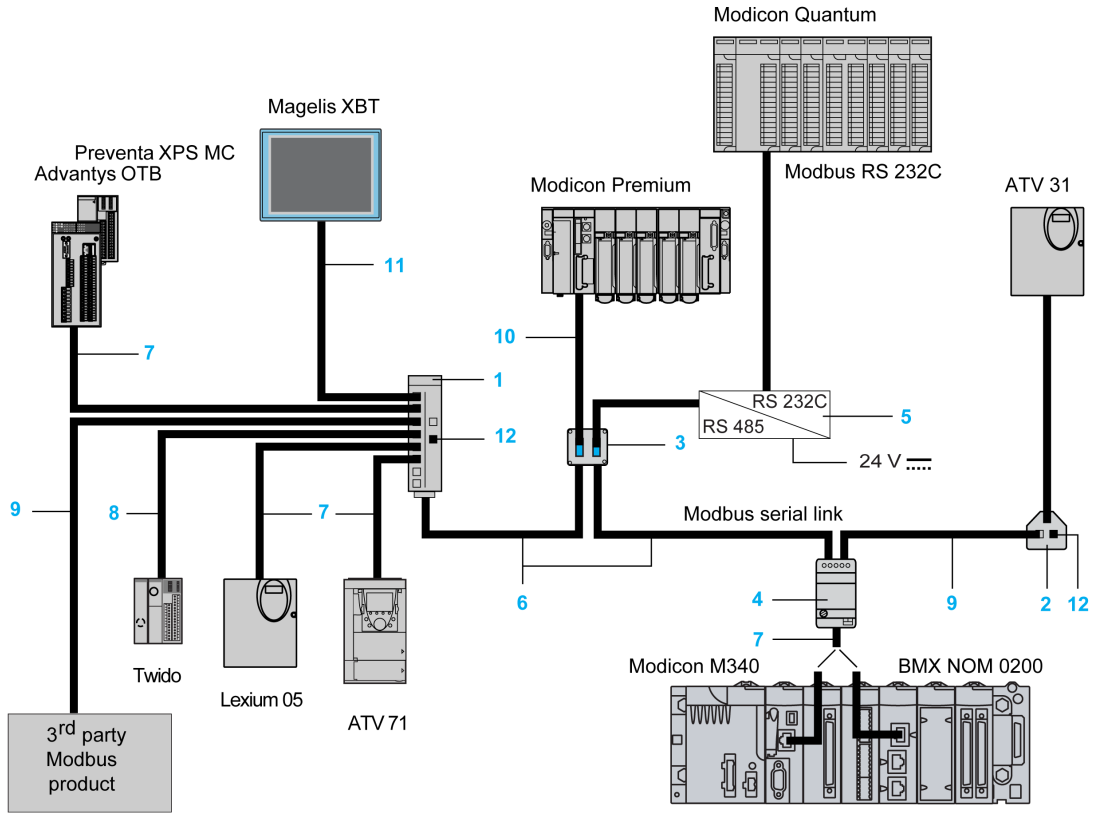
Serial Connector for Data Circuit-terminating Equipment	Wiring
9-pin SUB-D female connector	<ul style="list-style-type: none"> ● TCSMCN3M4M3S2 cable ● TCSXCN3M4F3S4 cable
25-pin SUB-D female connector	<ul style="list-style-type: none"> ● TCSMCN3M4M3S2 cable ● TSXCTC09 Adapter

Cabling

Cabling System

Several cables and accessories are required in order to set up a serial link.

The figure below shows an example of Modbus serial link and character mode cabling system. The **cables** (see page 41) and **connecting accessories** (see page 42) referenced in the figure are described in the next tables:



Cables

The table below shows the available cables that are compatible with serial communication on these processors and module:

Figure Reference	Designation	Characteristics	Length	Product reference
6	RS485 double shielded twisted pair trunk cable	Two bare ends	100 m	TSX CSA 100
			200 m	TSX CSA 200
			500 m	TSX CSA 500
7	Modbus RS485 cable	Two RJ45 male connectors	0.3 m	VW3 A8 306 R03
			1 m	VW3 A8 306 R10
			3 m	VW3 A8 306 R30
-	Modbus RS485 cable	<ul style="list-style-type: none"> ● One RJ45 male connector ● One fifteen-pin SUB-D male connector 	3 m	VW3 A8 306
8	Modbus RS485 cable	<ul style="list-style-type: none"> ● One RJ45 male connector ● One mini-DIN connector 	0.3 m	TWD XCA RJ003
			1 m	TWD XCA RJ010
			3 m	TWD XCA RJ030
9	Modbus RS485 cable	<ul style="list-style-type: none"> ● One RJ45 male connector ● One bare end 	3 m	VW3 A8 306 D30
10	Modbus RS485 cable	<ul style="list-style-type: none"> ● One miniature connector ● One 15-pin SUB-D connector 	3 m	TSX SCP CM 4630
11	RS485 cable for Magelis XBT display and terminal	<ul style="list-style-type: none"> ● One RJ45 male connector ● One 25-pin SUB-D female connector <p>Note: This cable is not compatible with BMX NOM 0200 module</p>	2.5 m	XBT-Z938
-	RS485 cable for devices that are powered via the serial link	Two RJ45 male connectors Note: This cable is not compatible with BMX NOM 0200 module.	3 m	XBT-Z9980
-	Four-wire RS232 cable for Data Terminal Equipment (DTE)	<ul style="list-style-type: none"> ● One RJ45 male connector ● One nine-pin SUB-D female connector 	3 m	TCS MCN 3M4F3C2
-	Four-wire RS232 cable for Data Circuit-terminating Equipment (DCE)	<ul style="list-style-type: none"> ● One RJ45 male connector ● One nine-pin SUB-D male connector 	3 m	TCS MCN 3M4M3S2
-	Seven-wire RS232 cable for Data Circuit-terminating Equipment (DCE)	<ul style="list-style-type: none"> ● One RJ45 male connector ● One 9-pin SUB-D male connector 	3 m	TCS XCN 3M4F3S4

Connecting Accessories

The table below shows the available connecting accessories that are compatible with serial communication on these processors and module:

Figure Reference	Designation	Characteristics	Product reference
1	Modbus splitter box	<ul style="list-style-type: none"> • Ten RJ45 connectors • One screw terminal block 	LU9 GC3
2	T-junction box	<ul style="list-style-type: none"> • Two RJ45 connectors • On-board 0.3 m cable with RJ45 connector at end 	VW3 A8 306 TF03
		<ul style="list-style-type: none"> • Two RJ45 connectors • On-board 1 m cable with RJ45 connector at end 	VW3 A8 306 TF10
-	Passive T-junction box	<ul style="list-style-type: none"> • Three screw terminal blocks • RC line end adapter 	TSX SCA 50
3	Passive 2-channel subscriber socket	<ul style="list-style-type: none"> • Two fifteen-pin SUB-D female connectors • Two screw terminal blocks • RC line end adapter 	TSX SCA 62
4	Isolated RS485 T-junction box	<ul style="list-style-type: none"> • One RJ45 connectors • One screw terminal block 	TWD XCA ISO
-	T-junction box	Three RJ45 connectors	TWD XCA T3RJ
-	Modbus / Bluetooth adapter	<ul style="list-style-type: none"> • One Bluetooth adapter with one RJ45 connector • One cordset for PowerSuite with two RJ45 connectors • One cordset for TwidoSuite with one RJ45 connector and one mini-DIN connector • One RJ45/SUB-D male 9-pin adapter for ATV speed drives 	VW3 A8 114
5	RS232C/RS485 line adapter without modem signals	19.2 kbit/s	XGS Z24
12	Line terminator for RJ45 connector	<ul style="list-style-type: none"> • Resistance of 120 Ω • Capacity of 1 nF 	VW3 A8 306 RC
-	Line terminator for screw terminal block	<ul style="list-style-type: none"> • Resistance of 120 Ω • Capacity of 1 nF 	VW3 A8 306 DRC
-	Adapter for non-standard devices	<ul style="list-style-type: none"> • Two 25-pin SUB-D male connectors 	XBT ZG999
-	Adapter for non-standard devices	<ul style="list-style-type: none"> • One 25-pin SUB-D male connector • One 9-pin SUB-D male connector 	XBT ZG909

Figure Reference	Designation	Characteristics	Product reference
-	Adapter for data terminal equipment	<ul style="list-style-type: none">● One 9-pin SUB-D male connector● One 25-pin SUB-D female connector	TSX CTC 07
-	Adapter for data terminal equipment	<ul style="list-style-type: none">● One 9-pin SUB-D male connector● One 25-pin SUB-D male connector	TSX CTC 10
-	Adapter for Data Circuit-terminating Equipment (DCE)	<ul style="list-style-type: none">● One 9-pin SUB-D female connector● One 25-pin SUB-D male connector	TSX CTC 09

NOTE: This list of cables and accessories is not exhaustive.

Part II

Software Implementation of Modbus Serial and Character Mode Communications

In This Part

This part provides an introduction to the software implementation of Modbus Serial and Character Mode communications using Control Expert software.

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
3	BMXNOM0200 Limitation and Implementation Rules	47
4	Modbus Serial Communication	53
5	Character Mode Communication	87
6	BMXNOM0200 Module Diagnostics	109
7	Language Objects of Modbus and Character Mode Communications	115
8	Dynamic Protocol Switching	151

Chapter 3

BMXNOM0200 Limitation and Implementation Rules

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
BMXNOM0200 Limitations Rules	48
BMXNOM0200 Implementation Rules	50

BMXNOM0200 Limitations Rules

Overview

The number of the BMXNOM0200 modules in a hardware configuration is linked to the:

- Platform (M340, M580, and Quantum)
- Module installation (in local racks or in X80 drop).
- Channel configuration (master or slave).

NOTE: Each BMXNOM0200 configured channel is considered an expert channel when you calculate the maximum number of expert channels in a configuration.

When the application is built, Control Expert checks that the limitation is not exceeded.

M340 Platform

NOTE: When installed with an M340 PLC, the BMXNOM0200 requires a CPU with minimum OS version 02.10

The maximum number of BMXNOM0200 modules that can be configured in an M340 PLC station is linked to:

- The Modicon M340 PLC capabilities (*see Modicon M340, Processors, Setup Manual*).
- The number of expert channels already configured.
- The number of channel configured on each BMXNOM0200 modules.

M580 Platform

In an M580 system, the maximum number of BMXNOM0200 modules that can be configured is given by the respective limitations for local racks and X80 drops (remote racks).

Local racks: The maximum number of BMXNOM0200 modules that can be configured in M580 local racks (that is local and extended local racks) is linked to:

- The maximum number of expert channels allowed in local configuration (*see Modicon M580, Hardware, Reference Manual*).
- The number of expert channels already configured.
- The number of channel configured on each BMXNOM0200 modules.

X80 drop: The maximum number of BMXNOM0200 modules that can be configured in each X80 drop (with an X80 performance EIO adapter module BMXCRA31210 or BMECRA31210) follow the following rules:

- a maximum of 36 expert channels.
- a maximum of six BMXNOM0200 modules configured.

NOTE: With M580 CPU OS version \leq V2.40, the maximum is limited to four BMXNOM0200 modules configured.

NOTE: In an M580 Hot Standby system, the BMXNOM0200 module can only be configured in X80 drops (main or extended remote racks).

Quantum Platform

In a Quantum system, a maximum of **16** BMXNOM0200 modules can be configured including the following limitation for X80 drop:

X80 drop: The maximum number of BMXNOM0200 modules that can be configured in each X80 drop (with an X80 performance EIO adapter module BMXCRA31210 or BMECRA31210) follow the following rules:

- a maximum of 36 expert channels.
- a maximum of four BMXNOM0200 modules configured.
- a maximum of four channels configured as master.

For example, the maximum configuration in an X80 drop can be reached with two BMXNOM0200 modules when both channels of each module are configured as master.

BMXNOM0200 Implementation Rules

Overview

The accessibility of the BMXNOM0200 module functions is linked to the:

- Platform (M340, M580, and Quantum)
- Module installation (in local racks or in X80 drop)
- Module firmware version.

The following tables give by platform the availability and restrictions of the BMXNOM0200 module functionalities. The tables give also the Control Expert **Hardware Catalog** devices to configure.

M340 Platform

Module features and requirements			M340 local rack	
BMXNOM0200 Communication protocols	Modbus	Master	Yes	Yes ⁽¹⁾
		Slave	Yes	Yes ⁽¹⁾
	Character Mode		Yes	Yes ⁽¹⁾
BMXNOM0200 Requirements	Firmware version		V1.0	Minimum V1.2
	Control Expert Hardware Catalog device		BMXNOM0200	BMXNOM0200.2 (SV>=1.2)
(1) Expert mode used to configure the time-out links individually from the application and thus adapt to the specific characteristic of certain modems. On RS-232 physical line, the configuration of hardware flow management signals, allows selecting between DTE and DCE modes.				

M580 Platform

The BMXNOM0200 module can be installed and configured in the M580 local racks as well as Modicon X80 drop with an X80 performance EIO adapter module (BMXCRA31210 or BMECRA31210).

NOTE: In an M580 Hot Standby system, the BMXNOM0200 module can only be installed and configured in an Modicon X80 drop (main or extended remote rack).

Module features and requirements			M580 local racks (main and extended)	X80 drop over M580 PAC (main and extended remote racks)		
BMXNOM0200 Communication protocols	Modbus	Master	Yes	Yes	Yes	
		Slave	Yes	No	Yes ⁽¹⁾	
	Character Mode		Yes	Yes	Yes	
Requirements	BMXNOM0200	Firmware version	Minimum V1.2	Minimum V1.4	Minimum V1.5	
		Control Expert Hardware Catalog device	BMXNOM0200.2 (SV>=1.2)	BMXNOM0200.3 (SV>=1.4)	BMXNOM0200.4 (SV>=1.5)	
	BMXCRA31210 Drop end communicator	Firmware version	–	Minimum V2.08	Minimum V2.14	
		Control Expert Hardware Catalog device	–	All ⁽²⁾	BMXCRA31210 (SV>=2.10)	
	Or					
	BMECRA31210 Drop end communicator	Firmware version	–	Minimum V2.00	Minimum V2.14	
Control Expert Hardware Catalog device		–	All ⁽²⁾	BMECRA31210 (SV>=2.10)		
(1) Modbus slave is only accessible and configurable on physical line RS-485 and RTU mode. (2) Select the Control Expert Hardware Catalog device according to the firmware version of the drop end communicator. – Not applicable						

Quantum Platform

The BMXNOM0200 module can only be installed and configured in an EIO Modicon X80 drop with an X80 performance EIO adapter module (BMXCRA31210 or BMECRA31210).

NOTE: Configuring the BMXNOM0200 module as a Modbus slave is only possible with a BMXCRA31210 and requires to interlink a Quantum 140NOC78•00 to the Quantum 140CRP31200. For more information, refer to the chapter Quick Start: BMXNOM0200 as a Modbus Slave over a Quantum PLC (*see page 155*).

Module features and requirements			X80 drop over Quantum PLC (main and extended remote racks)		
BMXNOM0200 Communication protocols	Modbus	Master	Yes	Yes	
		Slave	No	Yes ⁽¹⁾	
	Character Mode		Yes	Yes	
Requirements	BMXNOM0200	Firmware version	Minimum V1.4	Minimum V1.5	
		Control Expert Hardware Catalog device	BMXNOM0200.3 (SV>=1.4)	BMXNOM0200.4 (SV>=1.5)	
	BMXCRA31210 Drop end communicator	Firmware version	All	Minimum V2.14	
		Control Expert Hardware Catalog device	All ⁽²⁾	BMXCRA31210 (SV>=2.13)	
	Or				
	BMECRA31210 Drop end communicator	Firmware version		All	–
Control Expert Hardware Catalog device		All ⁽²⁾	–		
(1) Modbus slave is only accessible and configurable on physical line RS-485 and RTU mode. (2) Select the Control Expert Hardware Catalog device according to the firmware version of the drop end communicator. – Not applicable					

Chapter 4

Modbus Serial Communication

Subject of this Chapter

This chapter presents the software implementation process for Modbus serial communication for BMXNOM0200.

What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
4.1	Generalities	54
4.2	Modbus Serial Communication Configuration	61
4.3	Modbus Serial Communication Programming	73
4.4	Debugging Modbus Serial Communication	83

Section 4.1

Generalities

Subject of this Section

This section presents the general points relating to Modbus serial communication and its services.

What Is in This Section?

This section contains the following topics:

Topic	Page
About Modbus Serial	55
Performance	56
How to Access the Serial Link Parameters	58

About Modbus Serial

Introduction

Communicating via Modbus enables data exchange between all devices connected to the bus. The Modbus Serial is a protocol that creates a hierarchical structure (one master and several slaves).

The master manages all exchanges in two ways:

- The master exchanges with the slave and awaits a response.
- The master exchanges with all the slaves without waiting for a response (general broadcast).

NOTE: Be careful that two masters (on the same bus) do not send requests simultaneously otherwise the requests are lost and each report will have a bad result which could be 16#0100 (request could not be processed) or 16#ODFF (slave is not present).

 WARNING
CRITICAL DATA LOSS Only use communication ports for non-critical data transfers. Failure to follow these instructions can result in death, serious injury, or equipment damage.

Performance

At a Glance

The tables that follow can be used to evaluate typical Modbus communication exchange times according to different criteria.

The results displayed correspond to the average operation period for the `READ_VAR` function in milliseconds.

Exchange Time Definition

The Exchange Time is the time between the creation of an exchange and the end of that exchange. It includes the serial link communication time.

The exchange is created when the communication function call is made.

The exchange ends when one of the following events occurs:

- Data is received.
- An anomaly occurs.
- Time-out expires.

