



User Manual Nexto Xpress

MU216600 Rev. L

February 23, 2021

No part of this document may be copied or reproduced in any form without the prior written consent of Altus Sistemas de Automação S.A. who reserves the right to carry out alterations without prior advice.

According to current legislation in Brazil, the Consumer Defense Code, we are giving the following information to clients who use our products, regarding personal safety and premises.

The industrial automation equipment, manufactured by Altus, is strong and reliable due to the stringent quality control it is subjected to. However, any electronic industrial control equipment (programmable controllers, numerical commands, etc.) can damage machines or processes controlled by them when there are defective components and/or when a programming or installation error occurs. This can even put human lives at risk. The user should consider the possible consequences of the defects and should provide additional external installations for safety reasons. This concern is higher when in initial commissioning and testing.

The equipment manufactured by Altus does not directly expose the environment to hazards, since they do not issue any kind of pollutant during their use. However, concerning the disposal of equipment, it is important to point out that built-in electronics may contain materials which are harmful to nature when improperly discarded. Therefore, it is recommended that whenever discarding this type of product, it should be forwarded to recycling plants, which guarantee proper waste management.

It is essential to read and understand the product documentation, such as manuals and technical characteristics before its installation or use. The examples and figures presented in this document are solely for illustrative purposes. Due to possible upgrades and improvements that the products may present, Altus assumes no responsibility for the use of these examples and figures in real applications. They should only be used to assist user trainings and improve experience with the products and their features.

Altus warrants its equipment as described in General Conditions of Supply, attached to the commercial proposals.

Altus guarantees that their equipment works in accordance with the clear instructions contained in their manuals and/or technical characteristics, not guaranteeing the success of any particular type of application of the equipment.

Altus does not acknowledge any other guarantee, directly or implied, mainly when end customers are dealing with third-party suppliers. The requests for additional information about the supply, equipment features and/or any other Altus services must be made in writing form. Altus is not responsible for supplying information about its equipment without formal request. These products can use EtherCAT® technology (www.ethercat.org).

COPYRIGHTS

Nexto, MasterTool, Grano and WebPLC are the registered trademarks of Altus Sistemas de Automação S.A.

Windows, Windows NT and Windows Vista are registered trademarks of Microsoft Corporation.

OPEN SOURCE SOFTWARE NOTICE

To obtain the source code under GPL, LGPL, MPL and other open source licenses, that is contained in this product, please contact opensource@altus.com.br. In addition to the source code, all referred license terms, warranty disclaimers and copyright notices may be disclosed under request.

Contents

1.	Introduction	1
1.1.	Documents Related to this Manual	2
1.2.	Visual Inspection	2
1.3.	Technical Support	3
1.4.	Warning Messages Used in this Manual	3
2.	Technical Description	4
2.1.	Panels and Connections	4
2.2.	Product Features	5
2.2.1.	General Features	5
2.2.2.	RS-485	7
2.2.3.	CAN	8
2.2.4.	USB	8
2.2.5.	Ethernet	9
2.2.6.	Power Supply	10
2.2.7.	Digital Inputs	10
2.2.8.	Fast Inputs	11
2.2.9.	Digital Outputs	12
2.2.10.	Fast Outputs	13
2.2.11.	Analog Inputs	14
2.2.12.	Analog Outputs	16
2.3.	Compatibility with Other Products	17
2.4.	Performance	17
2.4.1.	Interval Time	17
2.4.2.	Application Times	17
2.4.3.	Time for Instructions Execution	18
2.4.4.	Initialization Times	18
2.5.	Physical Dimensions	19
2.6.	Purchase Data	20
2.6.1.	Integrand Items	20
2.6.2.	Product Code	20
2.7.	Related Products	20
3.	Installation	22
3.1.	Mechanical Installation	22
3.1.1.	Installing the controller	22
3.1.2.	Removing the controller	24
3.2.	Electrical Installation	25
3.3.	Ethernet Network Connection	27
3.3.1.	IP Address	27
3.3.2.	Gratuitous ARP	27

3.3.3.	Network Cable Installation	27
3.4.	Serial RS-485 and CAN Network Connection	28
4.	Initial Programming	29
4.1.	Memory Organization and Access	29
4.2.	Project Profiles	31
4.2.1.	Machine Profile	31
4.3.	CPU Configuration	32
4.4.	Libraries	34
4.5.	Inserting a Protocol Instance	34
4.5.1.	MODBUS Ethernet	34
4.6.	Finding the Device	36
4.7.	Login	38
4.8.	Run Mode	40
4.9.	Stop Mode	41
4.10.	Writing and Forcing Variables	41
4.11.	Logout	42
4.12.	Project Upload	42
4.13.	CPU Operating States	43
4.13.1.	Run	43
4.13.2.	Stop	43
4.13.3.	Breakpoint	43
4.13.4.	Exception	44
4.13.5.	Reset Warm	44
4.13.6.	Reset Cold	44
4.13.7.	Reset Origin	44
4.13.8.	Reset Process Command (IEC 60870-5-104)	44
4.14.	Programs (POUs) and Global Variable Lists (GVLs)	44
4.14.1.	MainPrg Program	44
4.14.2.	StartPrg Program	45
4.14.3.	UserPrg Program	45
4.14.4.	GVL IntegratedIO	45
4.14.5.	GVL System_Diagnostics	46
4.14.6.	GVL Disables	47
4.14.7.	GVL Qualities	48
4.14.8.	GVL ReqDiagnostics	49
5.	Configuration	52
5.1.	Controller's CPU	52
5.1.1.	General Parameters	52
5.1.2.	Time Synchronization	53
5.1.2.1.	IEC 60870-5-104	53
5.1.2.2.	SNTP	54
5.1.2.3.	Daylight Saving Time (DST)	54
5.1.3.	Internal Points	54
5.1.3.1.	Quality Conversions	56
5.1.3.1.1.	Internal Quality	56
5.1.3.1.2.	IEC 60870-5-104 Conversion	58
5.1.3.1.3.	MODBUS Internal Quality	59
5.2.	Serial Interface	60
5.2.1.	COM 1	60

5.2.2.	Advanced Configurations	60
5.3.	Ethernet Interface	61
5.3.1.	NET 1	61
5.3.2.	Reserved TCP Ports	61
5.4.	Controller Area Network Interface	62
5.4.1.	CAN	62
5.5.	Integrated I/O	62
5.5.1.	Digital Inputs	63
5.5.2.	Fast Inputs	64
5.5.2.1.	High-Speed Counters	66
5.5.2.1.1.	Counter Interrupts	70
5.5.2.2.	External Interruption	71
5.5.3.	Fast Outputs	72
5.5.3.1.	VFO/PWM	73
5.5.3.2.	PTO	75
5.5.4.	Analog Inputs	80
5.5.5.	RTD Inputs	81
5.5.6.	Analog Outputs	82
5.5.7.	I/O Mapping	83
5.6.	USB Port	84
5.6.1.	Mass Storage Devices	85
5.6.1.1.	General Storage	85
5.6.1.2.	Not Loading the Application at Startup	86
5.6.1.3.	Transferring an Application from the USB device	87
5.6.2.	USB to RS-232 Converters	87
5.6.3.	Modem Devices	88
5.6.4.	WiFi Adapters	91
5.7.	Communication Protocols	94
5.7.1.	MODBUS RTU MASTER	94
5.7.2.	MODBUS RTU SLAVE	94
5.7.3.	MODBUS ETHERNET	94
5.7.4.	OPC DA Server	94
5.7.5.	OPC UA Server	95
5.7.6.	EtherNet/IP	95
5.7.7.	IEC 60870-5-104 Server	95
5.7.8.	CANopen Manager	95
5.7.8.1.	Installing and inserting CANopen Devices	95
5.7.8.2.	CANopen Manager Configuration	96
5.7.8.3.	CANopen Slave Configuration	97
5.8.	Remote I/O Mode	98
5.8.1.	CANopen Slave	98
5.9.	User Web Pages	102
5.10.	SNMP	102
5.10.1.	Introduction	102
5.10.2.	SNMP in Nexto Xpress Controllers	102
5.10.3.	Configuration	103
5.10.4.	User and SNMP Communities	104
5.11.	RTC Clock	105
5.12.	Function Blocks and Functions	105

- 6. Maintenance 106
 - 6.1. Diagnostics 106
 - 6.1.1. Diagnostics via LED 106
 - 6.1.2. Diagnostics via WEB 107
 - 6.1.3. Diagnostics via Variables 108
 - 6.1.3.1. Summarized Diagnostics 108
 - 6.1.3.2. Detailed Diagnostics 109
 - 6.1.4. Diagnostics via Function Blocks 116
 - 6.2. Preventive Maintenance 116

1. Introduction

Nexto Xpress is a powerful compact Programmable Logic Controller (PLC) part of Nexto Series family of controllers and I/O modules. Nexto Xpress delivers high-speed processing power in a compact design with embedded I/O. There are several options to choose from, allowing the best solution for entry-level applications.