Exchange Time for One Word

The table below shows exchange times for one word of Modbus communication on a BMX NOM 0200 module (the Modbus slave is a BMX P34 1000 cyclic):

Exchange time ⁽¹⁾ in ms		Cycle time in ms		
		Cyclic	10	50
Baud rate of communication in bits per second	4800	65	68	100
	9600	38	47	50
	19200	29	38	50
	38400	24	30	50
	57600	17	20	50
	115200	17	20	50

(1) All exchange times listed above come from measures with an accuracy margin of +/-10 ms

Exchange Time for 100 Words

The table below shows exchange times for 100 words of Modbus communication on a BMX NOM 0200 module (the Modbus slave is a BMX P34 1000 cyclic):

Exchange time ⁽¹⁾ in ms		Cycle time in ms		
		Cyclic	10	50
Baud rate of communication in bits per second	4800	560	560	600
	9600	286	295	300
	19200	152	160	200
	38400	86	90	100
	57600	56	60	100
	115200	36	40	50

(1) All exchange times listed above come from measures with an accuracy margin of +/-10 ms

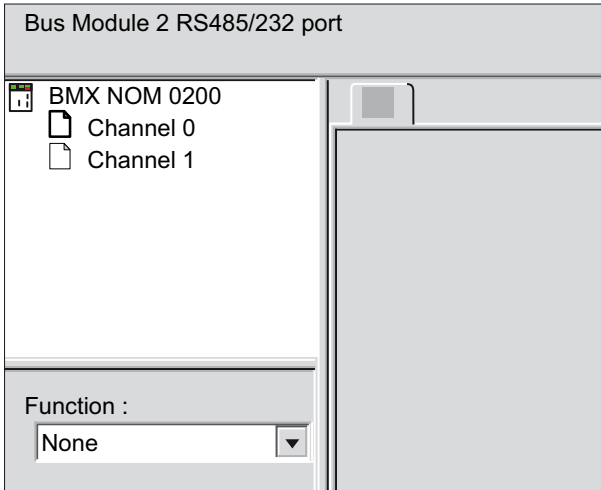
How to Access the Serial Link Parameters

At a Glance

The following pages explain how to access the serial ports configuration screen for the BMXNOM0200 module as well as the general elements of the Modbus and Character Mode link configuration and debug screens.

How to Access the Serial Link

The table below describes the procedure for accessing the serial link of a BMXNOM0200 module:

Step	Action
1	Open the hardware configuration editor.
2	Double-click on the BMXNOM0200 module.
3	Select the channel to configure (Channel 0 or Channel 1). Result with Channel 0 selected: 

Step	Action
4	<p>Select the Modbus link function.</p> <p>Result with Channel 0 selected:</p>

Description of the Configuration Screen

The following table shows the different elements of the configuration screens:

Key	Element	Function
1	Tabs	<p>The tab in the foreground indicates the mode currently in use (Configuration in this example). Each mode can be selected using the corresponding tab. The available modes are:</p> <ul style="list-style-type: none"> ● Configuration ● Debug (accessible in online mode only) ● Diagnostic (accessible in online mode only)
2	Module Zone	Displays module reference and module LEDs status in online mode.

Key	Element	Function
3	Channel zone	<p>Enables you to:</p> <ul style="list-style-type: none"> ● Display the following tabs by clicking on BMX NOM 0200: <ul style="list-style-type: none"> ○ Overview, which gives the characteristics of the device. ○ I/O Objects, which is used to presymbolize the input/output objects. ○ Fault, which shows the detected device faults (in online mode). ● Display the following tabs by clicking on Channel 0 or Channel 1: <ul style="list-style-type: none"> ○ Configuration ○ Debugging ○ Fault ● Display the channel name and symbol defined by the user (using the variables editor).
4	General parameters zone	<p>This enables you to choose the general parameters associated with the channel:</p> <ul style="list-style-type: none"> ● Function: The available functions are None, Modbus link and Character mode link. By default, the function is configured as None. ● Task: Defines the master task in which the implicit exchange objects of the channel will be exchanged. This zone is grayed out and cannot be configured.
5	Configuration, debugging or fault zone	<p>In configuration mode, this zone is used to configure the channel parameters. In debug mode, it is used to debug the communication channel. In diagnostic mode, it is used to display current detected errors either at module or at channel level.</p>

Section 4.2

Modbus Serial Communication Configuration

Subject of this Section

This section describes the software configuration process for Modbus serial communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
Modbus Serial Communication Configuration Screen	62
Application-linked Modbus Parameters	64
Signal and Physical Line Parameters in Modbus	66
Transmission-linked Modbus Parameters	69
How to Set the BMXNOM0200 MODBUS Slave Address Without Control Expert?	71

Modbus Serial Communication Configuration Screen

General

The following pages provide an introduction to the configuration screen for Modbus serial communication.

Access to the Configuration Screen

The following table describes the procedure for accessing the configuration screen for Modbus serial communication:

Step	Action
1	Open the BMX NOM 0200 sub-directory in the project browser (<i>see page 58</i>).
2	Select the Channel to configure and "Modbus link" function on the screen that appears.

Illustration

The figure below shows the default configuration screen for Modbus serial communication on Channel 0:

The screenshot shows the 'Configuration' window for Channel 0. The 'Type' is set to 'Slave'. Under 'Mastrer', 'Number of retries' is 0 and 'Answer delay' is 1 X 10 ms. Under 'Slave', 'Slave number' is 1. The 'Physical line' is set to 'RS485' and 'Signals' is set to 'RX/TX'. Under 'Transmission speed', it is 19200 bits/s. 'Delay between frames' is checked 'Default' at 2 ms. 'Data' is set to 'RTU(8 bits)' and 'Stop' is set to '1 bit'. 'Parity' is set to 'Even'. 'RTS/CTS delay' is 0 X 100 ms.

Description

These zones are used to configure channel parameters. In the online mode, these zones are accessible. In the offline mode, these zones are accessible, but some parameters may not be accessible and are grayed out.

The following table shows the different zones of the Modbus link configuration screen:

Element	Comment
Application parameters (<i>see page 64</i>)	These parameters are accessible via three zones: <ul style="list-style-type: none">● Type,● Master,● Slave.
Signal and physical line parameters (<i>see page 66</i>)	These parameters are accessible via three zones: <ul style="list-style-type: none">● Physical line,● Signals,● RTS/CTS delay.
Transmission parameters (<i>see page 69</i>)	These parameters are accessible via five zones: <ul style="list-style-type: none">● Transmission speed,● Delay between frames,● Data,● Stop bits,● Parity.

NOTE: When configuring Modbus Serial communication in master mode, the **Slave** zone is grayed out and cannot be modified and conversely.

Default Values

The following table shows the default values for Modbus Serial communication parameters:

Configuration parameters		Value
Application parameters	Type	Slave
	Slave number	1
Signal and physical line parameters	Physical line	RS485
	Signals	RX/TX
Transmission parameters	Transmission speed	19200 bits/s
	Delay between frames	2 ms
	Data	RTU (8 bits)
	Stop	1 bit
	Parity	Even

Application-linked Modbus Parameters

At a Glance

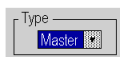
After configuring the communication channel, you need to enter the application parameters.

These parameters are accessible from three configuration zones:

- The **Type** zone,
- The **Master** zone,
- The **Slave** zone.

The Type Zone

This configuration zone appears on the screen as shown below:

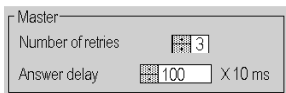


This zone enables you to select the role to be configured for the module in the Modbus serial communication:

- **Master**: When the module is the master.
- **Slave**: When the module is a slave.

The Master Zone

The configuration zone shown below is only accessible when **Master** is selected in the **Type** zone:



This zone enables you to enter the following parameters:


- **Number of retries**: number of connection attempts made by the master before defining the slave as absent.
The default value is 3.
Possible values range from 0 to 15.
A value of 0 indicates no retries by the master.
- **Answer delay**: the time between the master's initial request and a repeated attempt if the slave does not respond. This is the maximum time between the transmission of the last character of the master's request and the receipt of the first character of the request sent back by the slave.
The default value is 1 second (100*10 ms).
Possible values range from 10 ms to 10 s.

NOTE: The Answer delay of the master must be at least equal to the longest Answer delay of the slaves present on the bus.

NOTE: In broadcast mode, the value configured as **Answer Delay** is used as broadcast delay: minimum time between two exchanges in broadcast mode.

The Slave Zone

The configuration zone shown below is only accessible when **Slave** is selected in the **Type** zone:



The image shows a configuration window titled "Slave". It contains a "Slave number" field with a numeric keypad and a value of "98". To the right of the field is an unchecked checkbox labeled "External".

This zone enables you to enter the processor's slave number:

The default value is 1.

Possible values range from 1 to 247.

Selection of **External** grays the **Slave number** field and makes the module use the value of the slave address saved into its internal (*see page 71*) FLASH memory.

NOTE: If the address stored into the FLASH is not into the MODBUS range address, then the default slave address 248 will be used.

When the firmware of the module is updated, the default slave address stored into the FLASH is set to 248. A new command has to be used to re-initialize the FLASH address.

Signal and Physical Line Parameters in Modbus

At a Glance

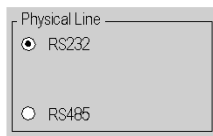
After configuring the communication channel, you need to enter the signal and physical line parameters.

These parameters are accessible via three zones:

- The **Physical Line** zone,
- The **Signals** zone,
- The **RTS/CTS Delay** zone.

The Physical Line Zone

This configuration zone shown below is accessible only on Channel 0 (it is grayed out and configured to RS485 on Channel 1):



In this zone, you can choose between two types of physical line for the serial port on the BMXNOM0200 module:

- The RS232 line,
- The RS485 line.

The Signals Zone

NOTE: If the **Physical line** configuration is **RS485**, the entire zone is grayed out and the default value is **RX/TX**.

The RS232 available signals are dependent to the Control Expert **Hardware Catalog** device as shown below:

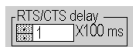
BMXNOM0200	BMXNOM0200.2 BMXNOM0200.3 BMXNOM0200.4
<p>Signals</p> <ul style="list-style-type: none"><input checked="" type="radio"/> RX/TX<input type="radio"/> RX/TX + RTS/CTS<input type="radio"/> RX/TX + RTS/CTS + DTR/DSR	<p>Signals</p> <ul style="list-style-type: none"><input checked="" type="radio"/> RX/TX<input type="radio"/> RX/TX + RTS/CTS DTE mode<input type="radio"/> RX/TX + RTS/CTS DCE mode<input type="radio"/> RX/TX + RTS/CTS + DTR/DSR/DCD

In this zone, you can select the signals supported by the RS232 physical line:

- RX/TX
- Hardware flow management signals:
 - RX/TX + RTS/CTS
 - RX/TX + RTS/CTS DTE mode
 - RX/TX + RTS/CTS DCE mode
- Modem signals:
 - RX/TX + RTS/CTS + DTR/DSR
 - RX/TX + RTS/CTS + DTR/DSR/DCD

The RTS/CTS Delay Zone

This configuration zone appears on the screen as shown below:



RTS/CTS delay zone is available when configuring a signal with an RTS/CTS hardware flow control.

The RTS/CTS hardware flow control algorithm aims at preventing the overflow reception buffer (full duplex).

The RTS/CTS delay corresponds to the time out delay between the RTS rise up and the CTS rise up. A RTS/CTS delay value different from 0 also corresponds to the maximum waiting time between each character transmission after the rise of RTS and CTS signals. If the value is set to 0, UARTs can get stuck in a waiting state for an infinite time until the CTS rise up so the value 0 is used only in particular cases such as looping the RTS signal to the CTS signal in order to check that all connection are operating correctly.

NOTE: The default value is 0 ms.

Transmission-linked Modbus Parameters

At a Glance

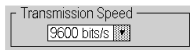
After configuring the communication channel, you need to enter the transmission parameters.

These parameters are accessible from five zones:

- The **Transmission Speed** zone,
- The **Delay Between Frames** zone,
- The **Data** zone,
- The **Stop** zone,
- The **Parity** zone.

The Transmission Speed Zone

This configuration zone appears on the screen as shown below:



You can use it to select the transmission speed of the Modbus serial link. The selected speed has to be consistent with the other devices. The configurable values are 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 and 115200 (only on channel 0 in RS232 mode) bits per second.

The Delay Between Frames Zone

This configuration zone shown below is only accessible in RTU mode (it is grayed in ACSCII mode):



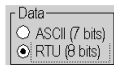
The **Delay Between Frames** is the minimum time separating two frames on reception. This delay is managed when the BMXNOM0200 (master or slave) is receiving messages.

NOTE: The default value depends on the selected transmission speed.

NOTE: The delay between frames should be the Default value in order to be Modbus compliant. In case a slave is not conform, the value can be changed and should be identical for the master and all slaves on the bus.

The Data Zone

This configuration zone appears on the screen as shown below:



This zone allows you to enter the type of coding used to communicate using Modbus serial link. This field is set according to the other devices connected on the bus. There are two configurable modes:

- RTU mode:
 - The characters are coded over 8 bits.
 - The end of the frame is detected when there is a silence of at least 3.5 characters.
 - The integrity of the frame is checked using a word known as the CRC checksum, which is contained within the frame.
- ASCII mode:
 - The characters are coded over 7 bits.
 - The beginning of the frame is detected when the ":" character is received.
 - The end of the frame is detected by a carriage return and a line feed.
 - The integrity of the frame is checked using a byte called the LRC checksum, which is contained within the frame.

The Stop Zone

This configuration zone appears on the screen as shown below:



The **Stop** zone allows you to enter the number of stop bits used for communication. This field is set according to the other devices. The configurable values are:

- **1 bit**
- **2 bits**

The Parity Zone

This configuration zone appears on the screen as shown below:



This zone enables you to determine whether a parity bit is added or not, as well as its type. This field is set according to the other devices. The configurable values are:

- **Even**
- **Odd**
- **None**

How to Set the BMXNOM0200 MODBUS Slave Address Without Control Expert?

Condition and Prerequisite

The FLASH address can be updated in any mode but it is taken into account only when an `operating` mode is performed.

The list below indicates the conditions and prerequisite to set the BMXNOM0200 MODBUS address without Control Expert:

- To use the FLASH address, the module must be configured:
 - In MODBUS slave protocol with the **EXTERNAL** checkbox.
 - In MODBUS master protocol or in **CHAR** mode and then switched to MODBUS slave protocol.

Update the MODBUS Slave Address into the FLASH by Applicative Commands

The table below indicates the operations to update the MODBUS slave address into the FLASH by applicative commands:

Step	Action
1	Store the slave address into the <code>%MWr.m.c.25</code> .
2	Set the bit <code>%MWr.m.c.24.7</code> .
3	Send the <code>WRITE_CMD</code> to the module channel.
4	Check the command end (<code>%MWr.m.c.0.1</code> fall down) and the command is accepted (<code>%MWr.m.c.1.1</code> is at zero means no error) => the FLASH is updated.
5	Perform one of the following operating modes onto the channel to take the new address into account: <ul style="list-style-type: none">● Application Download● Cold Start● Warm Start● Hot Swap● Switch protocol (TO SLAVE)
6	Perform a <code>READ_STS</code> onto the channel to check the slave address in the <code>%MWr.m.c.3</code> most significant byte.

NOTE: Several orders can be embedded in the same command. If one of the orders cannot be executed, the whole command will be rejected and no order is executed.

Update the MODBUS Slave Address into the FLASH Over the Serial Line

The table below indicates the operations to update the MODBUS slave address into the FLASH over the serial line:

Step	Action
1	Configure the MASTER equipment with the same serial line parameter than a channel of the module.
2	Connect the MASTER to the module in point to point.
3	Send the request 0x11 to the point to point address: 0xF8 0x11 0x01 channelnumber(0 or 1) slaveID(0..0xF8)
4	Check the response is OK => the FLASH is updated.
5	Perform an operating mode onto the channel to take the modification in step 4 into account.
6	Send a request 0x11 to check the new slave address: slaveID 0x11 0x01

NOTE: Do not modify the FLASH regularly to avoid to damage this component (100,000 writing cycles max).

Section 4.3

Modbus Serial Communication Programming

Subject of this Section

This section describes the programming process involved in implementing Modbus serial communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
Services Supported by a Modbus Link Master Module	74
Services Supported by a Modbus Link Slave Module	75
Detail of Modbus Expert Mode	77

Services Supported by a Modbus Link Master Module

At a Glance

When used as the master in a Modbus link, a BMXNOM0200 module supports several services via communication functions. These functions are platform-dependent.

Communication Functions

Specific communication functions are defined for sending and receiving data via a Modbus communication channel:

Platform	Read variables/registers	Write variables/registers	Send Modbus request
M580	READ_VAR	WRITE_VAR	DATA_EXCH
M340	READ_VAR	WRITE_VAR	DATA_EXCH
Quantum	READ_REG_QX	WRITE_REG_QX	EXCH_QX

For detailed information on these communication functions, refer to *EcoStruxure™ Control Expert, Communication, Block Library*.

Data Exchanges

Reading or writing of variables are carried out by addressing following requests to the targeted slave device.

These requests use communication functions:

Modbus request	Function code	Communication function
Read bits	16#01 or 16#02	READ_VAR, READ_REG_QX
Read words	16#03 or 16#04	READ_VAR, READ_REG_QX
Write bits	16#0F	WRITE_VAR, WRITE_REG_QX
Write words	16#10	WRITE_VAR, WRITE_REG_QX

More generally, it is possible to send any Modbus requests to a slave device by using the DATA_EXCH or EXCH_QX communication functions (platform-dependent).

Services Supported by a Modbus Link Slave Module

At a Glance

When used as a slave in a Modbus link, a BMXNOM0200 module supports several services.

Data Exchanges

A slave module manages the following requests:

Modbus request	Function code	PLC object
Read n output bits	16#01	%M
Read n output words	16#03	%MW
Write n output bits	16#0F	%M
Write n output words	16#10	%MW
Read/Write n output words	16#17	%MW

NOTE: Read/Write multiple %MW

The `WRITE` performs before the `READ` to be able to write and read the same registers in same time as `IOscanning`. If the exchange size of the `WRITE` or the `READ` is out of boundary then the return status will be "ILLEGAL DATA ADDRESS", but if only the `READ` fail then the `WRITE` will be done with the same status.

Diagnostics and Maintenance

The diagnostics and maintenance requests managed by a Modbus slave BMXNOM0200 module are listed below:

Designation	Function code/sub-function code
Read exception status	16#07
Restart Communications Option	16#08 / 16#01
Return Diagnostic Register	16#08 / 16#02
Change ASCII Input Delimiter	16#08 / 16#03
Force Listen Only Mode	16#08 / 16#04
Clear Counters and Diagnostic Register	16#08 / 16#0A
Return Bus Message Count	16#08 / 16#0B
Return Bus Communication Error Count	16#08 / 16#0C
Return Bus Exception Error Count	16#08 / 16#0D
Return Slave Message Count	16#08 / 16#0E
Return Slave No Response Count	16#08 / 16#0F
Return Slave Negative Acknowledgements Count	16#08 / 16#10

Designation	Function code/sub-function code
Return Slave Busy Count	16#08 / 16#11
Return Bus Character Overrun Count	16#08 / 16#12
Get Communication event Counter	16#0B
Get Communication event Log	16#0C
Report Slave identification	16#11
Write Slave identification	16#11 / 16#01

Detail of Modbus Expert Mode

Expert Mode Communication

Expert mode is a set of commands that can be sent to the module to get extra features.

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.24	CONTROL	Explicit	INT	Command signal, change protocol
%MWr.m.c.24.0		Explicit	BOOL	Erase local counters
%MWr.m.c.24.1		Explicit	BOOL	Change dynamically the retries count in MODBUS master mode (%MW26)
%MWr.m.c.24.2		Explicit	BOOL	Change the slave answer delay (%MW28) for a specific slave (%MW27) in master mode
%MWr.m.c.24.3		Explicit	BOOL	Modify the default slave blind time, the slave ignore received char after a frame reception forwarded to the CPU (%MW29)
%MWr.m.c.24.4		Explicit	BOOL	Modify the MODBUS RTU internal timings t1,5ch (%MW31), t3,5ch (%MW30), and inter exchange delay (%MW32). This value update may disturb the module if it's working
%MWr.m.c.24.6		Explicit	BOOL	Change HALF/FULL DUPLEX modem management mode <ul style="list-style-type: none"> ● If set simultaneously with RTS_ON (%MWr.m.c.24.10 works also with RTS_OFF %MWr.m.c.24.11 and use DTR if %MWr.m.r.24.8 or %MWr.m.r.24.9 is used) the half duplex modem mode is activated ● If this bit is set but none of the RTS/DTR (neither %MWr.m.c.24.8, %MWr.m.c.24.9, %MWr.m.c.24.10, %MWr.m.c.24.11) the full duplex mode is activated <p>The %MW26 is used to set the StartDelay and %MW27 is used to set the EndDelay. So the bit %MW24.5 and %MW24.1 and %MW24.2 cannot be used simultaneously. NOTE: The user may have to restore the correct state of the RTS/DTR signals after the command has been accepted.</p>
%MWr.m.c.24.7	SAVE_SLAVE_ADDR	Explicit	BOOL	Save the Modbus slave address into the FLASH (%MW25).

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.24.8	DTR_ON	Explicit	BOOL	Set the DTR signal (positive voltage)
%MWr.m.c.24.9	DTR_OFF	Explicit	BOOL	Reset the DTR signal (negative voltage)
%MWr.m.c.24.10		Explicit	BOOL	Set the RTS signal (positive voltage)
%MWr.m.c.24.11		Explicit	BOOL	Reset the RTS signal (negative voltage)
%MWr.m.c.24.12	TO_MODBUS_MASTER	Explicit	BOOL	Switch to master mode
%MWr.m.c.24.13	TO_MODBUS_SLAVE	Explicit	BOOL	Switch to slave mode
%MWr.m.c.24.14	TO_CHAR_MODE	Explicit	BOOL	Switch to character mode
%MWr.m.c.25	SLAVE_ADDR	Explicit	INT	Modbus slave address to store in FLASH
%MWr.m.c.26		Explicit	INT	LOW BYTE : MasterRetries count: Retry number in master mode [0..15] see %MW24.1 StartDelay if %MW26.6 is set. Time to wait after the CTS is OK before to start to send the frame. It is useful for modem that requires extra time after CTS or do not manage the CTS signal (in this case the RTS must be connected to the CTS). This time is in millisecond, the precision is about 3ms. Can be performed only in RS232 mode.
%MWr.m.c.27		Explicit	INT	LOW BYTE : Slave for which the master will adapt the answer delay [0..248, 255=ALL] see %MW24.2 and %MW28 EndDelay if %MW24.6 is set. Time to wait after having sent a frame, before to release the RTS signal to let enough time to the MODEM to completely send the frame before hand-up. This time is in millisecond, the precision is about 3ms. Can be performed only in RS232 mode.
%MWr.m.c.28		Explicit	INT	Specific answer delay for a slave in 10ms [1..1000] see %MW24.2 and %MW27
%MWr.m.c.29		Explicit	INT	Blind time in 10ms [1..10] see %MW24.3
%MWr.m.c.30		Explicit	INT	T3,5char: Inter frame delay in milliseconds [0..10000]. The value used depends of the speed. If the value is smaller or greater than possible values, the lower limit or upper limit is applied, and the command is accepted. A value 0 means no change in RTU. The answer delay is computed again.

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.31		Explicit	INT	T1,5char : Delay between char in milliseconds [0..9999]. The value used depends of the speed. If the value is smaller or greater than possible values, the lower limit or upper limit is applied, and the command is accepted. A value 0 means compute T1,5 as T3,5ch – 2ch (default compute).
%MWr.m.c.32		Explicit	INT	Master inter exchange delay in RTU mode [0..256] in miliseconds. The value 0 means “no delay”, if the value is less than 10bits duration, the minimal value of 10 bits is used.

Sample of Code

```
(* master sideNOM is is rack 0 slot 9 *)
if HalfModemMaster then
  HalfModemMaster:=false;
  %MW0.9.0.24:=16#0450>(* switch to half duplex mode with RTS, and change MODBUS timings*)
  %MW0.9.0.26:=12>(* 12ms to wait before sending when CTS raise *)
  %MW0.9.0.27:=9; (* let RTS up 9ms after sending end *)
  %MW0.9.0.30:=0;
  %MW0.9.0.31:=0>(*use the value of the configuration screen equal 6ms *)
  %MW0.9.0.32:=50; (*50ms of delay before sending a new frame*)
  write_cmd(%ch0.9.0);(* send command and data to the NOM channel*)
end_if;
(* slave side the NOM is in rack 0 slot 3 *)
if HalfModemSlave then
  HalfModemSlave:=false;
  %MW0.3.0.24:=16#0448>(* switch to half duplex mode with RTS, and change the slave blind time*)
  %MW0.3.0.26:=12>(* 12ms to wait before sending when CTS raise *)
  %MW0.3.0.27:=9; (* let RTS up 9ms after sending end *)
  %MW0.3.0.29:=4; (* 4*10ms of blind time *)
  write_cmd(%ch0.3.0);(* send command and data to the NOM channel*)
end_if;
```

```

(* optional: sending the command automatically *)
if %S0 or %S1 or %S13 then
  memoSendCmd:=true;
end_if;
(* copy each cycle the module error to detect module disparition *)
memoSendCmd:=%I0.3.0.ERR;
(* if the module is OK send the command one time *)
if FE(memoSendCmd) then
  HalfModemSlave:=true;
end_if;

```

NOM Internal Register Readable

Nom internal registers can be accessed only in MODBUS mode by using the READ_VAR EF. Sample of code (the NOM module is in rack 0 slot 3):

```

if dataCh030GetChannelGlobalInfo then
  read_var(addm('0.3.0'), '%MW', 200, 3, dataCh030Mgt, dataCh030Buff);
(* Internal_Reg@200 are copied into the buffer dataCh030Buff *)
dataCh030GetChannelGlobalInfo := false;
end_if;

```

- Internal_Reg@0 : StartDelay in ms (precision about 3ms) (read or write access)
- Internal_Reg@1 : EndDelay in ms (precision about 3ms) (read or write access)
- Internal_Reg@200 : interface version number = 1
- Internal_Reg@201 : slave address stored in FLASH
- Internal_Reg@202 : 1=possible to change the FLASH, 0=forbidden to change it
- Internal_Reg@1000 : Modbus master RTU internal code ch0=1110, ch1=2110
- Internal_Reg@1002 : 0 = Full Duplex - Hardware flow control, or RS485 ; 1 = Half Duplex - Direction managed automatically by the module with RTS
- Internal_Reg@1010 : Internal sending inter char delay in bits (nbbits*1000/speed => duration in ms) [T1,5S].
- Internal_Reg@1012 : Internal reception inter char delay in bits [T1,5R].
- Internal_Reg@1014 : Internal sending inter frame delay in bits [T3,5S]
- Internal_Reg@1016 : Internal reception inter frame delay in bits [T3,5R]
- Internal_Reg@1018 : Delay to wait before sending the next frames in bits.
- Internal_Reg@1090 : MasterRetries count.
- Internal_Reg@1100 : Slave answer delay for broadcast in 10ms.
- Internal_Reg@1101 : Slave answer delay for slave 1 in 10ms.
- ...
- Internal_Reg@1348 : Slave answer delay for point to point address (248).

-
- Internal_Reg@1500 : Modbus RTU slave internal code ch0=1120, ch1=2120
 - Internal_Reg@1502 : 0 = Full Duplex - Hardware flow control, or RS485 ; 1 = Half Duplex - Direction managed automatically by the module with RTS
 - Internal_Reg@1510 : Internal sending inter char delay in bits ($\text{nbbits} \times 1000 / \text{speed} \Rightarrow \text{duration in ms}$) [T1,5S].
 - Internal_Reg@1512 : Internal reception inter char delay in bits [T1,5R].
 - Internal_Reg@1514 : Internal sending inter frame delay in bits [T3,5S].
 - Internal_Reg@1516 : Internal reception inter frame delay in bits [T3,5R].
 - Internal_Reg@1518 : Delay to wait before sending the next frames in bits.
 - Internal_Reg@1602 : Blind time after reception in ms.
 - Internal_Reg@1606 : Listen Only Mode active = 1, (not active = 0).
 - Internal_Reg@2000 : Modbus master ASCII internal code ch0=1210, ch1=2210
 - Internal_Reg@2002 : 0 = Full Duplex - Hardware flow control, or RS485 ; 1 = Half Duplex - Direction managed automatically by the module with RTS
 - Internal_Reg@2010 : Internal sending inter char delay in bits ($\text{nbbits} \times 1000 / \text{speed} \Rightarrow \text{duration in ms}$) [T1,5S].
 - Internal_Reg@2012 : Internal reception inter char delay in bits [T1,5R].
 - Internal_Reg@2014 : Internal sending inter frame delay in bits [T3,5S].
 - Internal_Reg@2014 : Internal sending inter frame delay in bits [T3,5S].
 - Internal_Reg@2014 : Internal sending inter frame delay in bits [T3,5S].
 - Internal_Reg@2016 : Internal reception inter frame delay in bits [T3,5R].
 - Internal_Reg@2018 : Delay to wait before sending the next frames in bits.
 - Internal_Reg@2090 : MasterRetries count.
 - Internal_Reg@2100 : Slave answer delay for broadcast in 10ms.
 - Internal_Reg@2101 : Slave answer delay for slave 1 in 10ms.
 - ...
 - Internal_Reg@2348 : Slave answer delay for point to point address (248).
 - Internal_Reg@2500 : Modbus ASCII slave internal code ch0=1220, ch1=2220
 - Internal_Reg@2502 : 0 = Full Duplex - Hardware flow control, or RS485 ; 1 = Half Duplex - Direction managed automatically by the module with RTS
 - Internal_Reg@2510 : Internal sending inter char delay in bits ($\text{nbbits} \times 1000 / \text{speed} \Rightarrow \text{duration in ms}$) [T1,5S].
 - Internal_Reg@2512 : Internal reception inter char delay in bits [T1,5R].
 - Internal_Reg@2514 : Internal sending inter frame delay in bits [T3,5S].
 - Internal_Reg@2516 : Internal reception inter frame delay in bits [T3,5R].
 - Internal_Reg@2518 : Delay to wait before sending the next frames in bits.
 - Internal_Reg@2600 : Slave address in use.
 - Internal_Reg@2602 : Blind time after reception in ms.
 - Internal_Reg@2606 : Listen Only Mode active = 1, (not active = 0).
 - Internal_Reg@3000 : Char mode internal code ch0=1000, ch1=2000
 - Internal_Reg@3002 : 0 = Full Duplex - Hardware flow control, or RS485 ; 1 = Half Duplex - Direction managed automatically by the module with RTS
 - Internal_Reg@3100 : 0=no stop criteria active, 1 stop on silence or stop on end of char
 - Internal_Reg@3102 : Internal silence in bits (min is 2 bits, max is 65535 bits)

-
- Internal_Reg@3104 : First End of frame byte to use 16#0100 means no byte
 - Internal_Reg@3106 : First EOF : 1=end of frame byte to let in the frame, 0=remove the end of frame byte
 - Internal_Reg@3108 : Second end of frame byte
 - Internal_Reg@3110 : Second EOF : 1=end of frame byte to let in the frame, 0=remove the end of frame byte

Section 4.4

Debugging Modbus Serial Communication

Modbus Serial Communication Debug Screen

General

The Modbus serial communication debug screen can only be accessed in online mode.

Accessing the Debug Screen

The following table describes the procedure for accessing the debug screen for Modbus serial communication:

Step	Action
1	Access the configuration screen for Modbus serial communication. <i>(see page 62)</i>
2	Select the "Debug" tab on the screen that appears.

Description of the Debug Screen

The debug screen is divided into two or three zones:

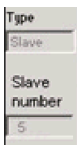
- The Type and Slave number zone,
- The Counters zone,
- The Signals zone (if RS232).

The Type and Slave number Zone

If the module has the function of Master in the Modbus link, this zone looks as following:

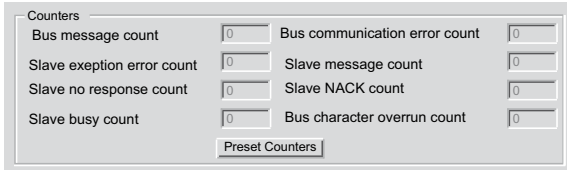


If the module has the function of Slave in the Modbus link, this zone looks as following:



The Counters Zone

This zone looks like this:



The screenshot shows a window titled "Counters" with a light gray background. It contains eight input fields, each with a numerical value of "0". The fields are arranged in two columns. The left column contains: "Bus message count", "Slave exception error count", "Slave no response count", and "Slave busy count". The right column contains: "Bus communication error count", "Slave message count", "Slave NACK count", and "Bus character overrun count". At the bottom center of the window is a button labeled "Preset Counters".

This zone shows the various debugging counters.

The Reset Counters button resets all the debug mode counters to zero.

Counter Operation

The Modbus serial communication debugging counters are:

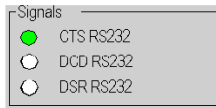
- **Bus message counter:** This counter indicates the number of messages that the module has detected on the serial link. Messages with a negative CRC check result are not counted.
- **Bus communication error counter:** This counter indicates the number of negative CRC check results counted by the module. If a character error (overflow, parity error) is detected, or if the message is less than 3 bytes long, the system that receives the data cannot perform the CRC check. In such cases, the counter is incremented accordingly.
- **Slave exception error counter:** This counter indicates the number of Modbus exception errors detected by the module.
- **Slave message counter:** This counter indicates the number of messages received and processed by the Modbus link.
- **Slave 'no response' counter:** This counter indicates the number of messages sent by the remote system for which it has received no response (neither a normal response, nor an exception response). It also counts the number of messages received in broadcast mode.
- **Negative slave acknowledgement counter:** This counter indicates the number of messages sent to the remote system for which it has returned a negative acknowledgement.
- **Slave busy counter:** This counter indicates the number of messages sent to the remote system for which it has returned a "slave busy" exception message.
- **Bus character overflow counter:** This counter indicates the number of messages sent to the module that it is unable to acquire because of character overflow on the bus. Overflow is caused by:
 - Character-type data that are transmitted on the serial port more quickly than they can be stored,
 - A loss of data due to a hardware event.

NOTE: For all counters, the count begins at the most recent restart, clear counters operation or module power-up.

The Signals Zone

This zone displays only if RS232 is selected in configuration screen. If RS485 is selected in configuration screen, this window is not displayed at all.

The Signals zone looks like this:



This zone indicates the activity of the signals:

- **CTS RS232:** shows the activity of the CTS signal.
- **DCD RS232:** shows the activity of the DCD signal.
- **DSR RS232:** shows the activity of the DSR signal.

Chapter 5

Character Mode Communication

Subject of this Section

This chapter presents the software implementation of communication using Character Mode for BMXNOM0200.

What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
5.1	Generalities	88
5.2	Character Mode Communication Configuration	89
5.3	Character Mode Communication Programming	99
5.4	Debugging Character Mode communication	107

Section 5.1

Generalities

About Character Mode Communication

Introduction

Communication in Character Mode enables dialog and communication functions to be carried out with the following devices:

- Regular peripherals (printer, keyboard-screen, workshop terminal, etc.),
- Specialized peripherals (barcode readers, etc.),
- Calculators (checking, production management, etc.),
- Heterogeneous devices (numerical commands, variable speed controllers, etc),
- External modem.

 WARNING
CRITICAL DATA LOSS
Only use communication ports for non-critical data transfers.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

Section 5.2

Character Mode Communication Configuration

Subject of this Section

This section describes the configuration process used when implementing Character Mode communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
BMXNOM0200 Character Mode Communication Configuration Screen	90
Message End Detection Parameters in Character Mode	92
Transmission Parameters in Character Mode	94
Signal and Physical Line Parameters in Character Mode	96

BMXNOM0200 Character Mode Communication Configuration Screen

General

The following pages provide an introduction to the configuration screen for Character Mode communication.

Accessing the Configuration Screen

The following table describes the procedure for accessing the configuration screen for Character Mode communication:

Step	Action
1	Open the BMX NOM 0200 sub-directory in the project browser (<i>see page 58</i>).
2	Select the Channel to configure and the Character mode link function on the screen that appears.

Character Mode Configuration Screen

The figure below shows the default configuration screen for Character Mode communication on Channel 0:

The screenshot shows the 'Configuration' screen for Channel 0. It is divided into several sections:

- Stop on reception:** Two sections for Character 1 and Character 2. Each has a 'Stop' checkbox, 'CR' and 'LF' checkboxes, and a 'Character included' checkbox. The 'LF' field is set to 0.
- Transmission speed:** A dropdown menu set to '9600 bits/s'.
- Stop on silence:** A 'Stop' checkbox and a field set to '2 ms'.
- Data:** Radio buttons for '7 bits', '8 bits' (selected), '1 bit', and '2 bits'.
- Parity:** Radio buttons for 'Even', 'Odd' (selected), and 'None'.
- Physical line:** Radio buttons for 'RS232' (selected) and 'RS485'.
- Signals:** Radio buttons for 'RX/TX' (selected), 'RX/TX + RTS/CTS DTE mode', 'RX/TX + RTS/CTS DCE mode', and 'RX/TX + RTS/CTS + DTR/DSR/DCD'.
- RTS/CTS delay:** A field set to '0' followed by 'X 100 ms'.
- Polarization:** Radio buttons for 'None' (selected), 'Unique polarization', and 'Distributed polarization'. This section is grayed out.

NOTE: In this example, the **Polarization** and **RTS/CTS delay** zones are grayed out respectively because an RS232 physical line and RX/TX signals have been chosen.

Description

These zones are used to configure channel parameters. In the online mode, these zones are accessible. In the offline mode, these zones are accessible but some parameters may not be accessible and are grayed out.

The following table shows the different zones of the Character Mode communication configuration screen:

Element	Comment
Message end detection parameters (<i>see page 92</i>)	These parameters are accessible via two zones: <ul style="list-style-type: none">● Stop on reception,● Stop on silence.
Signal and physical line parameters (<i>see page 96</i>)	These parameters are accessible via four zones: <ul style="list-style-type: none">● Physical line,● Signals,● RTS/CTS delay,● Polarization.
Transmission parameters (<i>see page 94</i>)	These parameters are accessible via four zones: <ul style="list-style-type: none">● Transmission speed,● Data,● Stop bits,● Parity.