This product portfolio targets small control systems, offering models containing from a few digital inputs and outputs up to options with 43 I/O points concentrated in a single controller, including analog inputs and outputs with temperature support (RTD sensors). In case of additional I/O needs, the system can be easily expanded through CANopen using the Remote I/O Mode. This mode transforms the product into a non-programmable slave I/O device, which can then be connected to a XP3xx controller with CANopen Manager protocol. It can also be expanded using other available ports like Ethernet and RS-485.

Nexto Xpress is suitable for small applications and remote distributed I/O. It may be applied in verticals such as infrastructure, building automation, water, wastewater, food, textiles, factory automation, machines and several other OEM solutions. Additionally, it is an ideal solution for complementing big applications along with Nexto Series portfolio, extending the range of applications using the same technology and engineering environment. This is a great advantage for OEMs and systems integrators with needs of small to large applications.

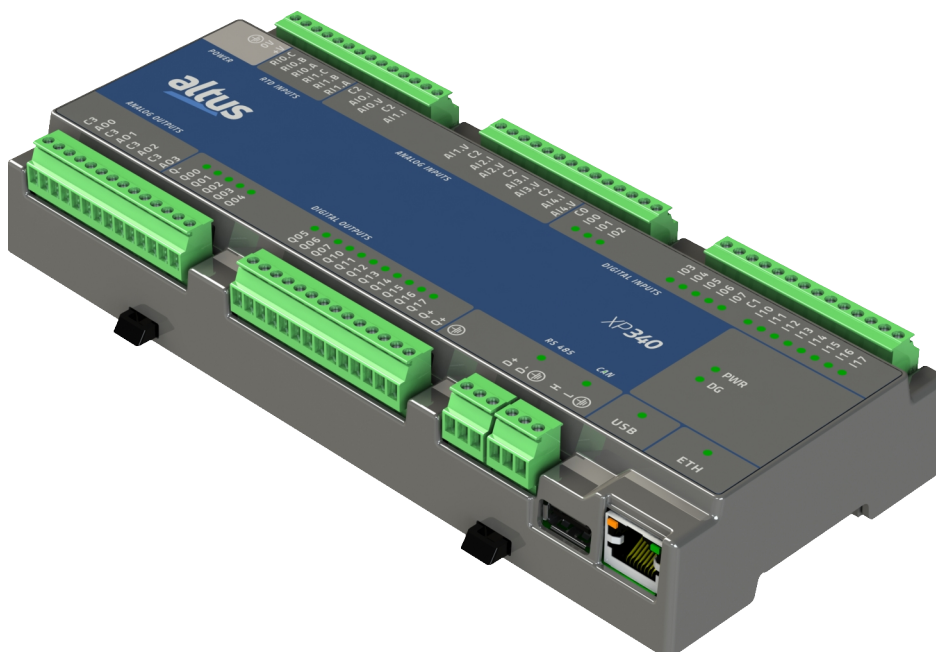


Figure 1: Nexto Xpress

1.1. Documents Related to this Manual

This manual will focus on information that is specific for the controllers of Nexto Xpress family. For other functionalities that are identical along all controllers of Nexto Series, this manual will just point to the corresponding manual of Nexto Series that contains the information. These related manuals are described on the following table, and are available in its last version on the site http://www.altus.com.br/site_en/.

Code	Description	Language
CE114000	Nexto Series – Technical Characteristics	English
CT114000	Série Nexto – Características Técnicas	Portuguese
CS114000	Serie Nexto – Especificaciones y Configuraciones	Spanish
MU216600	Nexto Xpress User Manual	English
MU216000	Manual de Utilização Nexto Xpress	Portuguese
MU214600	Nexto Series User Manual	English
MU214000	Manual de Utilização Série Nexto	Portuguese
MU214605	Nexto Series CPUs User Manual	English
MU214100	Manual de Utilização CPUs Série Nexto	Portuguese
MU299609	MasterTool IEC XE User Manual	English
MU299048	Manual de Utilização MasterTool IEC XE	Portuguese
MP399609	MasterTool IEC XE Programming Manual	English
MP399048	Manual de Programação MasterTool IEC XE	Portuguese
MU214606	MQTT User Manual	English
MU223603	IEC 60870-5-104 Server Device Profile	English
NAP151	Utilização do Tunneller OPC	Portuguese
NAP165	Comunicação OPC UA com Controladores ALTUS	Portuguese
	OPC UA Communication with ALTUS Controllers	English

Table 1: Documents Related

1.2. Visual Inspection

Before resuming the installation process, it is advised to carefully visually inspect the equipment, verifying the existence of transport damage. Verify if all parts requested are in perfect shape. In case of damages, inform the transport company or Altus distributor closest to you.

CAUTION:

Before taking the modules off the case, it is important to discharge any possible static energy accumulated in the body. For that, touch (with bare hands) on any metallic grounded surface before handling the modules. Such procedure guaranties that the module static energy limits are not exceeded.

It's important to register each received equipment serial number, as well as software revisions, in case they exist. This information is necessary, in case the Altus Technical Support is contacted.

1.3. Technical Support

For Altus Technical Support contact in São Leopoldo, RS, call +55 51 3589-9500. For further information regarding the Altus Technical Support existent on other places, see <https://www.altus.com.br/en/> or send an email to altus@altus.com.br.

If the equipment is already installed, you must have the following information at the moment of support requesting:

- The model of the used equipments and the installed system configuration
- The product serial number
- The equipment revision and the executive software version, written on the tag fixed on the product side
- CPU operation mode information, acquired through MasterTool IEC XE
- The application software content, acquired through MasterTool IEC XE
- Used programmer version

1.4. Warning Messages Used in this Manual

In this manual, the warning messages will be presented in the following formats and meanings:

DANGER:
Reports potential hazard that, if not detected, may be harmful to people, materials, environment and production.

CAUTION:
Reports configuration, application or installation details that must be taken into consideration to avoid any instance that may cause system failure and consequent impact.

ATTENTION:
Identifies configuration, application and installation details aimed at achieving maximum operational performance of the system.

2. Technical Description

This chapter presents all technical features of Nexto Xpress controllers.

2.1. Panels and Connections

The following figure shows the XP325 front panel:

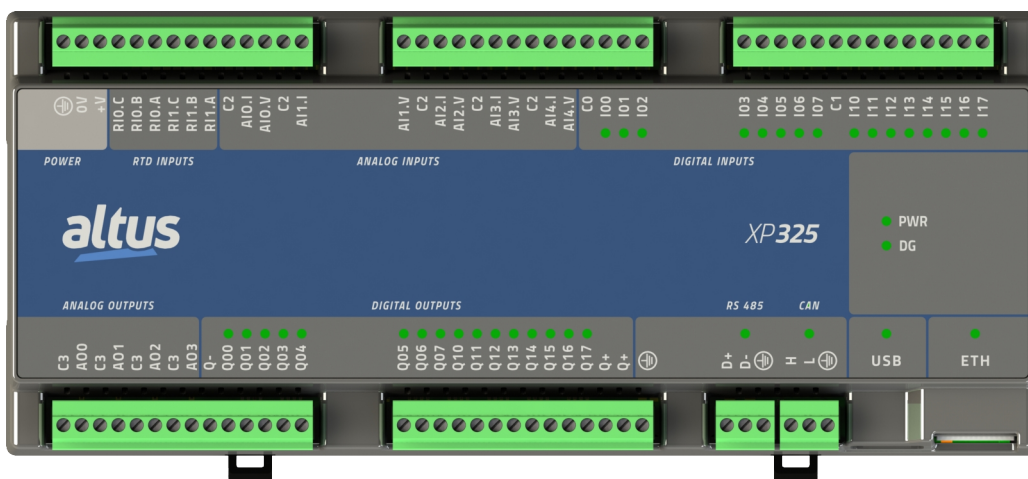


Figure 2: XP325 front panel

The front panel contains the identification of the I/O and communication interfaces available on Nexto Xpress controllers. The digital I/O interfaces have one LED for each point to indicate the logic state, while the communication interfaces have one LED each to indicate activity. The availability of these interfaces on each model is described on next section.