Default Values

The following table shows the default values for Character Mode communication parameters:

Configuration parameters		Channel 0	Channel 1
Message end detection parameters	Stop on reception	Without	Without
	Stop on silence	Without	Without
Signal and physical line parameters	Physical line	RS232	RS485
	Signals	RX/TX	RX/TX
	RTS/CTS delay	Not applicable	Not applicable
	Polarization	Not applicable	None
Transmission parameters	Transmission speed	9600 bits/s	9600 bits/s
	Data	8 bits	8 bits
	Stop	1 bit	1 bit
	Parity	Odd	Odd

Message End Detection Parameters in Character Mode

At a Glance

After configuring the communication channel, you need to enter the message end detection parameters.

These parameters are accessible via two zones:

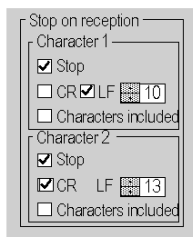
- The **Stop on reception** Zone: stop on reception of a special character.
- The **Stop on silence** Zone: stop on silence.

Conditions of Use

Selecting **Stop on silence** means that **Stop on reception** is deselected and vice versa.

The Stop on reception Zone

This configuration zone appears on the screen as shown below:



A reception request can be terminated once a specific character is received.

By checking the **Stop** option, it is possible to configure **Stop on reception** to be activated by a specific end-of-message character:

- **CR**: enables you to detect the end of the message by a carriage return.
- **LF**: enables you to detect the end of the message by a line feed.
- **Data entry field**: enables you to identify an end-of-message character other than the carriage return or line feed characters, using a decimal value:
 - Between 0 and 255 if the data is coded over 8 bits
 - Between 0 and 127 if the data is coded over 7 bits
- **Character included**: enables you to include the end-of-message character in the reception table of the PLC application.

It is possible to configure two end-of-reception characters. In the above window, the end of reception of a message is detected by an line feed or carriage return character.

The Stop on silence Zone

This configuration zone appears on the screen as shown below:



This zone enables you to detect the end of a message on reception by the absence of message end characters over a given time.

Stop on silence is validated by checking the **Stop** box. The duration of the silence (expressed in milliseconds) is set using the data entry field.

The minimal value of this duration is the time corresponding to the transmission of 1.5 characters. Expressed in number of bits, and depending on the configuration of start and stop bits, the minimal silence duration is as follows:

Total character length (bit)	Minimal silence duration (bit)
8	12
9	12
10	15
11	15

Convert the number in right column in time according to the configured speed transmission.

NOTE: The available values range from 1 ms to 10000 ms and depend on the transmission speed selected.

Transmission Parameters in Character Mode

At a Glance

After configuring the communication channel, you need to enter the transmission parameters.

These parameters are accessible via four zones:

- The **Transmission speed** zone,
- The **Data** zone,
- The **Stop** zone,
- The **Parity** zone.

The Transmission speed Zone

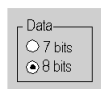
This configuration zone appears on the screen as shown below:



You can use this zone to select the transmission speed of the Character Mode protocol. The selected speed has to be consistent with the other devices. The configurable values are 300, 600, 1200, 2400, 4800, 9600, 19200, 57600 and 115200 (only on channel 0 in RS232 mode) bits per second.

The Data Zone

This configuration zone appears on the screen as shown below:



In this zone, you can specify the size of the data being exchanged on the link.

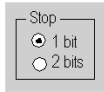
The available values are:

- **7 bits**
- **8 bits**

You are advised to adjust the number of data bits according to the remote device being used.

The Stop Zone

This zone looks like this:



The **Stop** zone allows you to enter the number of stop bits used for communication. You are advised to adjust the number of stop bits according to the remote device being used.

The configurable values are:

- **1 bit**
- **2 bits**

The Parity Zone

This configuration zone appears on the screen as shown below:



This zone enables you to determine whether a parity bit is added or not, as well as its type. You are advised to adjust parity according to the remote device being used.

The configurable values are:

- **Even**
- **Odd**
- **None**

Signal and Physical Line Parameters in Character Mode

At a Glance

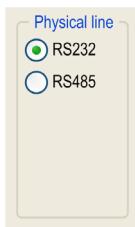
After configuring the communication channel, you need to enter the signal and physical line parameters.

These parameters are accessible via three zones:

- The **Physical line** zone
- The **Signals** zone
- The **RTS/CTS Delay** zone

The Physical line Zone

This configuration zone appears on the screen as shown below:

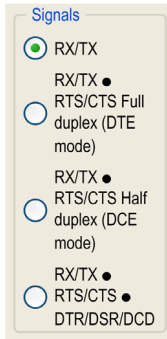


In this zone, you can choose between two types of physical line for the serial port on the BMXNOM0200 module:

- The RS 232 line
- The RS 485 line

The Signals Zone

This configuration zone appears on the screen as shown below:



Signals

- RX/TX
- RX/TX + RTS/CTS Full duplex (DTE mode)
- RX/TX + RTS/CTS Half duplex (DCE mode)
- RX/TX + RTS/CTS + DTR/DSR/DCD

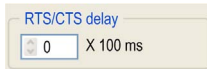
In this zone, you can select the signals supported by the RS 232 physical line:

- RX/TX
- RX/TX + RTS/CTS Full Duplex (DTE mode)
- RX/TX + RTS/CTS Half Duplex (DCE mode)
- RX/TX + RTS/CTS + DTR/DSR/DCD

If the RS 485 is configured, the entire zone is grayed out and the default value is RX/TX.

The RTS/CTS Delay Zone

This configuration zone appears on the screen as shown below:



RTS/CTS delay

X 100 ms

RTS/CTS Delay zone is available only when both RS232 and RX/TX+RTS/CTS or RX/TX+RTS/CTS+DTR/DSR/DCD check boxes are selected. An RTS/CTS hardware flow control is performed.

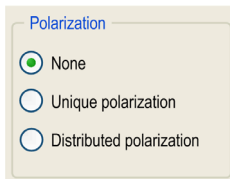
The RTS/CTS hardware flow control algorithm aims at preventing the overflow reception buffer (full duplex).

The RTS/CTS delay corresponds to the time out delay between the RTS rise up and the CTS rise up. A RTS/CTS delay value different from 0 also corresponds to the maximum waiting time between each character transmission after the rise of RTS and CTS signals. If the value is set to 0, UARTs can get stuck in a waiting state for an infinite time until the CTS rise up so the value 0 is used only in particular cases such as looping the RTS signal to the CTS signal in order to check that all connection cables are operating correctly.

NOTE: The default value is 0 ms.

The Polarization zone

This configuration zone shown below is accessible when **RS485** is selected in the **Physical line** zone:



The image shows a configuration window titled "Polarization" with a light beige background. It contains three radio button options: "None" (which is selected, indicated by a green dot), "Unique polarization", and "Distributed polarization".

This zone gives the capability to choose between three types of configuration for the polarization on the channel:

- **None** to use no polarization in case you have your own termination.
- **Unique polarization** to use a low impedance polarization like in Modbus networks (the goal of this kind of polarization is to let the master maintain the default state).
- **Distributed polarization** to use a high polarization impedance (the goal of this kind of polarization is to let each device contribute to maintain the default state).

Section 5.3

Character Mode Communication Programming

Subject of this Section

This section describes the programming process used when implementing Character Mode communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
Character Mode Communication Functions	100
Detail of Character Mode Expert Mode	104

Character Mode Communication Functions

At a Glance

Specific communication functions are defined for sending and receiving data via a communication channel in Character Mode. These functions are platform-dependent.

Communication Functions

Specific communication functions are defined for sending and receiving data via a communication channel in Character Mode:

Platform	Sending a character string	Reading a character string	Reading a byte array
M580	PRINT_CHAR	INPUT_CHAR	INPUT_BYTE
M340	PRINT_CHAR	INPUT_CHAR	INPUT_BYTE
Quantum	PRINT_CHAR_QX	INPUT_CHAR_QX	–

For detailed information on these communication functions, refer to *EcoStruxure™ Control Expert, Communication, Block Library*.

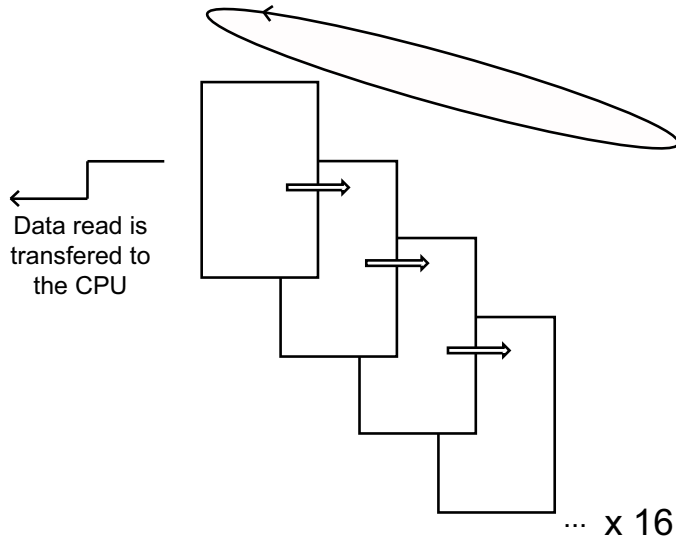
NOTE: For the `INPUT_CHAR` function, a configured timeout is necessary if the channel is configured without **Stop on silence**, to acknowledge the activity bit of the function.

Internal Mechanism of the BMXNOM0200 Module

The BMXNOM0200 can store a total of 16 frames in emission or reception. The frames in buffers are managed in FIFO order. Over RS-232 lines, they are managed in full duplex mode.

The data received is stored in 16 cycling buffers in series, each buffer containing 1,024 bytes.

The figure below represents this mechanism:



Number of Frames Received in Buffers

When the serial port is configured in the character mode, `%MWr.m.c.7` indicates the number of frames in the receiving buffer of the BMXNOM0200 module.

This WORD is incremented each time the BMXNOM0200 receive a frame over an RS-232 line.

Receiving Data

Frames are retrieved by the application program using receiving character functions, `INPUT_CHAR` and `INPUT_CHAR_QX EF` to receive a string, or the `INPUT_BYTE EF` to receive binary data.

The receiving data EF may be executed before data is received by the module. In this case, the module waits for data from the line and then sends it to the CPU.

The EF can also be executed when the frame was already received (for example, after checking the `%MWr.m.c.7` with `READ_STS`). In this case, the module immediately sends the buffered frame to the CPU.

It is also possible to force the module to wait for data from the line by setting the reset parameter of the EF to 1 (. In this case, the data previously buffered are flushed, and the BMXNOM0200 waits for new data to be sent to the CPU.

The module behavior differs according to:

- the channel configuration (**with** or **without** stop parameters),
- the input parameters of the communication functions,
- the states of the buffer before activation of the communication function.

NOTE: The maximum size of a frame sent by the BMXNOM0200 to the CPU is 1024 byte. However internally the reception frame size has a maximum size of 1025 byte if an end of frame byte is configured and this byte is not to be included into the data sent to the CPU.

The table below gives the module behavior under following conditions:

- Channel configured **without** stop parameters
- Input parameter of the EF (NB, or INPUT_LEN) set to 0.

If	Then
If the buffer is not empty before the activation of the EF	The module sends to the CPU the content of the buffer with a max of 1024 characters.
If the buffer is empty before the activation of the EF	The module waits until the reception of the first characters before sending it to the CPU.
If the reset buffer is selected before the activation of the EF	The module flushes the buffer first and waits until the reception of the first next characters.

The table below gives the module behavior under following conditions:

- Channel configured **without** stop parameters, and
- Input parameter of the EF (NB, or INPUT_LEN) set to a value greater than zero.

If	Then
If the buffer is not empty before the activation of the EF	The module waits until the buffer contains NB or INPUT_LEN bytes before sending it to the CPU.
If the buffer is empty before the activation of the EF	The module waits until the reception of the NB or INPUT_LEN characters before sending it to the CPU
If the reset buffer is selected before the activation of the EF	The module flushes the buffer and waits until the reception of the next NB or INPUT_LEN characters.

The table below gives the module behavior under following conditions:

- Channel configured **with** stop parameters (stop on reception of a special character or stop on silence), and
- Input parameter of the EF (`NB`, or `INPUT_LEN`) set to 0.

If	Then
If the buffer contains a message before the activation of the EF	The module sends the message to the CPU with a maximum of 1024 characters.
If the buffer is empty before the activation of the EF	The module waits until the reception of the first message before sending it to the CPU with a max of 1024 characters.
If the reset buffer is selected before the activation of the EF	The module flushes the buffer and waits until the reception of the first next messages.

Zero-size Frames

Zero-size frames are discarded. If an end of frame byte is configured, and not requested as part of the data, a zero size frame received by the BMXNOM0200 is not sent to the CPU. In this case, if an end of frame byte is received without any data before it, this frame is discarded and no information is sent to the CPU.

Receiving Several Frames During a MAST Task

Several frames can be forwarded by the BMXNOM0200 to the CPU during one MAST task, and several `INPUT_CHAR` EF instances may be launched in parallel that address the same BMXNOM0200 module. This can be required if a huge flow of data arrives over the serial line.

Cancelling and Timeout

Cancel and Timeout are forwarded to the BMXNOM0200 module. A Timeout condition and Cancel orders applied to an instance of `INPUT_CHAR` are forwarded to the BMXNOM0200 module. The corresponding pending task is removed from the BMXNOM0200 module task queue.

Internal Mechanism of the BMXNOM0200 Module: Emission

Use the `PRINT_CHAR` or `PRINT_CHAR_QX` EF to send data over the serial line of the BMXNOM0200 module.

NOTE: If several frames have been sent (several EF instances have been called) and a silence has been configured, the BMXNOM0200 module inserts a silence time between each frame.

It is possible to launch up to 16 EF requests: they are sent serially with a silence between each request.

Detail of Character Mode Expert Mode

Expert Mode Communication

Expert mode is a set of commands that can be sent to the module to get extra features.

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.24	CONTROL	Explicit	INT	Command signal, change protocol.
%MWr.m.c.24.0		Explicit	BOOL	Erase local counters.
%MWr.m.c.24.4		Explicit	BOOL	Modify the silence internal timings (%MW30). This value update may disturb the module if it's working.
%MWr.m.c.24.5		Explicit	BOOL	Modify the char mode end of frame byte 0 (%MW26) and byte 1 (%MW27)
%MWr.m.c.24.6		Explicit	BOOL	<p>Change HALF/FULL DUPLEX modem management mode.</p> <ul style="list-style-type: none"> ● If set simultaneously with RTS_ON (%MWr.m.c.24.10 works also with RTS_OFF %MWr.m.c.24.11 and use DTR if .8 or .9 is used) the half duplex modem mode is activated. ● If this bit is set but none of the RTS/DTR (neither %MWr.m.c.24.8, %MWr.m.c.24.9, %MWr.m.c.24.10, %MWr.m.c.24.11), the full duplex mode is activated. <p>The %MW26 is used to set the StartDelay and %MW27 is used to set the EndDelay. So the bit %MW24.5 and %MW24.1 and %MW24.2 cannot be used simultaneously</p> <p>NOTE: The user may have to restore the correct state of the RTS/DTR signals after the command has been accepted.</p>
%MWr.m.c.24.7		Explicit	BOOL	Save the Modbus slave address into the FLASH (%MW25).
%MWr.m.c.24.8	DTR_ON	Explicit	BOOL	Set the DTR signal (positive voltage)
%MWr.m.c.24.9	DTR_OFF	Explicit	BOOL	Reset the DTR signal (negative voltage)
%MWr.m.c.24.10		Explicit	BOOL	Set the RTS signal (positive voltage)
%MWr.m.c.24.11		Explicit	BOOL	Reset the RTS signal (negative voltage)
%MWr.m.c.24.12	TO_MODBUS_MASTER	Explicit	BOOL	switch to master mode

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.24.13	TO_MODBUS_SLAVE	Explicit	BOOL	switch to slave mode
%MWr.m.c.24.14	TO_CHAR_MODE	Explicit	BOOL	Switch to character mode
%MWr.m.c.25		Explicit	INT	Modbus slave address to store in FLASH
%MWr.m.c.26		Explicit	INT	<p>New EOF in char mode (eq %KW6) if %MW24.5 is set:</p> <ul style="list-style-type: none"> ● Bit 0: 1 byte 1 is set below, 0 no more byte 1 ● Bit 1: 1 add the byte 1, 0 do not add the byte 1 ● Bit2..7 : must be null.HIGH BYTE: the end of frame byte 1 <p>StartDelay if %MW26.6 is set. Time to wait after the CTS is OK before to start to send the frame. It is useful for modem that requires extra time after CTS or do not manage the CTS signal (in this case the RTS must be connected to the CTS). This time is in millisecond, the precision is about 3ms. Can be performed only in RS232 mode.</p>
%MWr.m.c.27		Explicit	INT	<p>New EOF in char mode (eq %KW7) if %MW24.5 is set:</p> <ul style="list-style-type: none"> ● Bit 0: 1 byte 2 is set below, 0 no more byte 2 ● Bit 1: 1 add the byte 2, 0 do not add the byte 2 ● Bit2..7 : must be null.HIGH BYTE: the end of frame byte 2 <p>StartDelay if %MW24.6 is set. Time to wait after having sent a frame, before to release the RTS signal to let enough time to the MODEM to completely send the frame before hand-up. This time is in millisecond, the precision is about 3ms. Can be performed only in RS232 mode.</p>
%MWr.m.c.28		Explicit	INT	Reserved
%MWr.m.c.29		Explicit	INT	Reserved

Address	Standard Symbol	Exchange Type	Type	Meaning
%MWr.m.c.30		Explicit	INT	silence: Inter frame delay in milliseconds [0..10000]. The value used depends of the speed. If the value is smaller or greater than possible values, the lower limit or upper limit is applied, and the command is accepted. A value 0 means no silence.
%MWr.m.c.31		Explicit	INT	Reserved
%MWr.m.c.32		Explicit	INT	Reserved

Sample of Code

```

if HalfModemChar then
HalfModemChar:=false;
  %MW0.9.0.24:=16#0440; (* switch to half duplex mode with RTS*)
  %MW0.9.0.26:=12; (* 12ms to wait before sending when CTS raise *)
  %MW0.9.0.27:=9; (* let RTS up 9ms after sending end *)
  write_cmd(%ch0.9.0); (* send command and data to the NOM channel*)
end_if;

```

Section 5.4

Debugging Character Mode communication

Character Mode Communication Debug Screen

General

The Character Mode debug screen is accessible in online mode.