Additionally, on the right side of front panel there are 2 LEDs used to indicate power and diagnostics. The following table shows the LEDs description. For further information regarding the LEDs status and meaning, see [Maintenance](#) chapter.

LED	Description
PWR	Status of internal power supply
DG	Diagnostic indication
Ixx.x	Status of digital inputs
Qxx.x	Status of digital outputs
D+/-	Status of RS-485 interface (blinks on activity)
H/L	Status of CAN interface (blinks on activity)
USB	Status of USB port (turns on when device is mounted)
ETH	Status of Ethernet interface (turns on when connected, blinks on activity)

Table 2: LEDs Description

2.2. Product Features

2.2.1. General Features

	XP300	XP315	XP325	XP340
Digital Inputs	12			
Fast Inputs	4			
Digital Outputs	12			
Fast Outputs	4			
Max. number of high-speed counters	1			
Max. number of external interruptions	2			
Max. number of PTO outputs	2			
Max number of VFO/PWM outputs	4			
V/I analog inputs (AI)	-	5	5	5
RTD analog inputs (AI)	-	2	2	2
V/I analog outputs (AO)	-	-	4	4
Ethernet TCP / IP interface	1			
RS-485 Serial interface	1			
CAN Interface	1			
USB Host port	1			
CANOpen Manager protocol	Yes			
IEC 60870-5-104 Server protocol	No	No	No	Yes
User web pages (Webvisu)	No	No	No	Yes
Remote I/O Mode	Yes			
Addressable input variables memory (%I)	2 KB			
Addressable output variables memory (%Q)	2 KB			
Addressable variables memory (%M)	1 KB			
Symbolic variables memory	2 MB			
Program memory	2 MB	2 MB	2 MB	6 MB
Retain/persistent memory (user configurable)	7.5 KB			
Source code memory (backup)	26 MB			
User files memory (backup)	8 MB			
Maximum number of tasks	16			
Programming languages	Structured Text (ST) Ladder Diagram (LD) Sequential Function Chart (SFC) Function Block Diagram (FBD) Continuous Function Chart (CFC)			
Online changes	Yes			
Watchdog	Yes			
Real-time clock (RTC)	Yes Resolution of 1 ms, max. variance of 3 seconds per day, retention time of 14 days.			
Status and diagnostic indication	LEDs, web pages and CPU's internal memory			
Isolation				
Protective earth \oplus to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Logic/RS-485/CAN/USB to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Ethernet to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Power Supply/Analog I/O to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Digital Inputs to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Digital Inputs Group I0x to I1x	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			




	XP300	XP315	XP325	XP340
Digital Outputs to all	1,500 Vdc / 1 minute (1,000 Vac / 1 minute)			
Maximum power dissipation	5 W			
Maximum wire size	0.5 mm ² (20 AWG) with ferrule 1.5 mm ² (16 AWG) without ferrule			
Minimum wire temperature rating	75 °C			
Wire material	Copper only			
IP level	IP 20			
Conformal coating	Yes			
Operating temperature	-20 to 60 °C			
Storage temperature	-25 to 75 °C			
Operating and storage relative humidity	5% to 96%, non-condensing			
Vibration resistance (IEC 60068-2-6, sinus)	7 mm from 5 to 8.4 Hz 2 G from 8.4 to 500 Hz 10 sweeps each axis, 1 octave per minute			
Shock resistance (IEC 60068-2-27, half-sine)	15 G for 11 ms, 6 shocks in each of 3 axis			
Standards	IEC 61131-2 CE – 2011/65/EU (RoHS), 2014/35/EU (LVD) and 2014/30/EU (EMC) UL/cUL Listed - UL 61010-1 (file E473496)   			
Product dimensions (W x H x D)	215.5 x 98.8 x 34.0 mm			
Package dimensions (W x H x D)	270.0 x 102.0 x 40.0 mm			
Weight	370 g			
Weight with package	430 g			

Table 3: General Features

Notes:

Persistent and Retain symbolic variables memory: Area where are allocated the retentive symbolic variables. The retentive data keep its respective values even after a CPU's cycle of power down and power up. The persistent data keep its respective values even after the download of a new application in the CPU.

ATTENTION:

The declaration and use of symbolic persistent variables should be performed exclusively through the *Persistent Vars* object, which may be included in the project through the tree view in *Application -> Add Object -> Persistent Variables*. It should not be used the *VAR PERSISTENT* expression in the declaration of field variables of POU's.

The full list of when the symbolic persistent variables keep their values and when the values are lost can be found in the following table. Additionally to the persistent area size mentioned on general features table, 44 bytes are reserved for the storage of information about the persistent variables (not available for use).

The following table shows the behavior of retentive and persistent variables for different situations where “-“ means the value is lost and “X” means the value is kept.

Command	Symbolic Variable	Retain variable	Persistent variable
Reset warm / Power-on/off cycle	-	X	X
Reset cold	-	-	X
Reset Origin	-	-	-
Download	-	-	X
Online change	X	X	X
Reboot PLC	-	X	X
Clean All	-	-	X
Reset Process (IEC 60870-5-104)	-	X	X

Table 4: Post-command Variable Behavior

Notes:

Maximum number of tasks: This value represents the maximum total of user and system tasks. The detailed description of possible user tasks can be found on Project Profiles section of User Manual. Before MasterTool IEC XE v3.30, this value was defined as “5”.

Isolation: The *Logic* term refers to the internal interfaces such as processors, memories and USB, serial and CAN communication interfaces.

Conformal coating: Conformal coating protects the electronic components inside the product from moisture, dust and other harsh elements to electronic circuits.

Operating temperature: The minimum operating temperature is 0°C for units with product revision inferior to AS/AS/AW/AE for XP300/XP315/XP325/XP340 respectively.

2.2.2. RS-485

	RS-485
Connector	3-pin terminal block
Physical interface	RS-485
Communication direction	RS-485: half duplex
RS-485 max. transceivers	32
Termination	Yes (Configurable)
Baud rate	2400, 4800, 9600, 19200, 38400, 57600, 115200 bps
Protocols	Master/Slave MODBUS RTU Open protocol

Table 5: RS-485 Serial Interface Features

2.2.3. CAN

	CAN
Connector	3-pin terminal block
Physical interface	CAN bus
Supported standards	CAN 2.0A 2.0B (11-bit and 29-bit identifiers)
Max. number of nodes	64
Termination	Yes (Configurable)
Baud rate	10, 20, 50, 100, 125, 250, 500, 800, 1000 kbit/s
Protocols	CANOpen Manager (Master) CANOpen Slave CAN low level

Table 6: CAN Interface Features

2.2.4. USB

	USB
Connector	USB A Female
Physical interface	USB V2.0
Baud rate	1.5 Mbps (Low Speed), 12 Mbps (Full Speed) and 480 Mbps (High Speed)
Maximum current	500 mA
Supported devices	Mass storage USB RS-232 Serial Converter USB 3G/4G Modem USB WiFi Adapter

Table 7: USB Interface Features

Notes:

USB RS-232 Serial Converter: See the list of supported devices on respective section [USB to RS-232 Converters](#).

USB 3G/4G Modem: See the list of supported devices on respective section [Modem Devices](#).

USB WiFi Adapter: See the list of supported devices on respective section [WiFi Adapters](#).