Accessing the Debug Screen

The following table describes the procedure for accessing the debug screen for Character Mode communication:

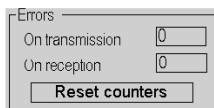
Step	Action
1	Access the configuration screen for Character Mode communication. (<i>see page 90</i>)
2	Select the Debug tab on the screen that appears.

Description of the Debug Screen

The debug screen consists of an **Error** zone and a **Signals** zone (if RS232).

The Error Zone

The **Error** zone looks like this:



The screenshot shows a window titled "Errors" with two input fields: "On transmission" and "On reception", both containing the value "0". Below these fields is a button labeled "Reset counters".

This zone indicates the number of communication interruptions counted by the module:

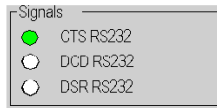
- **On transmission:** corresponds to the number of interruptions on transmission (image of $\%MW4$ word).
- **On reception:** corresponds to the number of interruptions on reception (image of $\%MW5$ word).

The **Reset counters** button resets both counters to zero.

The Signals Zone

This zone is displayed only if RS232 is selected in configuration screen. If RS485 is selected in configuration screen, this window is not displayed at all.

The **Signals** zone looks like this:



This zone indicates the activity of the signals:

- **CTS RS232:** shows the activity of the CTS signal.
- **DCD RS232:** shows the activity of the DCD signal.
- **DSR RS232:** shows the activity of the DSR signal.

Chapter 6

BMXNOM0200 Module Diagnostics

Subject of this Chapter

This chapter describes the diagnostics aspect in the implementation of BMXNOM0200 communication module.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Detailed Diagnostics by Communication Channel	110
Diagnostics of a BMXNOM0200 Module	112

Detailed Diagnostics by Communication Channel

At a Glance


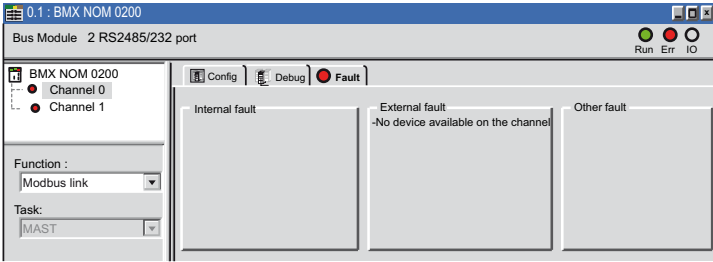
The channel Diagnostics function displays detected errors when they occur, classified according to their category:

- **Internal detected error**
 - self-tests in progress
- **External events**
 - device missing
 - device inoperative
 - serial-link communication time-out
- **Other detected errors**
 - line tool error
 - configuration error
 - communication loss
 - application error

A detected channel error is indicated in the **Debug** tab when the  LED, located in the **Error** column, turns red.

Accessing the Channel Diagnostic Screen

The table below shows the procedure for accessing the channel diagnostic screen.

Step	Action
1	Open the module debugging screen.
2	<p>For the inoperative channel, click on the button  situated in the Error column. Result: The list of detected channel errors appears.</p>  <p>Note: Channel diagnostics information can also be accessed by program (instruction READ_STS).</p>

Channel Detected Errors List

The summary table below shows the various detected errors for a configured serial link:

Detected errors classification	Language objects
Internal fault: <ul style="list-style-type: none">● Self-tests in progress	<ul style="list-style-type: none">● %MWr.m.c.2.4
External fault: <ul style="list-style-type: none">● No device available on the channel● Device fault● Time-out error (CTS)	<ul style="list-style-type: none">● %MWr.m.c.2.0● %MWr.m.c.2.1● %MWr.m.c.2.3
Other fault: <ul style="list-style-type: none">● Line tool error● Hardware configuration fault● Problem communicating with the PLC● Application error	<ul style="list-style-type: none">● %MWr.m.c.2.2● %MWr.m.c.2.5● %MWr.m.c.2.6● %MWr.m.c.2.7

Diagnostics of a BMXNOM0200 Module

At a Glance

The module diagnostics function displays anomalies when they occur, classified according to their category:

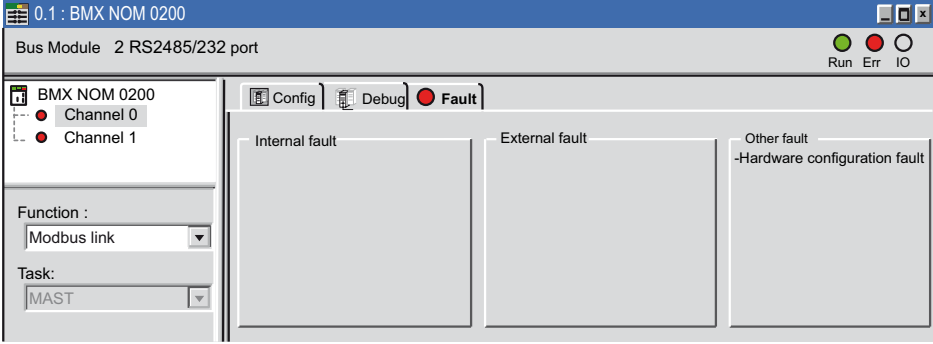
- **Internal detected error:**
 - module event
- **External event:**
 - Wiring control (broken-wire, overload or short-circuit)
- **Other anomalies:**
 - inoperative channel
 - configuration anomaly
 - module missing or off

A detected module error is indicated by a number of LEDs changing to red, such as:

- in the rack-level configuration editor:
 - the LED of the rack number
 - the LED of the slot number of the module on the rack
- in the module-level configuration editor:
 - the **Err** and **I/O** LEDs, depending on the type detected error
 - the **Channel** LED in the **Channel** field

Accessing the Module Diagnostic Screen

The table below shows the procedure for accessing the module diagnostic screen.

Step	Action
1	Open the module debugging screen.
2	<p>Click on the module reference in the channel zone and select the Fault tab. Result: The list of module detected errors appears.</p>  <p>Note: It is not possible to access the module diagnostics screen if a configuration error, major breakdown error, or module missing error is detected. The following message then appears on the screen: "The module is missing or different from that configured for this position."</p>

Module Detected Errors List

The summary table below shows the various detected errors for a communication module:

Detected errors classification	Language objects
<p>Internal fault:</p> <ul style="list-style-type: none"> ● Module detected failure 	<ul style="list-style-type: none"> ● %MWr.m.MOD.2.0
<p>External fault:</p> <ul style="list-style-type: none"> ● Terminal block 	<ul style="list-style-type: none"> ● %MWr.m.MOD.2.2
<p>Other fault:</p> <ul style="list-style-type: none"> ● Faulty channel(s) ● Hardware configuration fault ● Module missing or off 	<ul style="list-style-type: none"> ● %MWr.m.MOD.2.1 ● %MWr.m.MOD.2.5 ● %MWr.m.MOD.2.6

Chapter 7

Language Objects of Modbus and Character Mode Communications

Subject of this Chapter

This chapter describes the language objects associated with Modbus and Character Mode communications and the different ways of using them.

What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
7.1	Language Objects and IODDTs of Modbus and Character Mode Communications	116
7.2	General Language Objects and IODDTs for Communication Protocols	124
7.3	Language Objects and IODDTs Associated with Modbus Communication	128
7.4	Language Objects and IODDTs associated with Character Mode Communication	136
7.5	The IODDT Type T_GEN_MOD Applicable to All Modules	144
7.6	Language Objects and Device DDTs Associated with Modbus Communication	146

Section 7.1

Language Objects and IODDTs of Modbus and Character Mode Communications

Subject of this Section

This section provides an overview of the general points concerning IODDTs and language objects for Modbus and Character Mode communications.

What Is in This Section?

This section contains the following topics:

Topic	Page
Introduction to the Language Objects for Modbus and Character Mode Communications	117
Implicit Exchange Language Objects Associated with the Application-Specific Function	118
Explicit Exchange Language Objects Associated with the Application-Specific Function	119
Management of Exchanges and Reports with Explicit Objects	121

Introduction to the Language Objects for Modbus and Character Mode Communications

General

The IODDTs are predefined by the manufacturer. They contain input/output language objects belonging to the channel of an application-specific module.

Modbus and Character Mode communications have three associated IODDTs:

- T_COM_STS_GEN, which applies to communication protocols except Fipio and Ethernet.
- T_COM_MB_BMX, which is specific to Modbus communication.
- T_COM_CHAR_BMX, which is specific to Character Mode communication.

NOTE: IODDT variables can be created in two different ways:

- Using the I/O objects tab (*see EcoStruxure™ Control Expert, Operating Modes*).
- Using the Data Editor (*see EcoStruxure™ Control Expert, Operating Modes*).

Types of Language Objects

In each IODDT we find a set of language objects that enable us to control them and check that they are operating correctly.

There are two types of language objects:

- Implicit Exchange Objects: These objects are automatically exchanged on each cycle revolution of the task associated with the processor.
- Explicit Exchange Objects: These objects are exchanged on the application's request, using explicit exchange instructions.

Implicit exchanges concern the status of the processors, communication signals, slaves, etc.

Explicit exchanges are used to define the processor settings and perform diagnostics.

Implicit Exchange Language Objects Associated with the Application-Specific Function

At a Glance

Use of an integrated, application-specific interface or the addition of a module automatically enhances the language objects application used to program this interface or module.

These objects correspond to the input/output images and software data of the module or integrated application-specific interface.

Reminders

The module inputs (%I and %IW) are updated in the PLC memory at the start of the task, or when the PLC is in RUN or STOP mode.

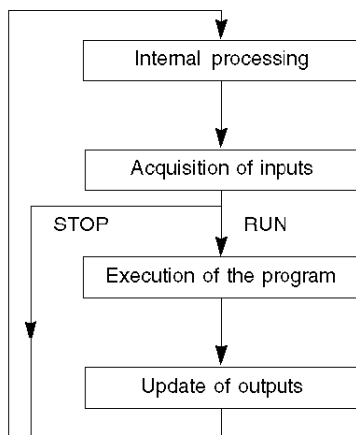
The outputs (%Q and %QW) are updated at the end of the task, only when the PLC is in RUN mode.

NOTE: When the task is in STOP mode, either of the following are possible, depending on the configuration selected:

- Outputs are set to fallback position (fallback mode).
- Outputs are maintained at their last value (maintain mode).

Illustration

The diagram below shows the operating cycle of a PLC task (cyclical execution):



Explicit Exchange Language Objects Associated with the Application-Specific Function

At a Glance

Explicit exchanges are exchanges performed at the user program's request, using the following instructions:

- `READ_STS` (see *EcoStruxure™ Control Expert, I/O Management, Block Library*): read status words
- `WRITE_CMD` (see *EcoStruxure™ Control Expert, I/O Management, Block Library*): write command words

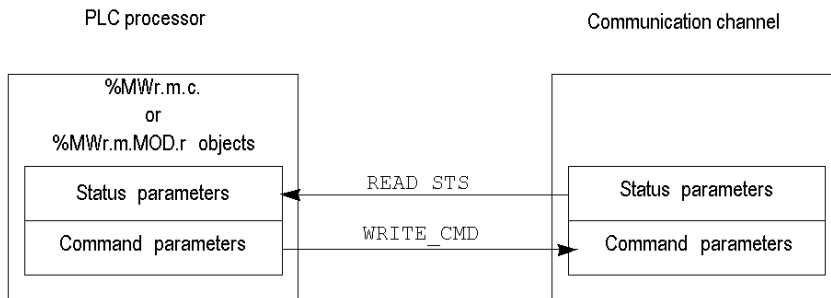
These exchanges apply to a set of `%MW` objects of the same type (status, commands or parameters) belonging to a channel.

NOTE: These objects provide information about the processor or the module, can be used to command them (e.g.: switch command) and to define their operating modes (save and restore adjustment parameters in application).

NOTE: The `READ_STS` and `WRITE_CMD` instructions are executed at the same time as the task that calls them and always correctly. The result of these instructions is available immediately after their execution.

General Principle for Using Explicit Instructions

The diagram below shows the different types of explicit exchanges that can be made between the processor and the communication channel:



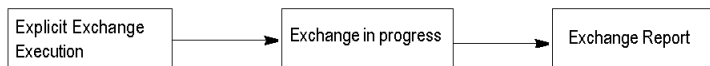
Managing Exchanges

During an explicit exchange, it is necessary to check its performance in order that data is only taken into account when the exchange has been correctly executed.

To this end, two types of information are available:

- Information concerning the exchange in progress (*see EcoStruxure™ Control Expert, I/O Management, Block Library*).
- The exchange report (*see EcoStruxure™ Control Expert, I/O Management, Block Library*).

The following diagram illustrates the management principle for an exchange:



NOTE: In order to avoid several simultaneous explicit exchanges for the same channel, it is necessary to test the value of the word EXCH_STS ($\%MW\mathbb{r}.m.c.0$) of the IODDT associated to the channel before to call any EF using this channel.

Management of Exchanges and Reports with Explicit Objects

At a Glance

When data is exchanged between the PLC memory and the module, the module may require several task cycles to acknowledge this information.

All IODDTs use two words to manage exchanges:

- EXCH_STS (%MWr.m.c.0) : exchange in progress.
- EXCH_RPT (%MWr.m.c.1) : report.

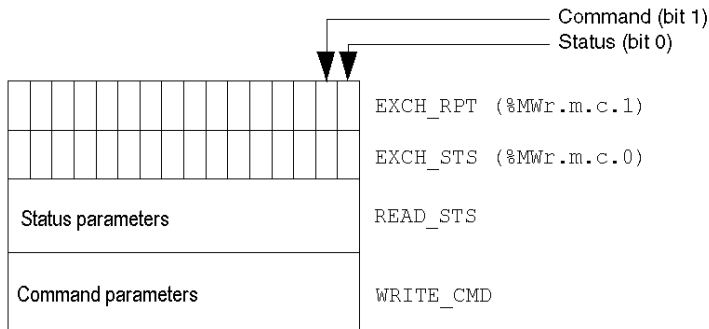
NOTE:

Depending on the localization of the module, the management of the explicit exchanges (%MW0.0.MOD.0.0 for example) will not be detected by the application:

- for in-rack modules, explicit exchanges are done immediately on the local PLC Bus and are finished before the end of the execution task, so the READ_STS, for example, is always finished when the %MW0.0.mod.0.0 bit is checked by the application.
- for remote bus (Fipio for example), explicit exchanges are not synchronous with the execution task, so the detection is possible by the application.

Illustration

The illustration below shows the different significant bits for managing exchanges:



Description of Significant Bits

Each bit of the words EXCH_STS (%MW_{r.m.c.0}) and EXCH_RPT (%MW_{r.m.c.1}) is associated with a parameter type:

- Rank 0 bits are associated with the status parameters:
 - The STS_IN_PROGR bit (%MW_{r.m.c.0.0}) indicates whether a read request for the status words is in progress.
 - The STS_ERR bit (%MW_{r.m.c.1.0}) specifies whether a read request for the status words is accepted by the module channel.
- Rank 1 bits are associated with the command parameters:
 - The CMD_IN_PROGR bit (%MW_{r.m.c.0.1}) indicates whether command parameters are being sent to the module channel.
 - The CMD_ERR bit (%MW_{r.m.c.1.1}) indicates whether or not the command parameters are accepted by the module channel.

NOTE: r corresponds to the number of the rack and m to the position of the module in the rack, while c corresponds to the channel number in the module.

NOTE: Exchange and report words also exist at module level EXCH_STS (%MW_{r.m.MOD.0}) and EXCH_RPT (%MW_{r.m.MOD.1}) as per T_GEN_MOD type IODDTs.

Explicit Exchange Execution Flags: EXCH_STS

The table below shows the EXCH_STS word (%MW_{r.m.c.0}) explicit exchange control bits:

Standard symbol	Type	Access	Meaning	Address
STS_IN_PROGR	BOOL	R	Reading of channel status words in progress	%MW _{r.m.c.0.0}
CMD_IN_PROGR	BOOL	R	Command parameters exchange in progress	%MW _{r.m.c.0.1}
ADJ_IN_PROGR	BOOL	R	Adjust parameters exchange in progress	%MW _{r.m.c.0.2}
RECONF_IN_PROGR	BOOL	R	Reconfiguration of the module in progress	%MW _{r.m.c.0.15}

NOTE: If the module is not present or is disconnected, exchanges using explicit objects (READ_STS, for example) are not sent to the processor (STS_IN_PROG (%MW_{r.m.c.0.0}) = 0), but the words are refreshed.

Explicit Exchange Report: EXCH_RPT

The table below shows the EXCH_RPT (%MWr.m.c.1) word report bits:

Standard symbol	Type	Access	Meaning	Address
STS_ERR	BOOL	R	Detected error reading channel status words (1 = Detected failure)	%MWr.m.c.1.0
CMD_ERR	BOOL	R	Detected error during a command parameter exchange (1 = Detected failure)	%MWr.m.c.1.1
ADJ_ERR	BOOL	R	Interruptions while exchanging adjustment parameters (1 = Detected failure)	%MWr.m.c.1.2
RECONF_ERR	BOOL	R	Interruptions during reconfiguration of the channel (1 = Detected failure)	%MWr.m.c.1.15

Section 7.2

General Language Objects and IODDTs for Communication Protocols

Subject of this Section

This section presents the general language objects and IODDTs that apply to all communication protocols except Fipio and Ethernet.

What Is in This Section?

This section contains the following topics:

Topic	Page
Details of IODDT Implicit Exchange Objects of Type T_COM_STS_GEN	125
Details of IODDT Explicit Exchange Objects of Type T_COM_STS_GEN	126

Details of IODDT Implicit Exchange Objects of Type T_COM_STS_GEN

At a Glance

The following table presents the IODDT implicit exchange objects of type T_COM_STS_GEN applicable to all communication protocols except Fipio.

Error bit

The table below presents the meaning of the CH_ERROR error bit (%I.r.m.c.ERR):

Standard symbol	Type	Access	Meaning	Address
CH_ERROR	EBOOL	R	Communication channel error bit.	%I.r.m.c.ERR

Details of IODDT Explicit Exchange Objects of Type T_COM_STS_GEN

At a Glance

This section presents the T_COM_STS_GEN type IODDT explicit exchange objects applicable to all communication protocols except Fipio and ethernet. It includes the word type objects whose bits have a specific meaning. These objects are described in detail below.

In this part, the IODDT_VAR1 variable is of type T_COM_STS_GEN.

Observations

In general, the meaning of the bits is given for bit status 1. In specific cases, each bit status is explained.

Not all bits are used.

Explicit Exchange Execution Flags: EXCH_STS

The table below shows the meaning of channel exchange control bits from the EXCH_STS channel (%MWr.m.c.0):

Standard symbol	Type	Access	Meaning	Address
STS_IN_PROGR	BOOL	R	Read channel status words in progress.	%MWr.m.c.0.0
CMD_IN_PROGR	BOOL	R	Command parameter exchange in progress.	%MWr.m.c.0.1

Explicit Exchange Report: EXCH_RPT

The table below presents the meaning of the EXCH_RPT exchange report bits (%MWr.m.c.1):

Standard symbol	Type	Access	Meaning	Address
STS_ERR	BOOL	R	Detected read error for channel status words.	%MWr.m.c.1.0
CMD_ERR	BOOL	R	Detected error during command parameter exchange.	%MWr.m.c.1.1

Standard Channel Faults: CH_FLT

The table below shows the meaning of the bits of the status word CH_FLT (%MWr.m.c.2):

Standard symbol	Type	Access	Meaning	Address
NO_DEVICE	BOOL	R	No devices are working on the channel.	%MWr.m.c.2.0
ONE_DEVICE_FLT	BOOL	R	A device on the channel is inoperating.	%MWr.m.c.2.1
BLK	BOOL	R	Terminal block is not connected.	%MWr.m.c.2.2
TO_ERR	BOOL	R	Time out overtaken (analysis needed).	%MWr.m.c.2.3
INTERNAL_FLT	BOOL	R	Detected internal error or channel self-testing.	%MWr.m.c.2.4
CONF_FLT	BOOL	R	Different hardware and software configurations.	%MWr.m.c.2.5
COM_FLT	BOOL	R	Communication analysis needed with the channel.	%MWr.m.c.2.6
APPLI_FLT	BOOL	R	Application detected error (adjustment or configuration).	%MWr.m.c.2.7

Reading is performed by the READ_STS (IODDT_VAR1) instruction .

Section 7.3

Language Objects and IODDTs Associated with Modbus Communication

Subject of this Section

This section presents the language objects and IODDTs associated with Modbus communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
Details concerning Explicit Exchange Language Objects for a Modbus Function	129
Details of the IODDTs Implicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT	130
Details of the IODDTs Explicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT	131
Details of language objects associated with configuration Modbus mode	134

Details concerning Explicit Exchange Language Objects for a Modbus Function

At a Glance

The table below shows the language objects for Modbus communications in master or slave mode. These objects are not integrated into the IODDTs.

List of Explicit Exchange Objects in Master or Slave mode

The table below shows the explicit exchange objects:

Address	Type	Access	Meaning
%MWr.m.c.4	INT	R	Number of responses received correctly.
%MWr.m.c.5	INT	R	Number of responses received with CRC error.
%MWr.m.c.6	INT	R	Number of responses received with an exception code in slave mode.
%MWr.m.c.7	INT	R	Number of messages sent in slave mode.
%MWr.m.c.8	INT	R	Number of messages sent without response in slave mode.
%MWr.m.c.9	INT	R	Number of responses received with a negative acknowledgement.
%MWr.m.c.10	INT	R	Number of messages repeated in slave mode.
%MWr.m.c.11	INT	R	Number of detected character errors.
%MWr.m.c.24.0	BOOL	RW	Reset of detected error counters.

Details of the IODDTs Implicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT

At a Glance

The tables below show the implicit exchange objects of the IODDTs of types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT that are applicable to Modbus serial communications. They differ in terms of **configuration objects availability** (*see page 133*).

CH_ERROR bit

The following table shows the meaning of the error bit CH_ERROR (%Ir.m.c.ERR):

Standard symbol	Type	Access	Meaning	Address
CH_ERROR	EBOOL	R	Communication channel detected error bit	%Ir.m.c.ERR

Word object in Modbus Master Mode

The table below shows the meaning of the bit of the INPUT_SIGNALS word (%IW.r.m.c.0):

Standard symbol	Type	Access	Meaning	Address
DCD	BOOL	R	Data carrier detect RS232 signal (only applicable to BMX NOM 0200 module)	%IW.r.m.c.0.0
CTS	BOOL	R	Clear to send RS232 signal	%IW.r.m.c.0.2
DSR	BOOL	R	Data set ready RS232 signal (only applicable to BMX NOM 0200 module)	%IW.r.m.c.0.3

NOTE: %IW.r.m.c.0.2 is at 1 when the voltage on CTS signal is positive. It is also applicable to DCD and DSR.

Word object in Modbus Slave Mode

The language objects are identical to those of the Modbus master function. Only the objects in the following table differ.

The table below shows the meaning of the bit of the INPUT_SIGNALS word (%IW.r.m.c.0):

Standard symbol	Type	Access	Meaning	Address
LISTEN_ONLY	BOOL	R	Listen only mode	%IW.r.m.c.0.8

Details of the IODDTs Explicit Exchange Objects of Types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT

At a Glance

This part presents the explicit exchange objects of the IODDTs of types T_COM_MB_BMX and T_COM_MB_BMX_CONF_EXT that are applicable to Modbus serial and differ in terms of **configuration objects availability** (*see page 133*). It includes the word type objects whose bits have a specific meaning. These objects are described in detail below.

In this part, the IODDT_VAR1 variable is of the T_COM_STS_GEN type.

Observations

In general, the meaning of the bits is given for bit status 1. In specific cases, each bit status is explained.

Not all bits are used.

Explicit Exchange Execution Flags: EXCH_STS

The following table shows the meanings of the exchange control bits of the EXCH_STS channel (%MWr.m.c.0):

Standard symbol	Type	Access	Meaning	Address
STS_IN_PROGR	BOOL	R	Reading of channel status words in progress.	%MWr.m.c.0.0
CMD_IN_PROGR	BOOL	R	Command parameter exchange in progress.	%MWr.m.c.0.1
ADJ_IN_PROGR	BOOL	R	Adjustment parameter exchange in progress (not applicable to the BMX NOM 0200 module).	%MWr.m.c.0.2

Explicit Exchange Report: EXCH_RPT

The table below presents the various meanings of the EXCH_RPT exchange report bits (%MWr.m.c.1):

Standard symbol	Type	Access	Meaning	Address
STS_ERR	BOOL	R	Detected read error for channel status words.	%MWr.m.c.1.0
CMD_ERR	BOOL	R	Anomaly during command parameter exchange.	%MWr.m.c.1.1
ADJ_ERR	BOOL	R	Anomaly while exchanging adjustment parameters (not applicable to the BMX NOM 0200 module).	%MWr.m.c.1.2

Standard Channel Detected Faults: CH_FLT

The following table explains the various meanings of the CH_FLT status word bits (%MWr.m.c.2):

Standard symbol	Type	Access	Meaning	Address
NO_DEVICE	BOOL	R	No devices are working on the channel.	%MWr.m.c.2.0
ONE_DEVICE_FLT	BOOL	R	A device on the channel is inoperating.	%MWr.m.c.2.1
BLK	BOOL	R	Terminal block is not connected.	%MWr.m.c.2.2
TO_ERR	BOOL	R	Time out overtaken (analysis needed).	%MWr.m.c.2.3
INTERNAL_FLT	BOOL	R	Internal detected error or channel self-testing.	%MWr.m.c.2.4
CONF_FLT	BOOL	R	Different hardware and software configurations.	%MWr.m.c.2.5
COM_FLT	BOOL	R	Communication analysis needed with the channel.	%MWr.m.c.2.6
APPLI_FLT	BOOL	R	Application detected error (adjustment or configuration error).	%MWr.m.c.2.7

Reading is performed by the READ_STS instruction (IODDT_VAR1).

Specific channel status: %MWr.m.c.3

The table below shows the various meanings of the bits of the PROTOCOL channel status word (%MWr.m.c.3):

Standard symbol	Type	Access	Meaning	Address
PROTOCOL	INT	R	Byte 0 = 16#06 for Modbus Master mode. Byte 0 = 16#07 for Modbus Slave mode. Byte 0 = 16#03 for Character mode.	%MWr.m.c.3

Reading is performed by the READ_STS (IODDT_VAR1) instruction.

Channel command: %MWr.m.c.24

The table below shows the various meanings of the bits of the CONTROL (%MWr.m.c.24) word:

Standard symbol	Type	Access	Meaning	Address
DTR_ON	BOOL	R/W	Set the Data Terminal Ready signal.	%MWr.m.c.24.8
DTR_OFF	BOOL	R/W	Reset the Data Terminal Ready signal.	%MWr.m.c.24.9
TO_MODBUS_MASTER	BOOL	R/W	Change from Character mode or Modbus Slave mode to Modbus Master mode.	%MWr.m.c.24.12
TO_MODBUS_SLAVE	BOOL	R/W	Change from Character mode or Modbus Master mode to Modbus Slave mode.	%MWr.m.c.24.13
TO_CHAR_MODE	BOOL	R/W	Change from Modbus to Character Mode.	%MWr.m.c.24.14

The command is carried out with the `WRITE_CMD (IODDT_VAR1)` instruction.

For further information about how to change protocols, you can refer to **protocol changes** (*see page 151*).

External Configuration Objects of Type `T_COM_MB_BMX_CONF_EXT: %MWr.m.c.24.7` and `%MWr.m.c.25`

The table below shows the meaning of the `CONTROL (%MWr.m.c.24.7)` bit and of the `CONTROL_DATA (%MWr.m.c.25)` word that are specifically intended for the BMX NOM 0200 module programming:

Standard symbol	Type	Access	Meaning	Address
<code>SAVE_SLAVE_ADDR</code>	BOOL	R/W	Save the control data into the FLASH memory	<code>%MWr.m.c.24.7</code>
<code>SLAVE_ADDR</code>	INT	R/W	Modbus slave address to store in the FLASH memory, from 0 to 248 (0 for Master). NOTE: Be aware that this functionality is optional and there is no reason to use it intensively. As the technology involved is a FLASH, it may damage the chip.	<code>%MWr.m.c.25</code>

Details of language objects associated with configuration Modbus mode

At a Glance

The following tables present all configuration language objects for communication Modbus mode. These objects are not integrated in the IODDTs, and may be displayed by the application program.

List of explicit exchange objects for Master mode

The table below shows the explicit exchange objects.

Address	Type	Access	Meaning
%KWr.m.c.0	INT	R	The byte 0 of this word corresponds to the type: <ul style="list-style-type: none"> ● Value 6 corresponds to Master ● Value 7 corresponds to Slave
%KWr.m.c.1	INT	R	The byte 0 of this word corresponds to the transmission speed. This byte can take several values: <ul style="list-style-type: none"> ● Value -2 (0xFE) corresponds to 300 bits/s ● Value -1 (0xFF) corresponds to 600 bits/s ● Value 0 (0x00) corresponds to 1200 bits/s ● Value 1 (0x01) corresponds to 2400 bits/s ● Value 2 (0x02) corresponds to 4800 bits/s ● Value 3 (0x03) corresponds to 9600 bits/s ● Value 4 (0x04) corresponds to 19200 bits/s (default value) ● Value 5 (0x05) corresponds to 38400 bits/s ● Value 6 (0x06) corresponds to 57600 bits/s (applicable to BMX NOM 0200 module only) ● Value 7 (0x07) corresponds to 115200 bits/s (applicable to BMX NOM 0200 module only) <p>The byte 1 of this word corresponds to the format:</p> <ul style="list-style-type: none"> ● Bit 8: number of bits (1 = 8 bits (RTU), 0 = 7 bits (ASCII)) ● bit 9 = 1: parity management (1 = with, 0 = without) ● Bit 10: parity Type (1 = odd, 0 = even) ● Bit 11: number of stop bits (1 = 1 bit, 0 = 2 bits) ● Bit 13: physical line (1 = RS232, 0 = RS485) ● Bit 14: DTR/DSR/DCD modem signals (applicable to BMX NOM 0200 module only and for RS232 physical line only). If this bit is set to 1, modem signals are managed. ● Bit 15 : RTS/CTS hardware flow management signals. If RS232 is selected this bit can take 2 different values: 0 for RX/TX and 1 for RX/TX + RTS/CTS. If RS485 is selected the default value is 0 and corresponds to RX/TX.
%KWr.m.c.2	INT	R	Delay between frames (in RTU mode only): value in ms from 2 to 10000 ms (depends on the transmission speed and format selected). Its default value is 2 ms if the default box is checked. 10 s corresponds to infinite wait.

Address	Type	Access	Meaning
%KWr.m.c.3	INT	R	In Modbus Master Mode this object corresponds to the answer delay in ms from 10 ms to 1000 ms. 100 ms is the value by default. 10 s corresponds to infinite wait.
%KWr.m.c.4	INT	R	Only available in Modbus Master mode. Byte 0 of this word is the number of retries from 0 to 15. The value by default is 3.
%KWr.m.c.5	INT	R	If RS232 is selected this word corresponds to RTS/CTS delay time in hundreds of ms from 0 to 100. If RS485 is selected the default value is 0.

List of explicit exchange objects for Slave mode

The language objects for the Modbus slave function are identical to those of the Modbus master function. The only difference is for the following objects:.

Address	Type	Access	Meaning
%KWr.m.c.3	INT	R	In Modbus Slave Mode the byte 0 of this object corresponds to the slave number [0/1, 247]. For the BMX NOM 0200 module, the value 0 means that the slave number is coded in the FLASH memory
%KWr.m.c.4	INT	R	Used only in Modbus Master mode.

Section 7.4

Language Objects and IODDTs associated with Character Mode Communication

Subject of this Section

This section presents the language objects and IODDTs associated with Character Mode communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
Details concerning Explicit Exchange Language Objects for Communication in Character Mode	137
Details of IODDT Implicit Exchange Objects of Type T_COM_CHAR_BMX	138
Details of IODDT Explicit Exchange Objects of Type T_COM_CHAR_BMX	139
Details of language objects associated with configuration in Character mode	142

Details concerning Explicit Exchange Language Objects for Communication in Character Mode

At a Glance

The following tables show all configuration language objects for communication in Character Mode. These objects are not integrated into the IODDTs.

List of Explicit Exchange Objects

The table below shows the explicit exchange objects:

Address	Type	Access	Meaning
%MWr.m.c.4	INT	R	Anomaly in transmitted characters.
%MWr.m.c.5	INT	R	Anomaly in received characters.
%MWr.m.c.24.0	BOOL	RW	Resets error counters when it is set to 1.
%QWr.m.c.0 = 16#DEAD	INT	RW	Reboot the BMX NOM 0200.

Details of IODDT Implicit Exchange Objects of Type T_COM_CHAR_BMX

At a Glance

The tables below show the implicit exchange objects of the IODDT of the T_COM_CHAR_BMX type that are applicable to Character Mode communication.

Error bit

The following table shows the meaning of the error bit CH_ERROR (%Ir.m.c.ERR):

Standard symbol	Type	Access	Meaning	Address
CH_ERROR	EBOOL	R	Communication channel error bit.	%Ir.m.c.ERR

Signal object on input

The table below shows the meaning of the bit of the INPUT_SIGNALS word (%IW.r.m.c.0):

Standard symbol	Type	Access	Meaning	Address
DCD	BOOL	R	Data Carrier Detect RS232 signal (applicable to BMX NOM 0200 module only).	%IW.r.m.c.0.0
CTS	BOOL	R	Clear to send RS232 signal.	%IW.r.m.c.0.2
DSR	BOOL	R	Data Set ready RS232 signal (applicable to BMX NOM 0200 module only).	%IW.r.m.c.0.3

NOTE: %IW.r.m.c.0.2 is at 1 when the voltage on the CTS signal is positive. It is also applicable to DCD and DSR.

Details of IODDT Explicit Exchange Objects of Type T_COM_CHAR_BMX

At a Glance

This part presents the explicit exchange objects of the IODDT of the T_COM_CHAR_BMX type that are applicable to Character Mode communication. It includes the word type objects whose bits have a specific meaning. These objects are described in detail below.

In this part, the IODDT_VAR1 variable is of the T_COM_STS_GEN type.

Observations

In general, the meaning of the bits is given for bit status 1. In specific cases, each bit status is explained.

Not all bits are used.

Explicit Exchange Execution Flag: EXCH_STS

The following table shows the meanings of the exchange control bits of the EXCH_STS channel (%MWr.m.c.0):

Standard symbol	Type	Access	Meaning	Address
STS_IN_PROGR	BOOL	R	Read channel status words in progress.	%MWr.m.c.0.0
CMD_IN_PROGR	BOOL	R	Command parameter exchange in progress.	%MWr.m.c.0.1
ADJ_IN_PROGR	BOOL	R	Adjustment parameter exchange in progress (not applicable to BMX NOM 0200 module).	%MWr.m.c.0.2

Explicit Exchange Report: EXCH_RPT

The table below presents the meaning of the EXCH_RPT exchange report bits (%MWr.m.c.1):

Standard symbol	Type	Access	Meaning	Address
STS_ERR	BOOL	R	Detected read error for channel status words.	%MWr.m.c.1.0
CMD_ERR	BOOL	R	Anomaly during command parameter exchange.	%MWr.m.c.1.1
ADJ_ERR	BOOL	R	Anomaly while exchanging adjustment parameters (not applicable to the BMX NOM 0200 module).	%MWr.m.c.1.2

Standard Channel Detected Faults, CH_FLT

The following table explains the various meanings of the CH_FLT status word bits (%MWr.m.c.2) :

Standard symbol	Type	Access	Meaning	Address
NO_DEVICE	BOOL	R	No device is working on the channel.	%MWr.m.c.2.0
ONE_DEVICE_FLT	BOOL	R	A device on the channel is inoperating.	%MWr.m.c.2.1
BLK	BOOL	R	Terminal block is not connected.	%MWr.m.c.2.2
TO_ERR	BOOL	R	Time out overtaken (analysis needed).	%MWr.m.c.2.3
INTERNAL_FLT	BOOL	R	Internal detected error or channel self-testing.	%MWr.m.c.2.4
CONF_FLT	BOOL	R	Different hardware and software configurations.	%MWr.m.c.2.5
COM_FLT	BOOL	R	Communication analysis is needed with the PLC.	%MWr.m.c.2.6
APPLI_FLT	BOOL	R	Application detected error (adjustment or configuration error).	%MWr.m.c.2.7

Reading is performed by the READ_STS instruction (IODDT_VAR1).

Specific Channel Status, %MWr.m.c.3

The table below shows the various meanings of the bits of the PROTOCOL (%MWr.m.c.3) channel status word:

Standard symbol	Type	Access	Meaning	Address
PROTOCOL	INT	R	Byte 0 = 16#03 for Character Mode function.	%MWr.m.c.3

Reading is performed by the READ_STS (IODDT_VAR1) instruction.

%MWr.m.c.24 Channel Command

The table below shows the various meanings of the bits of the CONTROL (%MWr.m.c.24) word:

Standard symbol	Type	Access	Meaning	Address
DTR_ON	BOOL	R/W	Set the Data Terminal Ready signal.	%MWr.m.c.24.8
DTR_OFF	BOOL	R/W	Reset the Data Terminal Ready signal.	%MWr.m.c.24.9

The command is carried out with the WRITE_CMD (IODDT_VAR1) instruction.