2.2.5. Ethernet

	Ethernet
Connector	Shielded female RJ45
Auto crossover	Yes
Maximum cable length	100 m
Cable type	UTP or ScTP, category 5
Baud rate	10/100 Mbps
Physical layer	10/100 BASE-TX
Data link layer	LLC
Network layer	IP
Transport layer	TCP (Transmission Control Protocol) UDP (User Datagram Protocol)
Application layer	MODBUS TCP Client and Server MODBUS RTU Master/Slave OPC DA Server OPC UA Server EtherNet/IP Scanner IEC 60870-5-104 Server MQTT Client HTTP Server MasterTool IEC XE programming protocol SNTP Client SNMP Agent (Ethernet Network Management)
Diagnostics	LED (Link/Activity)

Table 8: Ethernet Interface Features

2.2.6. Power Supply

	Power Supply
Nominal input voltage	24 Vdc
Input voltage	19.2 to 30 Vdc
Maximum input current (in-rush)	50 A / 300 us
Maximum input current	300 mA

Table 9: Power Supply Features

2.2.7. Digital Inputs

	Digital Inputs
Input type	Optoisolated sink type 1 Two isolated groups of 8 inputs each
Input voltage	24 Vdc 15 to 30 Vdc for logic level 1 0 to 5 Vdc for logic level 0
Input impedance	4.95 k Ω
Maximum input current	6.2 mA @ 30 Vdc
Input state indication	Yes
Response time	0.1 ms
Input filter	Disabled or 2 ms to 255 ms – by software

Table 10: Digital Inputs Features

Note:

Input filter: The filter sampling is performed on MainTask (or Refresh function), then it's recommended to use multiple values of the task interval.

2.2.8. Fast Inputs

	Fast Inputs
Number of fast inputs	4 (can be used as high-speed counter, External interrupt or normal input)
Max. number of high-speed counters	1
Max. number of external interrupts	2
Connector configuration	I00, I01, I02 and I03
Input voltage	24 Vdc 15 to 30 Vdc for logic level 1 0 to 5 Vdc for logic level 0
Input impedance	1.85 k Ω
Input maximum current	16.2 mA @ 30 Vdc
Configuration mode	1-input modes Normal digital input External interrupt 2-input modes Up/Down (A count, B direction) with zero (uses I00, I01, I02) Quadrature 2x (uses I00, I01) Quadrature 2x with zero (uses I00, I01, I02) Quadrature 4x (uses I00, I01) Quadrature 4x with zero (uses I00, I01, I02)
Counting direction control	By software or hardware
Counting input detection edge	Rising edge, active at logic level 1 (except for quadrature 4x, where it counts on both edges)
Data format	Signed 32-bit integer
Operation limit	From - 2,147,483,648 to 2,147,483,647
Maximum input frequency	100 kHz
Minimum pulse width @ 24 Vdc	2 μ s

Table 11: Fast Inputs Features

2.2.9. Digital Outputs

	Digital Outputs
Output type	Optoisolated transistor source type
Maximum output current	1.5 A per output 12 A total
Leakage current	35 μ A
On state resistance	105 m Ω
External power supply	19.2 to 30 Vdc
Switching time	20 μ s - off-to-on transition @ 24 Vdc 500 μ s - on-to-off transition @ 24 Vdc
Maximum switching frequency	250 Hz
Configurable parameters	Yes
Output state indication	Yes
Output protections	Yes, protection against surge voltages

Table 12: Digital Outputs Features

Note:

Switching time: The required time to turn off one specific output depends on the load.

2.2.10. Fast Outputs

Fast Outputs		
Number of outputs	4 (can be used as VFO/PWM, PTO or normal output)	
Max. number of PTO outputs	2	
Max number of VFO/PWM outputs	4 when using no PTO 2 when using 1 PTO 0 when using 2 PTO	
Connector configuration	Q14, Q15, Q16 and Q17	
Maximum current	0.5 A @ 30 Vdc by output 2 A @ 30 Vdc total	
Output type	Transistor source	
Pulse generation maximum frequency	200 kHz @ 60 mA	
Minimum pulse width @ 24 Vdc	MINIMUM LOAD	MINIMUM PULSE TIME
	400 Ω	320 ns
State indication	Through static reserved operands	
Protections	TVS diode at all transistor outputs	
Operation voltage	19.2 to 30 Vdc	
Output impedance	700 m Ω	
Output modes	Normal digital output VFO/PWM PTO (Q14 and Q16 only. Adjacent output is forced to normal digital output)	
Functions executed by software	PTO	VFO/PWM
	Writing of number of pulses to be generated Writing of acceleration and deceleration number of pulses Start/end outputs operation Fast outputs diagnostics Fast outputs current state monitoring	Writing of the frequency value to be generated (1 Hz to 200 kHz). Writing of outputs duty cycle (1% to 100%) Start/end of outputs operations Fast outputs diagnostics.

Table 13: Fast Outputs Features

2.2.11. Analog Inputs

Analog Inputs	
Input type	Voltage or current input, single ended, individually configured
Data format	16 bits in two's complement, justified to the left
Converter resolution	12 bits monotonicity guaranteed, no missing codes
Conversion time	400 μ s (all V/I and RTD channels enabled)
Input state indication	Yes
Module protections	Yes, protection against surge voltages and polarity inversion

Table 14: Analog Inputs Features

Input ranges	Voltage Input Mode		
	Range	Engineering Scale	Resolution
	0 to 10 Vdc	0 to 30,000	2.5 mV
Precision	± 0.3 % of full scale @ 25 °C ± 0.010 % of full scale / °C		
Over scale	3 % of full scale		
Maximum input voltage	12 Vdc		
Input impedance	21 k Ω		
Configurable parameters	Signal type per input Filters		
Low pass filter time constant	100 ms, 1 s, 10 s or disabled		

Table 15: Voltage Input Mode Features

Input ranges	Current Input Mode		
	Range	Engineering Scale	Resolution
	0 to 20 mA 4 to 20 mA	0 to 30,000 0 to 30,000	5.12 μ A 5.12 μ A
Precision	± 0.3 % of full scale @ 25 °C ± 0.015 % of full scale / °C		
Over scale	3 % of full scale		
Maximum input current	30 mA		
Input impedance	119 Ω		
Configurable parameters	Signal type per input Filters Open Loop Value		
Low pass filter time constant	100 ms, 1 s, 10 s or disabled		

Table 16: Current Input Mode Features

Note:

Input ranges: When configured as 4 to 20 mA, input signals lower than 4 mA will result in negative values (-7,500 for 0 mA). Starting from MasterTool IEC XE version 3.16, a new parameter called *Open Loop Value* was included to select the behavior in this situation. The default value is *Disabled* (which provides a linear reading as described above), having also the option to provide a fixed reading equal to lower and upper limits ("0" or "30000").

	RTD Input
Precision	± 0.5 % of full scale @ 25 °C
Supported scales	Pt100, Pt1000, 0 to 400 Ω , 0 to 4000 Ω
Excitation current	1 mA
Resistance range (scale)	0 to 400 Ω (used for PT100) 0 to 4000 Ω (used for PT1000)
Over Scale	5 % of full scale
Configurable parameters	Signal type per input Filters
Low pass filter time constant	100 ms, 1 s, 10 s or disabled
Maximum sensor cable impedance	20 Ω

Table 17: RTD Input Features

Note:

Maximum sensor cable impedance: Maximum total resistance added by the two wires of the sensor.

2.2.12. Analog Outputs

Analog Outputs	
Output type	Voltage or current output, individually configured
Data format	16 bits in two's complement, justified to the left
Converter resolution	12 bits monotonicity guaranteed, no missing codes
Update time	450 μ s (all outputs enabled)
Output state indication	Yes
Module protections	Yes, protection against surge voltages and polarity inversion

Table 18: Analog Outputs Features

Voltage Output Mode			
Output ranges	Range	Engineering Scale	Resolution
	0 to 10 V	0 to 30,000	2.5 mV
Precision	± 0.3 % of full scale @ 25 °C ± 0.025 % of full scale / °C		
Stabilization time	4 ms		
Maximum output value	+ 10.3 Vdc		
Load impedance	> 1 k Ω		
Configurable parameters	Signal type per output		

Table 19: Voltage Output Mode Features

Current Output Mode			
Output ranges	Range	Engineering Scale	Resolution
	0 to 20 mA	0 to 30,000	5.18 μ A
	4 to 20 mA	0 to 30,000	5.18 μ A
Precision	± 0.3 % of full scale @ 25 °C ± 0.020 % of full scale / °C		
Stabilization time	4 ms		
Maximum output value	+ 20.6 mA		
Load impedance	< 600 Ω		
Configurable parameters	Signal type per output		

Table 20: Current Output Mode Features

Note:

Output ranges: When configured as 4 to 20 mA, the output can be set to values lower than 4 mA by assigning negative values to the output variable (-7,500 for 0 mA).