For further information about how to change protocols, you can refer to protocol changes ([see page 151](#)).

%QWr.m.c.0 Word Object

The table below shows the meaning of the bit 0 of %QWr.m.c.0 word:

Standard symbol	Type	Access	Meaning	Address
STOP_EXCH	BOOL	R/W	Stop all exchanges on rising edge (available on the BMX NOM 0200 module only).	%QWr.m.c.0.0

Details of language objects associated with configuration in Character mode

At a Glance

The following tables present all configuration language objects for communication Character mode. These objects are not integrated in the IODDTs, and may be displayed by the application program.

List of explicit exchange objects for Character mode

The table below shows the explicit exchange objects.

Address	Type	Access	Meaning
%KWr.m.c.0	INT	R	The byte 0 of this word corresponds to the type. Value 3 corresponds to Character Mode.
%KWr.m.c.1	INT	R	<p>The byte 0 of this word corresponds to the transmission speed. This byte can take several values:</p> <ul style="list-style-type: none"> ● Value -2 (0xFE) corresponds to 300 bits/s ● Value -1 (0xFF) corresponds to 600 bits/s ● Value 0 (0x00) corresponds to 1200 bits/s ● Value 1 (0x01) corresponds to 2400 bits/s ● Value 2 (0x02) corresponds to 4800 bits/s ● Value 3 (0x03) corresponds to 9600 bits/s (default value) ● Value 4 (0x04) corresponds to 19200 bits/s ● Value 5 (0x05) corresponds to 38400 bits/s ● Value 6 (0x06) corresponds to 57600 bits/s (can be taken only for BMX NOM 0200 module) ● Value 7 (0x07) corresponds to 115200 bits/s (can be taken only for BMX NOM 0200 module) <p>The byte 1 of this word corresponds to the format:</p> <ul style="list-style-type: none"> ● Bit 8: number of bits (1 = 8 bits (RTU), 0 = 7 bits (ASCII)) ● bit 9 = 1: parity management (1 = with, 0 = without) ● Bit 10: parity Type (1 = odd, 0 = even) ● Bit 11: number of stop bits (1 = 1 bit, 0 = 2 bits) ● Bit 13: physical line (1 = RS232, 0 = RS485) ● Bit 14: DTR/DSR/DCD modem signals. For BMX NOM 0200 module and if RS232 is selected, this bit can take 2 different values: 1 means that modem signals are managed, 0 means that they are not (default value for BMX P34 or if RS485 is selected) ● Bit 15 : RTS/CTS hardware flow management signals. If RS232 is selected this bit can take 2 different values: 0 for RX/TX and 1 for RX/TX + RTS/CTS. If RS485 is selected the default value is 0 and corresponds to RX/TX
%KWr.m.c.2	INT	R	Entered value in ms of stop on silence (depends on the transmission speed and format selected). Value 0 means no silence detection.

Address	Type	Access	Meaning
%KWr.m.c.3	INT	R	<p>This word corresponds to the polarization type:</p> <ul style="list-style-type: none"> Value 0 on both bit 14 and bit 15 corresponds to no polarization (This is the default value for BMX P34 or if RS232 is selected) Bit 14: value 1 corresponds to low impedance (Modbus like) polarization and can be taken only for BMX NOM 0200 module and if RS485 is selected Bit 15: value 1 corresponds to high impedance polarization and can be taken only for BMX NOM 0200 module and if RS485 is selected
%KWr.m.c.5	INT	R	<p>This word corresponds to RTS/CTS delay time in hundreds of ms from 0 to 100 if RS232 is selected. If RS485 is selected the default value is 0.</p>
%KWr.m.c.6	INT	R	<p>Bit 0 of Byte 0 can have 2 values:</p> <ul style="list-style-type: none"> value 1 corresponds to the stop checkbox in the Stop on reception area for character 1 when checked value 0 corresponds to the stop checkbox in the Stop on reception area for character 1 when unchecked <p>Bit 1 of Byte 0 can have 2 values:</p> <ul style="list-style-type: none"> value 1 corresponds to the Character Included checkbox in the Stop on reception area for character 1 when checked value 0 corresponds to the Character Included checkbox in the Stop on reception area for character 1 when unchecked <p>Byte 1 of this word corresponds to the entered value of stop on reception of character 1 from 0 to 255.</p>
%KWr.m.c.7	INT	R	<p>Bit 0 of Byte 0 can have 2 values:</p> <ul style="list-style-type: none"> value 1 corresponds to the stop checkbox in the Stop on reception area for character 2 when checked value 0 corresponds to the stop checkbox in the Stop on reception area for character 2 when unchecked <p>Bit 1 of Byte 0 can have 2 values:</p> <ul style="list-style-type: none"> value 1 corresponds to the Character Included checkbox in the Stop on reception area for character 2 when checked value 0 corresponds to the Character Included checkbox in the Stop on reception area for character 2 when unchecked <p>Byte 1 of this word corresponds to the entered value of stop on reception of character 2 from 0 to 255.</p>

Section 7.5

The IODDT Type T_GEN_MOD Applicable to All Modules

Details of the Language Objects of the IODDT of Type T_GEN_MOD

Introduction

The Modicon X80 modules have an associated IODDT of type T_GEN_MOD.

Observations

In general, the meaning of the bits is given for bit status 1. In specific cases an explanation is given for each status of the bit.

Some bits are not used.

List of Objects

The table below presents the objects of the IODDT.

Standard Symbol	Type	Access	Meaning	Address
MOD_ERROR	BOOL	R	Module detected error bit	%I.r.m.MOD.ERR
EXCH_STS	INT	R	Module exchange control word	%MWr.m.MOD.0
STS_IN_PROGR	BOOL	R	Reading of status words of the module in progress	%MWr.m.MOD.0.0
EXCH_RPT	INT	R	Exchange report word	%MWr.m.MOD.1
STS_ERR	BOOL	R	Event when reading module status words	%MWr.m.MOD.1.0
MOD_FLT	INT	R	Internal detected errors word of the module	%MWr.m.MOD.2
MOD_FAIL	BOOL	R	module inoperative	%MWr.m.MOD.2.0
CH_FLT	BOOL	R	Inoperative channel(s)	%MWr.m.MOD.2.1
BLK	BOOL	R	Terminal block incorrectly wired	%MWr.m.MOD.2.2
CONF_FLT	BOOL	R	Hardware or software configuration anomaly	%MWr.m.MOD.2.5
NO_MOD	BOOL	R	Module missing or inoperative	%MWr.m.MOD.2.6
EXT_MOD_FLT	BOOL	R	Internal detected errors word of the module (Fipio extension only)	%MWr.m.MOD.2.7
MOD_FAIL_EXT	BOOL	R	Internal detected error, module unserviceable (Fipio extension only)	%MWr.m.MOD.2.8
CH_FLT_EXT	BOOL	R	Inoperative channel(s) (Fipio extension only)	%MWr.m.MOD.2.9
BLK_EXT	BOOL	R	Terminal block incorrectly wired (Fipio extension only)	%MWr.m.MOD.2.10

Standard Symbol	Type	Access	Meaning	Address
CONF_FLT_EXT	BOOL	R	Hardware or software configuration anomaly (Fipio extension only)	%MWr.m.MOD.2.13
NO_MOD_EXT	BOOL	R	Module missing or inoperative (Fipio extension only)	%MWr.m.MOD.2.14

Section 7.6

Language Objects and Device DDTs Associated with Modbus Communication

Subject of this Section

This section presents the language objects and Device DDTs associated with Modbus communication.

What Is in This Section?

This section contains the following topics:

Topic	Page
BMX NOM 0200.x Device DDT	147
MOD_FLT Byte Description	150

BMX NOM 0200.x Device DDT

Introduction

This topic describes the Control Expert **NOM Device DDT**, the instance default naming is described in Device DDT Instance Naming Rule (*see EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual*).

Regarding the device DDT, its name contains the following information:

- platform with:
 - M for Modicon X80 module
- device type (COM for communication)
- function (NOM for BMX NOM 0200.x)
- direction:
 - IN
 - OUT

List of Implicit Device DDT

The following table shows the list of device DDT and their **X80** modules:

Device DDT	Modicon X80 modules
T_M_COM_NOM	BMX NOM 0200.x

Implicit Device DDT Description

The following table shows the T_M_COM_NOM status word bits:

Standard Symbol	Type	Meaning	Access
MOD_HEALTH	BOOL	0 = the module has a detected error 1 = the module is operating correctly	read
MOD_FLT	BYTE	internal detected errors byte (<i>see page 150</i>) of the module	read
COM_CH	ARRAY [0...1] of T_M_COM_NOM_CH	array of structure	

The following table shows the T_M_COM_NOM_CH[0...1] status word bits:

Standard Symbol		Type	Bit	Meaning	Access
FCT_TYPE		WORD		0 = Channel is not used	read
				3 = Character mode	
				6 = MODBUS Master	
				7 = MODBUS Slave	
CH_HEALTH		BOOL		0 = the channel has a detected error	read
				1 = the channel is operating correctly	
INPUT_SIGNALS [INT]	DCD	BOOL	0	Data Carrier Detect RS-232 signal (applicable to BMX NOM 0200 module only)	read
	CTS	BOOL	2	clear to send RS-232 signal	read
	DSR	BOOL	3	Data Set ready RS-232 signal (applicable to BMX NOM 0200 module only)	read
COMMAND [INT]	STOP_EXCH	BOOL	0	rising edge at 1: All exchanges in progress are stopped.	read / write

Explicit Device DDT Instances Description

Explicit exchanges (Read Status) - only applicable to Modicon X80 I/O channels - are managed with READ_STS_MX (Modicon M580) or READ_STS_QX (Modicon Quantum) EFB instances.

- Targeted channel address (ADDR) can be managed with ADDMX (*see EcoStruxure™ Control Expert, Communication, Block Library*) EF (connect ADDMX OUT to ADDR)
- READ_STS_MX (*see EcoStruxure™ Control Expert, I/O Management, Block Library*) or READ_STS_QX (*see EcoStruxure™ Control Expert, I/O Management, Block Library*) output parameter (STS) can be connected to a "T_M_XXX_YYY_CH_STS" DDT instance (variable to be created manually), where:
 - xxx represents the device type
 - yyy represents the function

Example: T_M_COM_NOM_CH_STS

The following table shows the T_M_COM_NOM_CH_STS status word bits:

Type	Type	Access
STRUCT	T_M_COM_NOM_CH_STS	

The following table shows the T_M_COM_NOM_CH_STS structure status word bits:

Standard Symbol	Type	Bit	Meaning	Access	
CH_FLT [INT]	NO_DEVICE	BOOL	0	no device is working on the channel	read
	ONE_DEVICE_FLT	BOOL	1	inoperable device on the channel	read
	BLK	BOOL	2	terminal block fault detected (not connected)	read
	TO_ERR	BOOL	3	time out detected error (defective wiring)	read
	INTERNAL_FLT	BOOL	4	internal detected error or channel self-testing	read
	CONF_FLT	BOOL	5	configuration detected fault: different hardware and software configurations	read
	COM_FLT	BOOL	6	problem communicating with the PLC	read
	APPLI_FLT	BOOL	7	application detected error (adjustment or configuration detected error)	read
PROTOCOL	BYTE		6 for Modbus Master, 3 for character mode	read	
ADDRESS	BYTE		slave address	read	

MOD_FLT Byte Description

MOD_FLT Byte in Device DDT

MOD_FLT byte structure:

Bit	Symbol	Description
0	MOD_FAIL	<ul style="list-style-type: none"> ● 1: Internal detected error or module failure detected. ● 0: No detected error
1	CH_FLT	<ul style="list-style-type: none"> ● 1: Inoperative channels. ● 0: Channels are operative.
2	BLK	<ul style="list-style-type: none"> ● 1: Terminal block detected error. ● 0: No detected error. <p>NOTE: This bit may not be managed.</p>
3	–	<ul style="list-style-type: none"> ● 1: Module in self-test. ● 0: Module not in self-test. <p>NOTE: This bit may not be managed.</p>
4	–	Not used.
5	CONF_FLT	<ul style="list-style-type: none"> ● 1: Hardware or software configuration detected error. ● 0: No detected error.
6	NO_MOD	<ul style="list-style-type: none"> ● 1: Module is missing or inoperative. ● 0: Module is operating. <p>NOTE: This bit is managed only by modules located in a remote rack with a BME CRA 312 10 adapter module. Modules located in the local rack do not manage this bit that remains at 0.</p>
7	–	Not used.

Chapter 8

Dynamic Protocol Switching

Changing Protocol with the BMXNOM0200 Module

General

This part describes how to change the protocol used by a BMXNOM0200 serial communication using the `WRITE_CMD (IODDT_VAR1)` command.

This command can be used to switch between the following three protocols:

- Modbus Slave
- Modbus Master
- Character Mode

NOTE: `IODDT_VAR1` variable must be either a `T_COM_MB_BMX` or a `T_COM_MB_BMX_CONF_EXT` type.

Changing Protocol: The Principle

You must create first an `IODDT` variable linked to the serial channel, then set to 1 the bit of word `IODDT_VAR1.CONTROL (%MWr.m.c.24)` that corresponds to the change of protocol desired:

- `TO_MODEBUS_MASTER` (Bit 12): Current protocol is changed to Modbus Master.
- `TO_MODEBUS_SLAVE` (Bit 13): Current protocol is changed to Modbus Slave.
- `TO_CHAR_MODE` (Bit 14): Current protocol is changed to Character Mode.

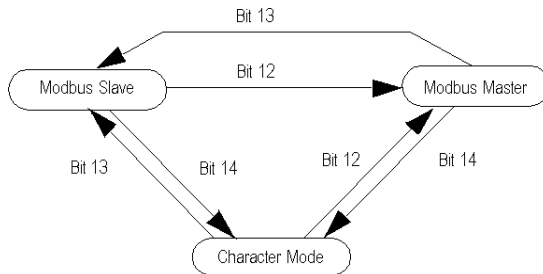
NOTE: A single bit can be set to 1 at a time: setting several bits to 1 will result in an error.

NOTE: `IODDT_VAR1.CONTROL (%MWr.m.c.24)` is part of the `IODDT` variable `IODDT_VAR1`.

Afterwards, apply the `WRITE_CMD` instruction to the `IODDT` variable linked to the serial channel.

NOTE: Be careful that two masters (on the same bus) do not send requests simultaneously otherwise the requests are lost and each report will have a bad result which could be `16#0100` (request could not be processed) or `16#ODFF` (slave is not present).

The diagram below shows the protocol changes to be made according to the bits of the IODDT_VAR1.CONTROL (%MWr.m.c.24) word set to 1:



Uses

Three protocol changes are used:

- **Transfer from Modbus Slave to Modbus Master:**
The aim of Modbus Master configuration is to send information about an event to another PLC. When a change is made from Modbus Slave configuration to Modbus Master configuration, transmission, signal and physical line parameters remain the same. Only the values of the following parameters specific to Modbus Master configuration are changed:
 - The Delay Between Frames is set to its default value, which depends on transmission speed.
 - Answer delay is set to 3 s
 - Number of retries set to 0
- **Transfer from Modbus Slave/Master to Character Mode**
Switching to Character Mode is used to send AT commands to a modem. When a change is made from Modbus configuration to Character Mode configuration, transmission, signal and physical line parameters remain the same. Only the message end detection parameter specific to Character Mode is set to stop on reception of the `x0d` ending character.
- **Transfer from Character Mode to Modbus Master and to Modbus Slave:**
The aim of Character Mode configuration is to communicate with a private protocol (a modem, for instance). Once the exchange has finished, the user switches to the Modbus Master configuration (with the answer delay set to 3 s and the number of retries set to 0) in order to send information about an event to another PLC. Once the message has been sent, the user returns to the Modbus Slave configuration: the slave number is set to the value stored in the FLASH memory or to 248 if none.

Cold and Warm Starts

Changes in protocol are not affected by the %S0 and %S1 bits (the bits set to 1 during a cold and warm start respectively). However, a cold or warm start of the PLC will configure the serial port to its default values or to values programmed into the application.

NOTE: The default configuration of the module is the following: to be easily configurable by a computer like a PC, the channel 0 is configured in RS232 slave mode, and the channel 1 in RS485 mode. Other parameters are: 19200 bauds, RTU, even, 1 stop bit, no flow control, 1,75 ms as default frame delay, slave number 248.

Part III

Quick Start: BMXNOM0200 as a Modbus Slave over a Quantum PLC

Overview

This part describe how to configure the BMXNOM0200 module as a Modbus RS-485 RTU slave in a Modicon X80 drop over a Quantum PLC.

The device to configure in the Control Expert **Hardware Catalog** is BMXNOM0200.4.

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
9	Overview	157
10	Configuration in Control Expert	163

Chapter 9

Overview

Prerequisites

To configure the BMXNOM0200.4, you have to:

- Use the following firmware versions:
 - BMXCRA31210: Minimum V2.14
 - BMXNOM0200: Minimum V1.5
- Interlink a Quantum 140NOC78•00 to the Quantum 140CRP31200

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Product Overview	158
Architecture Overview	159
Limitations	161

Product Overview

Presentation

The BMXNOM0200.4 is a new generic Control Expert device that you can find in the Control Expert hardware catalog within communication family.

Before adding the BMXNOM0200.4 device in the Modicon X80 drop, first you have to add a drop end communicator device. In Control Expert **Hardware Catalog**, select the device BMXCRA31210 (SV>=2.13).

Supported Protocols

For the BMXNOM0200 modules:

- channel 0 is RS232 or RS485,
- and channel 1 is only RS485.

Declaring the BMXNOM0200 module as a BMXNOM0200.4 in Control Expert allows you to configure the module as a:

- Modbus RTU slave on RS-485
- Modbus Serial RTU and ASCII Master on RS-232 and RS-485
- Character mode

Compatibility

This offer is compatible with the standard offer: BMXNOM0200, 140CRP31200, BMXCRA31210, and Quantum CPU.

Architecture Overview

Presentation

Modbus slave messages received by the BMXNOM0200.4 are transferred to the head of the drop (BMXCRA31210). Then, the head forwards the message on Ethernet I/O to the Quantum CPU.

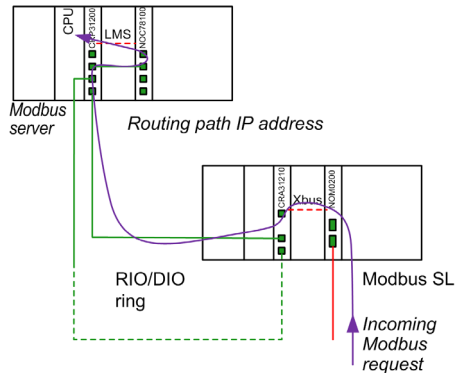
The Quantum 140CRP31200 does not treat incoming Modbus messages. You have to plug an additional 140NOC78•00 Ethernet module in the Quantum main rack and to interlink it with the CRP module.

After interlink, the drop head can send the Modbus messages to the 140NOC78•00. The 140NOC78•00 forwards the messages to the CPU.

For doing so, you must enter the IP address of the 140NOC78•00 (Modbus server routing path [\(see page 165\)](#)) in the drop end communicator module configuration (BMXCRA31210).

Illustration

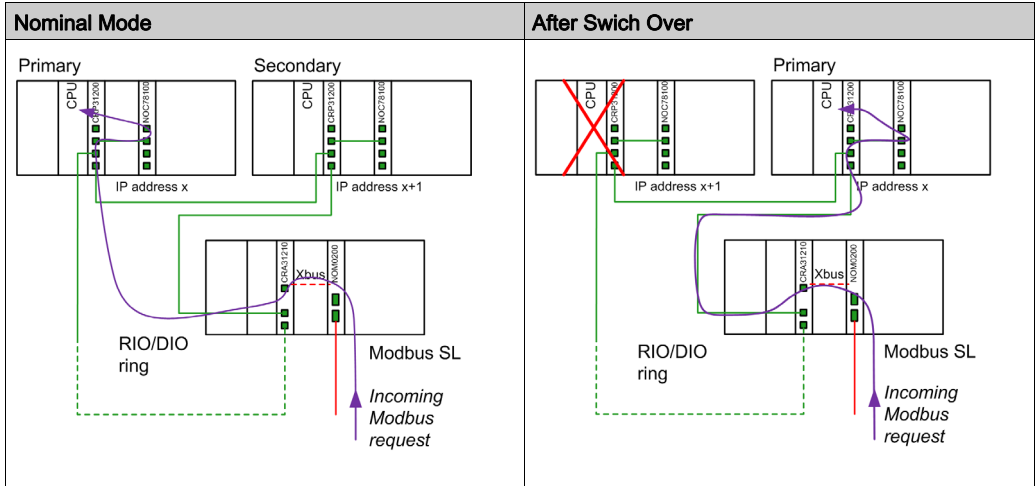
The Quantum CPU system treats the Modbus requests without any application program:



NOTE: The same path is used to route the Modbus response.

HSBY Specific Case

The 140NOC78• 00 IP address swaps in the case of PLC switch over. The Modbus requests are still forwarded to the operational CPU:



NOTE: The Modbus master application manages the repetition of requests in case of a message loss that could occur during a PLC switch over.

Limitations

Maximum Configuration

This table shows the maximum configuration of the BMXNOM0200.4:

Element	Maximum configuration
Master channel	4 per configured drop with a maximum of 36 expert channels per drop. NOTE: Each configured channel of the BMXNOM0200.4 counts for an expert channel.
Drop	4 BMXNOM0200.4 per drop.
Quantum system	16 BMXNOM0200
Modbus frame length	256 bytes

IP Address

You must configure the IP address of the Modbus routing path for each BMXCRA31210 that supports a Modbus slave BMXNOM0200.4 module.

Control Expert provides no control on the consistency of those IP addresses.

WARNING

UNINTENDED EQUIPMENT OPERATION

Check that the IP address is really the one of the Quantum that supports the Modbus server.

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

Supported Protocols

Only Modbus RTU is supported as slave protocol.

Only RS-485 is supported when Modbus slave is selected.

Supported Modbus Function Codes

This table lists the Modbus function codes (FC) supported by the Quantum server:

Binding to -> Modbus FC:	Variable type	Code	Function
01	%M	0X	Read coil status (output bit)
02	%M	1X	Read input status (input bit)
03	%MW	4X	Read holding registers
05	%M	0X	Force single coil
04	%MW	3X	Read input register
06	%MW	4X	Write single register
15	%M	0X	Write multiple coils
16	%MW	4X	Write multiple registers
23	%MW	4X	Read/write multiple registers

Chapter 10

Configuration in Control Expert

Introduction

Most of the operating modes are identical to BMXNOM0200 versions supported previously.

This chapter only details what is specific to the configuration of the BMXNOM0200.4 module in Control Expert.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Module Insertion	164
Module Configuration Screen	165

Module Insertion

Presentation

In a Quantum Ethernet I/O architecture, you can only insert the BMXNOM0200.4 modules in an EIO Modicon X80 drop, with BMXCRA31210 (SV >= 2.13) as EIO adapter module.

Procedure

Follow this procedure to insert the BMXNOM0200.4 module in a Modicon X80 remote drop:

Step	Action
1	Insert the 140CRP31200 module in the Quantum local rack.
2	Create on the EIO Bus an EIO Modicon X80 drop with a BMXCRA31210 (SV >= 2.13).
3	Insert the new BMXNOM0200.4 module in the drop.
4	Insert a 140NOC78•00 in the Quantum Local Bus .

Module Configuration Screen

Modbus Server Routing Path Configuration

This configuration is only possible in offline mode (PLC not connected).

Follow this procedure to set the Modbus server routing path:

Step	Action
1	Double-click the BMXCRA31210 module in the configurator editor.
2	Select the Cpu Modbus Server tab. <div data-bbox="321 467 1208 976" style="border: 2px solid blue; padding: 10px; margin: 10px 0;"> </div>
3	Select Enabled in the CPU modbus Server field.
4	Enter the IP address of the 140NOC78•00 in the Modbus server routing path field. The 140NOC78•00 manages the routing of the Modbus frames between Ethernet I/O and the CPU.

Control Expert provides no control on the consistency of those IP addresses.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Check that the IP address is really the one of the Quantum that supports the Modbus server.

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

NOTE: You must configure the IP address of the Modbus routing path for each BMXCRA31210 that supports a Modbus slave BMXNOM0200.4 module.

Access to Channel Configuration Screens

Follow this procedure to access the channel configuration screens of the BMXNOM0200.4 module:

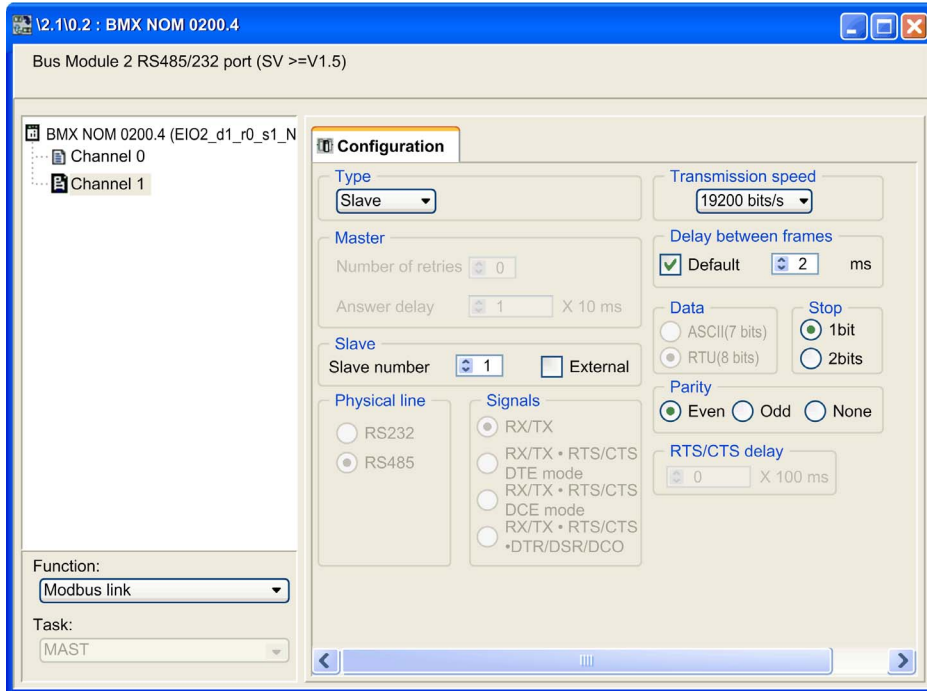
Step	Action
1	Open the BMXNOM0200.4 subdirectory in the project browser.
2	Select the channel to configure. By default: <ul style="list-style-type: none"> ● Channel 0 is configured with the Character mode link function. ● Channel 1 is configured with the Modbus link function. NOTE: Some parameters are not accessible and are grayed out.

To configure Modbus serial communication in master mode, refer to chapter *Modbus Serial Communication* (see page 53).

To configure character mode communication, refer to chapter *Character Mode Communication* (see page 87).

Slave Modbus Link Configuration Screen

This figure shows the slave configuration screen of the BMXNOM0200.4 module:



This table shows the default values of the parameters for Modbus slave configuration screen:

Configuration parameter	Default value
Type	Slave
Slave number	1
Physical line	RS-485 only
Signals	RX/TX only
Transmission speed	19200 bits/s
Delay between frames	2 ms
Data	8 bits only
Stop	1 bit
Parity	Even

NOTE: Modbus is a standard protocol. This module is based on a single mode of data exchange. When configuring Modbus serial communication in master mode, the slave parameters are grayed out and cannot be modified.



!

%I

According to the CEI standard, %I indicates a language object of type discrete IN.

%IW

According to the CEI standard, %IW indicates a language object of type analog IN.

%KW

According to the CEI standard, %KW indicates a language object of type constant word.

%M

According to the CEI standard, %M indicates a language object of type memory bit.

%MW

According to the CEI standard, %MW indicates a language object of type memory word.

%Q

According to the CEI standard, %Q indicates a language object of type discrete OUT.

%QW

According to the CEI standard, %QW indicates a language object of type analog OUT.

A

Address

On a network, the identification of a station. In a frame, a grouping of bits that identifies the frame's source or destination.

Altivar

AC variable speed drive.

ARRAY

An ARRAY is a table containing elements of a single type. The syntax is as follows: ARRAY [<limits>] OF <Type> 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.

ASCII

ASCII is the abbreviation of American Standard Code for Information Interchange. This is an American code (but which has become an international standard) that uses 7 bits to define every alphanumerical character used in English, punctuation symbols, certain graphic characters and other miscellaneous commands.

B

BOOL

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

Broadcast

Broadcast communications send packets from one station to every network destinations. Broadcast messages pertain to every network devices or only one device for which the address is not known.

BYTE

When 8 bits are grouped together, they are called a BYTE. You can enter a BYTE either in binary mode or in base 8. The BYTE type is encoded in an 8 bit format which, in hexadecimal format, ranges from 16#00 to 16#FF.

C

Configuration

The configuration gathers the data which characterizes the machine (invariant) and which is necessary for the module to operate. All this information is stored in the constant PLC %KW zone. The PLC application cannot modify them.

Control Expert

Schneider Automation PLC programming software.

CPU

CPU is the abbreviation of Central Processing Unit: generic name used for Schneider Electric processors.

CRC

CRC is the abbreviation of Cyclic Redundancy Checksum: it indicates whether no character has been "deformed" during frame transmission.

D

DFB

DFB is the abbreviation of Derived Function Block. DFB types are function blocks that can be defined by the user in ST (Structured Text), IL (Instruction List), LD (Ladder Diagram) or FBD (Function Block Diagram) language. Using these DFB types in an application makes it possible to:

- simplify the design and entry of the program;
- make the program easier to read;
- make it easier to debug;
- reduce the amount of code generated.

DINT

DINT is the abbreviation of Double INTeger (encoded in 32 bits). The upper/lower limits are as follows: $-(2 \text{ to the power of } 31)$ to $(2 \text{ to the power of } 31) - 1$. Example: -2147483648, 2147483647, 16#FFFFFFFF.

Discrete Module

Module with discrete inputs/outputs.

E**EBOOL**

EBOOL is the abbreviation of Extended BOOLean. An EBOOL type has a value (0 (FALSE) or 1 (TRUE), but also rising or falling edges and forcing functions. An EBOOL variable occupies one byte in memory. The byte contains the following information:

- one bit for the value;
- one bit for the history (whenever the object changes state, the value is copied to the history bit);
- one bit for forcing (equal to 0 if the object is not forced, or 1 if the bit is forced).

The default value of each bit is 0 (FALSE).

EF

EF is the abbreviation of Elementary Function. This is a block used in a program which performs a predefined logical function. A function does not have any information on the internal state. Several calls to the same function using the same input parameters always return the same output values. You will find information on the graphic form of the function call in the "[functional block (instance)]". Unlike a call to a function block, function calls include only an output which is not named and whose name is identical to that of the function. In FBD, each call is indicated by a unique [number] via the graphic block. This number is managed automatically and cannot be modified. You position and configure these functions in your program in order to execute your application. You can also develop other functions using the SDKC development kit.

F**FBD**

FBD is the abbreviation of Function Block Diagram. FBD is a graphical programming language that works like a flowchart. By adding simple logical blocks (AND, OR, etc.), each function or function block in the program is represented in this graphical format. For each block, the inputs are on the left and the outputs on the right. Block outputs can be linked to inputs of other blocks in order to create complex expressions.

Fipio

Field bus used to connect sensor or actuator type devices.

FLASH memory

FLASH memory is nonvolatile memory that can be overwritten. It is stored on a special EEPROM that can be erased and reprogrammed.

Frame

A frame is a group of bits that form a discrete block of information. Frames contain network control information or data. The size and composition of a frame is determined by the network technology being used.

Full duplex

A method of data transmission capable of transmitting and receiving over the same channel simultaneously.

H

Half duplex

A method of data transmission capable of communication in either of two directions, but in only one direction at a time.

Hub

A hub device connects a series of flexible and centralized modules to create a network.

I

INT

INT is the abbreviation of single INTeger (encoded in 16 bits). The upper/lower limits are as follows: -2 to the power of 15 to $(2$ to the power of 15) - 1. Example: -32768, 32767, 2#1111110001001001, 16#9FA4.

IODDT

IODDT is the abbreviation of Input/Output Derived Data Type. The term IODDT indicates a structured data type representing a module or a channel of a PLC module. Each expert module has its own IODDTs.

L

LED

LED is the abbreviation of Light emitting diode. An indicator that lights up when electricity passes through it. It indicates the operation status of a communication module.

LRC

LRC is the abbreviation of Longitudinal redundancy check: it has been devised to address the low probability of error detection of parity checking.

M

Master task

Main program task. It is obligatory and is used to carry out sequential processing of the PLC.

Momentum

I/O modules using several open standard communication networks.

N**Network**

There are two meanings of the word "network".

- In LD (Ladder Diagram): a network is a set of interconnected graphic elements. The scope of a network is local, concerning the organizational unit (section) of the program containing the network.
- With expert communication modules: a network is a set of stations that intercommunicate. The term "network" is also used to define a group interconnected graphic elements. This group then makes up part of a program that may comprise a group of networks.

P**PLC**

PLC is the abbreviation of Programmable logic controller. The PLC is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. PLCs are computers suited to survive the harsh conditions of the industrial environment.

Protocol

Describes message formats and a set of rules used by two or more devices to communicate using those formats.

R**RS232**

Serial communication standard which defines the voltage of the following service:

- a signal of +12 V indicates a logical 0,
- a signal of -12 V indicates a logical 1.

There is, however, in the case of any attenuation of the signal, detection provided up to the limits -3 V and +3 V. Between these two limits, the signal will be considered as invalid. RS232 connections are quite sensitive to interference. The standard specifies not to exceed a distance of 15 m or a maximum of 9600 bauds (bits/s).

RS485

Serial connection standard that operates in 10 V/+5 V differential. It uses two wires for send/receive. Their "3 states" outputs enable them to switch to listen mode when the transmission is terminated.

RTU

RTU is the abbreviation of Remote Terminal Unit. In RTU mode, data is sent as two four-bit, hexadecimal characters, providing for higher throughput than in ASCII mode for the same baudrate. Modbus RTU is a binary protocol and more time delay critical than the ASCII protocol.

S

Section

Program module belonging to a task which can be written in the language chosen by the programmer (FBD, LD, ST, IL, or SFC). A task can be composed of several sections, the order of execution of the sections corresponding to the order in which they are created. This order is modifiable.

SEPAM

Digital protection relay for protection, control and monitoring of power systems.

Socket

The association of a port with an IP address, serving as an identification of sender or recipient.

ST

ST is the abbreviation of Structured Text. The structured literal language is a developed language similar to computer programming languages. It can be used to organize a series of instructions.

STRING

A STRING variable is a series of ASCII characters. The maximum length of a string is 65,534 characters.

T

TAP

TAP is the abbreviation of Transmission Access Point: the bus connection unit.

Task

A group of sections and subroutines, executed cyclically or periodically for the MAST task, or periodically for the FAST task. A task possesses a level of priority and is linked to inputs and outputs of the PLC. These I/O are refreshed in consequence.

V

Variable

Memory entity of type BOOL, WORD, DWORD, etc., whose contents can be modified by the program currently running.

W

WORD

The type WORD is encoded in a 16 bit format and is used to perform processing on series of bits. This table shows the upper/lower limits of each of the bases that can be used:

Base	Lower limit	Upper limit
Hexadecimal	16#0	16#FFFF
Octal	8#0	8#177777
Binary	2#0	2#1111111111111111

Examples of representation:

Data	Representation in one of the bases
0000000011010011	16#D3
1010101010101010	8#125252
0000000011010011	2#11010011

X

XBT

Graphical operator terminal.

XPS

Safety module used for processing of safety signals to monitor both the component and the wiring of a safety system, including devices for general monitoring as well as application specific models.



B

BMXNOM0200, *17*
 limits, *48*
 presentation, *19*
BMXNOM0200.4
 Quantum PLC, *155*
BMXNOM0200H
 presentation, *19*

C

Cabling, *40*
certifications, *26*
changing protocols, *151*
channel data structure for all modules
 T_GEN_MOD, *144, 144*
channel data structure for character mode
communication
 T_COM_CHAR_BMX, *138, 139*
channel data structure for communication
protocols
 T_COM_STS_GEN, *125, 126*
channel data structure for modbus communi-
cation
 T_COM_MB_BMX, *130, 131*
connection devices, *31*

D

debugging Modbus, *83*

I

INPUT_BYTE, *100*
INPUT_CHAR, *100*
INPUT_CHAR_QX, *100*

L

limits
 BMXNOM0200, *48*

M

MOD_FLT, *150*

P

parameter settings, *115*
PRINT_CHAR, *100*
PRINT_CHAR_QX, *100*

S

standards, *26*

T

T_COM_CHAR_BMX, *138, 139*
T_COM_MB_BMX, *130, 131*
T_COM_STS_GEN, *125, 126*
T_GEN_MOD, *144, 144*
T_M_COM_NOM, *147*

W

wiring accessories, *40*