2.3. Compatibility with Other Products

To develop an application for Nexto Xpress controllers, it is necessary to check the version of MasterTool IEC XE. The following table shows the minimum version required (where the controllers were introduced) and the respective firmware version at that time:

Controller model	MasterTool IEC XE	Firmware version
XP300, XP315 and XP325	3.10 or above	1.7.0.0 or above
XP340	3.18 or above	1.8.0.0 or above

Table 21: Compatibility with other products

Additionally, along the development roadmap of MasterTool IEC XE some features may be included (like special FunctionBlocks, etc...), which can introduce a requirement of minimum firmware version. During the download of the application, MasterTool IEC XE checks the firmware version installed on the controller and, if it does not meets the minimum requirement, will show a message requesting to update. The latest firmware version can be downloaded from Altus website, and it is fully compatible with previous applications.

2.4. Performance

The performance of Nexto Xpress controller relies on:

- Application Interval Time
- User Application Time
- Operational System Time
- Number of integrated I/O channels enabled

2.4.1. Interval Time

The application and I/O update are executed on a cyclic (periodic) task called MainTask. The interval time of this task can be configured from 5 to 100 ms. The time spent for these operations is called Cycle Time, and should always be smaller than the interval, because the free time is used for communication and other low priority tasks of the controller.

Additionally, the integrated I/O can be updated asynchronously at any point of user application code using the refresh functions available on *LibIntegratedIo*, which is described in details on Nexto Series CPUs User Manual code MU214605.

2.4.2. Application Times

The execution time of the application (cycle time) depends on the following variables:

- Integrated inputs read time
- Task execution time
- Integrated outputs write time

The time required for reading and writing the integrated I/O is dependent of the number and the type of the I/O channels enabled. For digital I/O, all channels are always enabled and the time added to MainTask is not relevant. For analog I/O, the time added to MainTask is determined by the conversion time (for analog inputs) and by the update time (for analog outputs), both described on General Features table.

2.4.3. Time for Instructions Execution

The below table presents the necessary execution time for different instructions in Nexto Xpress CPUs.

Instruction	Language	Variables	Instruction Times (μ s)
1000 Contacts	LD	BOOL	10
1000 Divisions	ST	INT	180
		REAL	40
	LD	INT	180
		REAL	40
1000 Multiplications	ST	INT	13
		REAL	13
	LD	INT	13
		REAL	13
1000 Sums	ST	INT	13
		REAL	13
	LD	INT	13
		REAL	13

Table 22: Instruction Times

2.4.4. Initialization Times

The initialization time of Nexto Xpress controllers is approximately 40 s.

2.5. Physical Dimensions

Dimensions in mm.

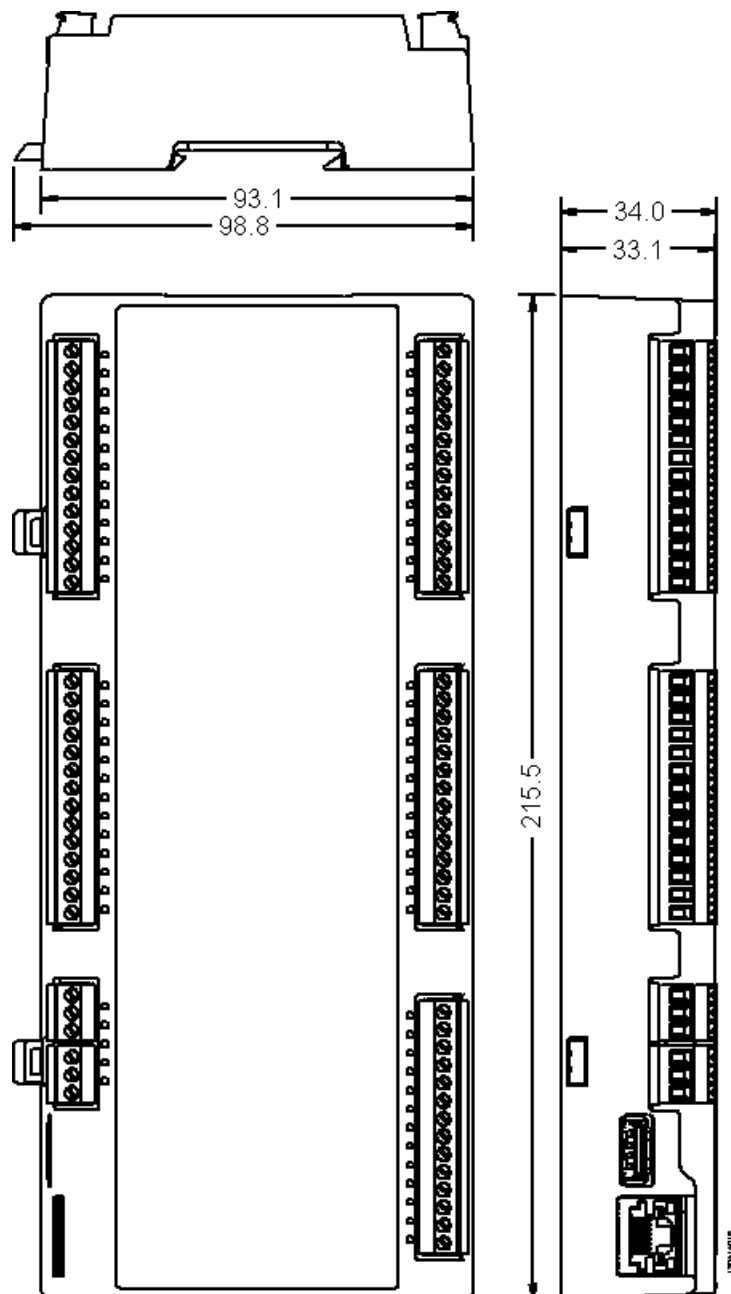


Figure 3: XP3xx Physical Dimensions

2.6. Purchase Data

2.6.1. Integrant Items

The product package has the following items:

- Compact PLC module
- Connectors
- Installation guide

2.6.2. Product Code

The following code should be used to purchase the product:

Code	Description
XP300	High-Speed Compact PLC with 16 DI, 16 DO Transistor, 1 Ethernet, 1 RS-485 Serial and CANopen Master
XP315	High-Speed Compact PLC with 16 DI, 16 DO Transistor, 5 V/I AI, 2 RTD AI (3 wire), 1 Ethernet, 1 RS-485 Serial and CANopen Master
XP325	High-Speed Compact PLC with 16 DI, 16 DO Transistor, 5 V/I AI, 2 RTD AI (3 wire), 4 AO, 1 Ethernet, 1 RS-485 Serial and CANopen Master
XP340	High-Speed Compact PLC with 16 DI, 16 DO Transistor, 5 V/I AI, 2 RTD AI (3 wire), 4 AO, 1 Ethernet, 1 RS-485 Serial, CANopen Master and user web pages support

Table 23: Nexto Xpress Controller Models

2.7. Related Products

The following products must be purchased separately when necessary:

Code	Description
MT8500	MasterTool IEC XE
NX9202	RJ45-RJ45 2 m Cable
NX9205	RJ45-RJ45 5 m Cable
NX9210	RJ45-RJ45 10 m Cable
AL-2600	RS-485 network branch and terminator
AL-2306	RS-485 cable for MODBUS or CAN network
AL-1766	CFDB9-Terminal Block Cable
FBS-USB-232M9	Universal USB-Serial converter cable / 2m
XP900	TP-Link nano Wireless 150 Mbps USB Adapter TL-WN725N (only available in Brazil)
XP910	Huawei E3276 3G/4G USB Modem with External Antenna Input (only available in Brazil)
XP920	7dbi GSM antenna with 2 meter cable (only available in Brazil)
XP930	3G/4G modem adapter cable for external GSM antenna (only available in Brazil)
AMJG0808	Simple cable RJ45-RJ45 2 m

Table 24: Related Products

Notes:

MT8500: MasterTool IEC XE is available in four different versions: LITE, BASIC, PROFESSIONAL and ADVANCED. For more details, please check MasterTool IEC XE User Manual - MU299609.

NX92xx: Cable for programming the CPUs of the Nexto Series and Ethernet point-to-point with another device with Ethernet interface communication.

AL-2600: This module is used for branch and termination of RS-485 networks. For each network node, an AL-2600 is required. The AL-2600 that are at the ends of network must be configured with termination, except when there is a device with active internal termination, the rest must be configured without termination.

AL-2306: Two shielded twisted pairs cable without connectors, used for networks based on RS-485 or CAN.

AL-1766: Cable with a female DB9 connector and terminals for communication between HMI P2 and Nexto Xpress/NX3003 controllers.

FBS-USB-232M9: Cable for use as a USB-Serial converter on the USB interface of Xpress controllers.

AMJG0808: Cable for programming the CPUs.

3. Installation

This chapter presents the necessary proceedings for the physical installation of Nexto Xpress controllers, as well as the care that should be taken with other installation within the panel where the controller is been installed.

CAUTION:
If the equipment is used in a manner not specified by in this manual, the protection provided by the equipment may be impaired.

3.1. Mechanical Installation

Nexto Xpress controllers were designed to be installed in a standard DIN rail. Additionally, the user shall provide a suitable enclosure that meets the system protection and safety requirements. The next sections shows the procedures for installing and removing the controller.

CAUTION:
For achieving the temperature specification of the controller, the installation must provide a free space around the device as described on section Panel Design of Nexto Series User Manual code MU214600.

3.1.1. Installing the controller

To install the controller on the DIN rail, first move the two locks on open position as indicated on the figure below:

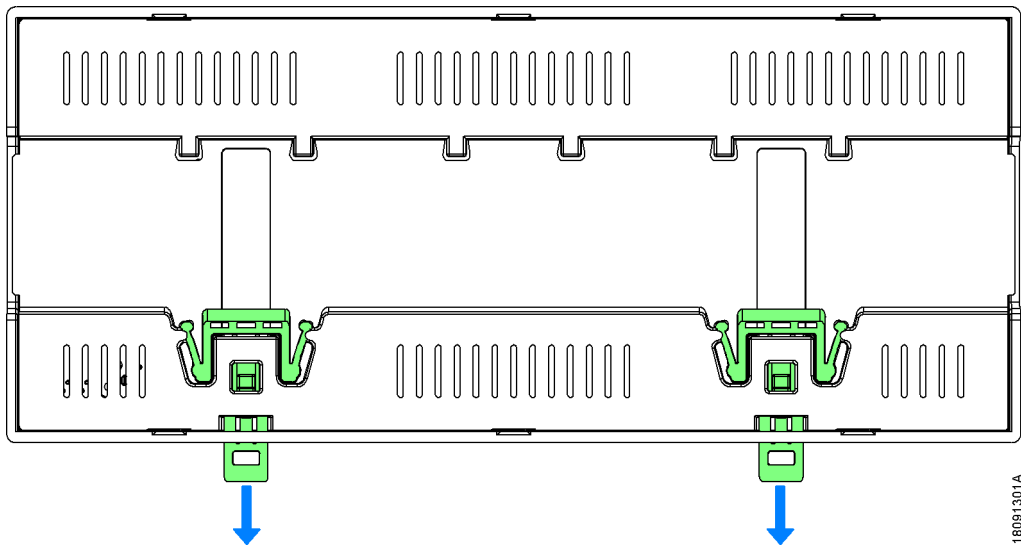


Figure 4: Moving the two locks to open position

Next, place the controller on the DIN rail fitting the top side first and then the bottom side, as indicated on steps 1 and 2 of the figure below:

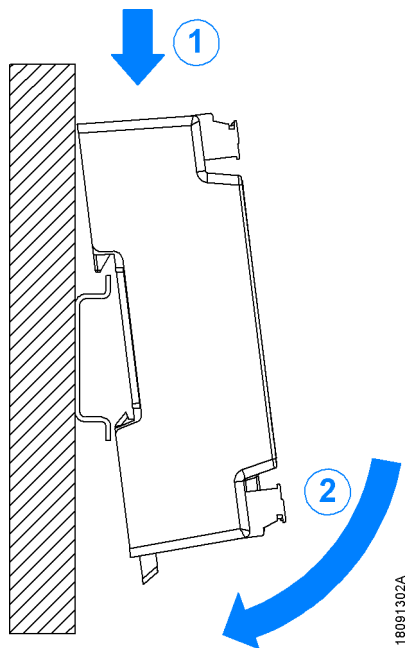


Figure 5: Fixing the controller on the DIN rail

Finally, move the two locks to closed position to lock the controller on the DIN rail, as shown on the figure below:

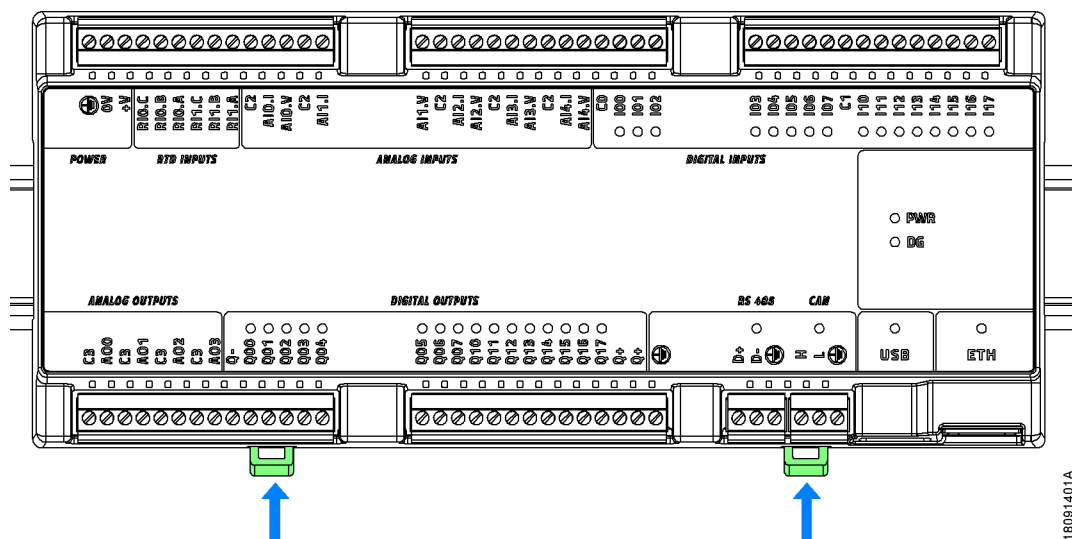


Figure 6: Locking the controller on the DIN rail

3.1.2. Removing the controller

To remove the controller from the DIN rail, just move the two locks to the open position as shown on the figure below:

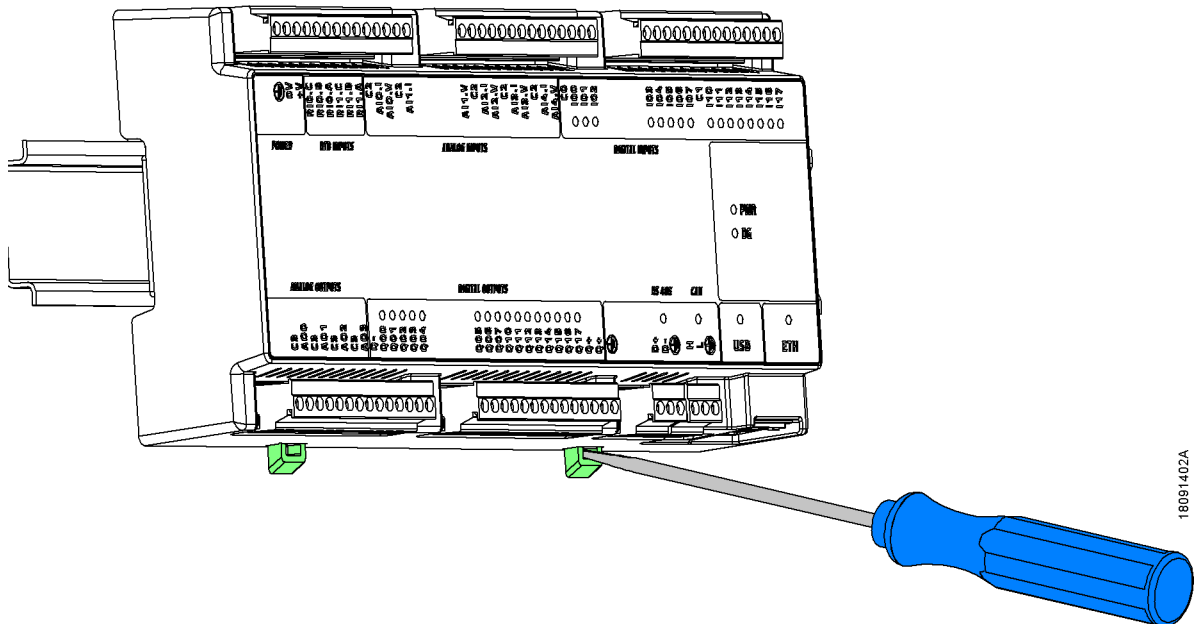


Figure 7: Unlocking the controller from the DIN rail

3.2. Electrical Installation

DANGER:

When executing any installation in an electric panel, certify that the main energy supply is OFF.

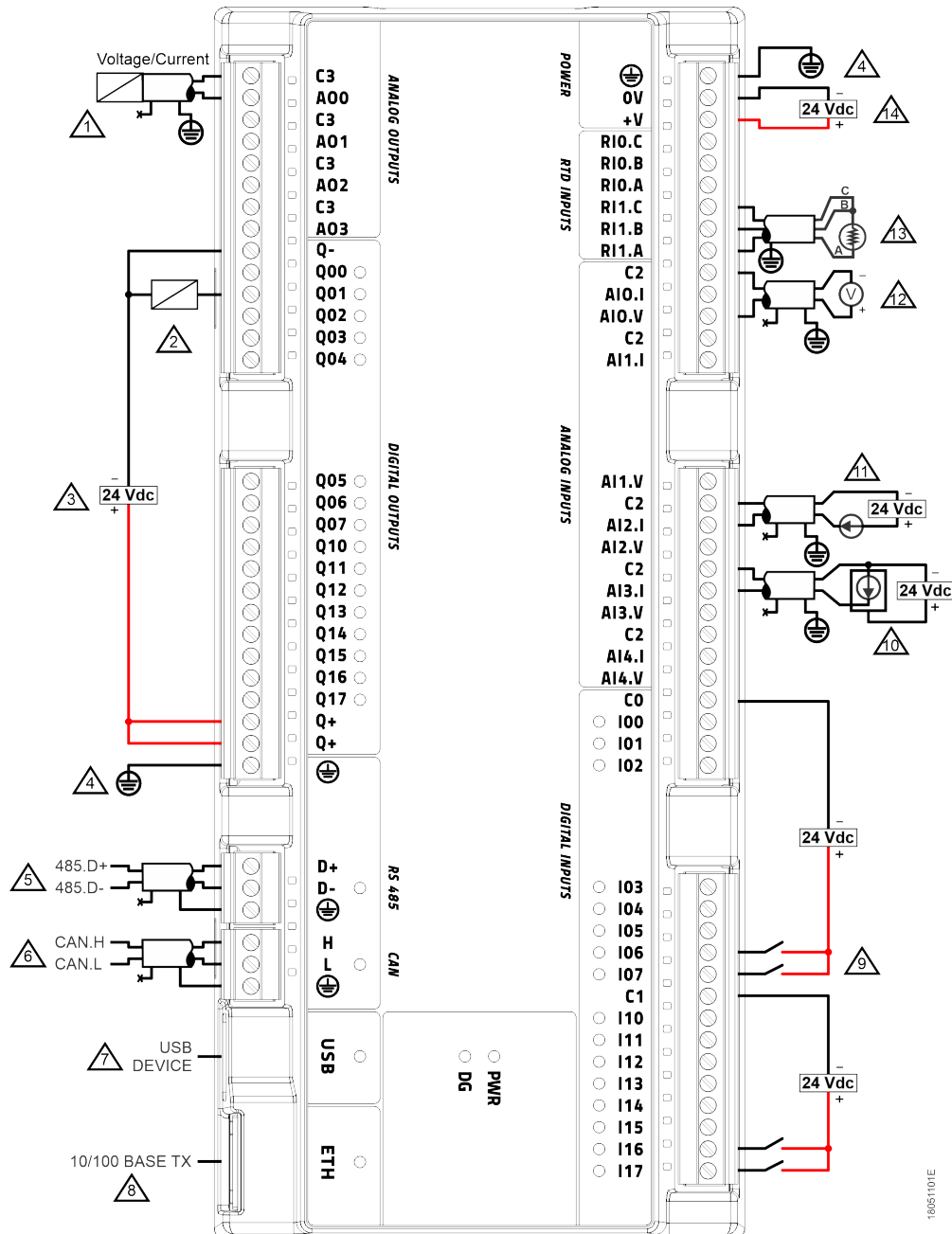


Figure 8: XP3xx Electrical Installation Diagram

Diagram Notes:

1. Typical connection of analog output on voltage/current mode
2. Typical connection of digital output (source type)
3. External power supply to supply outputs Q00 to Q17, terminals Q + must be connected to +24 Vdc, and terminal Q- must be connected to 0 Vdc
4. Protective Earth terminals for power supply and communication ports. Both shall be externally connected to ground
5. Typical connection of RS-485 serial interface
6. Typical connection of CAN interface
7. Please check the technical characteristics table of USB port for the list of supported devices
8. Use Ethernet cables informed on Related Products section
9. Typical connection of digital input (sink type). C0 and C1 are the common points for the isolated groups I0x and I1x respectively
10. Typical connection of current analog input (field device with power supplied separately from analog signal)
11. Typical connection of current analog input (field device with power supplied with the analog signal, 2-wire)
12. Typical connection of voltage analog input
13. Typical connection of RTD analog input (3-wire)
14. External power supply connection

3.3. Ethernet Network Connection

The ETH communication interface, identified as NET 1 on MasterTool IEC XE, allows the connection with an Ethernet network and programming with this tool.

The Ethernet network connection uses twisted pair cables (10/100Base-TX) and the speed detection is automatically made by the Nexto Xpress controller. This cable must have one of its endings connected to the interface that is likely to be used and another one to the HUB, switch, microcomputer or other Ethernet network point.

3.3.1. IP Address

The Ethernet interface comes with the following default parameters configuration:

	NET 1
IP Address	192.168.15.1
Subnet Mask	255.255.255.0
Gateway Address	192.168.15.253

Table 25: Default Parameters Configuration for Ethernet NET 1 Interface

First, the NET 1 interface must be connected to a PC network with the same subnet mask to communicate with MasterTool IEC XE, where the network parameters can be modified. For further information regarding configuration and parameters modifications, see [Ethernet Interface](#) chapter.

3.3.2. Gratuitous ARP

The NET1 Ethernet interface promptly sends ARP packets type in broadcast informing its IP and MAC address for all devices connected to the network. These packets are sent during a new application download by the MasterTool IEC XE software and in the controller startup when the application goes into Run mode.

Five ARP commands are triggered with a 200 ms initial interval, doubling the interval every new triggered command, totalizing 3 s. Example: first trigger occurs at time 0, the second one at 200 ms and the third one at 600 ms and so on until the fifth trigger at time 3 s.

3.3.3. Network Cable Installation

Nexto Xpress Ethernet port have standard pinout which are the same used in PCs. The connector type, cable type, physical level, among other details, are defined in the General Features table. Below is the description of the RJ-45 female connector, with the identification and description of the valid pinout for 10Base-T and 100Base-TX physical levels.

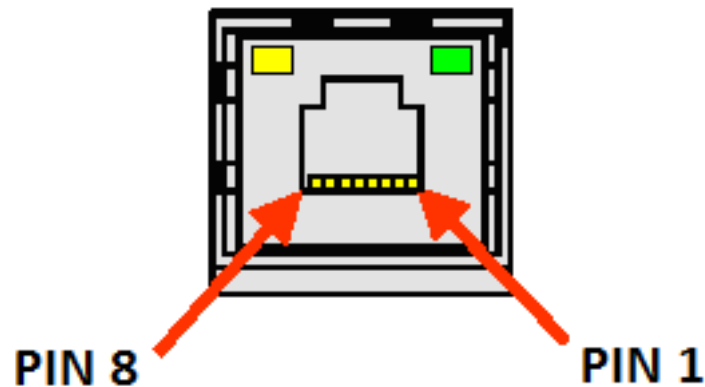


Figure 9: RJ45 Female Connector

Pin	Signal	Description
1	TXD +	Data transmission, positive
2	TXD -	Data transmission, negative
3	RXD +	Data reception, positive
4	NU	Not used
5	NU	Not used
6	RXD -	Data reception, negative
7	NU	Not used
8	NU	Not used

Table 26: RJ45 Female Connector Pin out

The interface can be connected in a communication network through a hub or switch, or straight from the communication equipment. In this last case, due to Auto Crossover feature, there is no need for a cross-over network cable, the one used to connect two PCs point to point via Ethernet port.

It is important to stress that it is understood by network cable a pair of RJ45 male connectors connected by a UTP or ScTP cable, category 5 whether straight connecting or cross-over. It is used to communicate two devices through the Ethernet port.

These cables normally have a connection lock which guarantees a perfect connection between the interface female connector and the cable male connector. At the installation moment, the male connector must be inserted in the module female connector until a click is heard, assuring the lock action. To disconnect the cable from the module, the lock lever must be used to unlock one from the other.

3.4. Serial RS-485 and CAN Network Connection

As illustrated on [Electrical Installation](#) diagram, both RS-485 and CAN interface uses two communication signals and a ground. The recommended cable is AL-2306, using one of the two pairs and the shield. If the controller is placed at one of the network ends, the internal termination shall be enabled (see [Controller's CPU](#) and [Serial Interface](#) configuration sections for CAN and RS-485 respectively).

4. Initial Programming

The main goal of this chapter is to help the programming and configuration of Nexto Xpress controllers, allowing the user to take the first steps before starting to program the device.

Just like for the other devices of Nexto Series, the programming of Nexto Xpress controllers is made through the MasterTool IEC XE (IDE) development interface, which offers a full IEC 61131-3 programming system with all languages defined by this standard (ST, LD, SFC, FBD, etc...) plus an additional one, the CFC. These languages can be used simultaneously on the same project, allowing the user to use the best features of each language, resulting in more efficient applications development, for easy documentation and future maintenance.

For further information regarding programming, see User Manual MasterTool IEC XE - MU299609, Programming Manual MasterTool IEC XE - MU399609 or IEC 61131-3 standard.

4.1. Memory Organization and Access

Different from other devices of Nexto Series (which are based on big-endian CPU), the Nexto Xpress controllers are based on a ARM CPU, which uses the traditional little-endian memory organization (the same found on x86 and Intel processors). On this type of memory organization, the least significant byte is stored first and will always be the smallest address (e.g. %QB0 will always be less significant than %QB1, as shown on the table below, where, for CPUNEXTO string, the letter O is byte 0 and the letter C is the byte 7).

Besides this, the memory access must be done carefully as the variables with higher number of bits (WORD, DWORD, LONG), use as index the most significant byte, in other words, the %QD4 will always have as most significant byte the %QB4. Therefore it will not be necessary to make calculus to discover which DWORD correspond to defined bytes. The Table 27, shows little and big endian organization.

MSB ← Little-endian → LSB								
BYTE	%QB7	%QB6	%QB5	%QB4	%QB3	%QB2	%QB1	%QB0
	C	P	U	N	E	X	T	O
WORD	%QW3		%QW2		%QW1		%QW0	
	CP		UN		EX		TO	
DWORD	%QD1				%QD0			
	CPUN				EXTO			
LWORD	%QL0							
	CPUNEXTO							
MSB ← Big-endian → LSB								
BYTE	%QB0	%QB1	%QB2	%QB3	%QB4	%QB5	%QB6	%QB7
	C	P	U	N	E	X	T	O
WORD	%QW0		%QW2		%QW4		%QW6	
	CP		UN		EX		TO	
DWORD	%QD0				%QD4			
	CPUN				EXTO			
LWORD	%QL0							
	CPUNEXTO							

Table 27: Example Memory Organization and Access

4. INITIAL PROGRAMMING

SIGNIFICANCE					OVERLAPPING				
Bit	Byte	Word	DWord	LWord	Byte	Word	DWord		
%QX0.7	%QB 00	%QW			%QB00	%QW			
%QX0.6									
%QX0.5									
%QX0.4									
%QX0.3									
%QX0.2									
%QX0.1									
%QX0.0									
%QX1.7	%QB 01	00			%QB01	00			
%QX1.6									
%QX1.5									
%QX1.4									
%QX1.3									
%QX1.2									
%QX1.1									
%QX1.0									
%QX2.7	%QB 02	%QW	00		%QB02	01	00		
%QX2.6									
%QX2.5									
%QX2.4									
%QX2.3									
%QX2.2									
%QX2.1									
%QX2.0									
%QX3.7	%QB 03	02			%QB03	02	01		
%QX3.6									
%QX3.5									
%QX3.4									
%QX3.3									
%QX3.2									
%QX3.1									
%QX3.0									
%QX4.7	%QB 04	%QW		00	%QB04	03		02	
%QX4.6									
%QX4.5									
%QX4.4									
%QX4.3									
%QX4.2									
%QX4.1									
%QX4.0									
%QX5.7	%QB 05	04			%QB05	04			
%QX5.6									
%QX5.5									
%QX5.4									
%QX5.3									
%QX5.2									
%QX5.1									
%QX5.0									
%QX6.7	%QB 06	%QW	04		%QB06	05			
%QX6.6									
%QX6.5									
%QX6.4									
%QX6.3									
%QX6.2									
%QX6.1									
%QX6.0									
%QX7.7	%QB 07	06			%QB07	06			
%QX7.6									
%QX7.5									
%QX7.4									
%QX7.3									
%QX7.2									
%QX7.1									
%QX7.0									

Table 28: Memory Organization and Access

4.2. Project Profiles

A project profile in the MasterTool IEC XE consists in an application template combined with a group of verification rules which guides the development of the application, reducing the programming complexity. For Nexto Xpress controllers, there is only one project profile available: Machine Profile.

The Project Profile is selected on the project creation wizard. Each project profile defines a template of standard names for the tasks and programs, which are pre-created according to the selected Project Profile. Also, during the project compilation (generate code), MasterTool IEC XE verify all the rules defined by the selected profile.

The following sections details the characteristics of each profile. It is important to note that the programming tool allows the profile change from an existent project (see project update section in the MasterTool IEC XE User Manual – MU299609), but it's up to the developer to make any necessary adjustments so that the project becomes compatible with the rules of the new selected profile.

ATTENTION:
Through the description of the Project profiles some tasks types are mentioned, which are described in the section 'Task Configuration', of the MasterTool IEC XE User Manual – MU299609.

4.2.1. Machine Profile

In the Machine Profile, by default, the application has a user task of the Cyclic type called MainTask. This task is responsible for implementing a single Program type POU called MainPrg. This program can call other programming units of the Program, Function or Function Block types, but any user code will run exclusively by MainTask task.

This profile is characterized by allowing shorter intervals in the MainTask, allowing faster execution of user code. This profile may further include an interruption task, called TimeInterruptTask00, with a higher priority than the MainTask, and hence, can interrupt its execution at any time.

Task	POU	Priority	Type	Interval	Event
MainTask	MainPrg	13	Cyclic	20 ms	-
TimeInterruptTask00	TimeInterruptPrg00	01	Cyclic	4 ms	-

Table 29: Machine Profile Tasks

Also, this profile supports the inclusion of additional tasks associated to counter and external interruptions, resulting in a maximum of 5 tasks for user application.

ATTENTION:
The suggested POU names associated with the tasks are not consisted. They can be changed, as long as they are also changed in the tasks configurations.

4.3. CPU Configuration

The controller's CPU configuration is located in the device tree, as shown on the figure below, and can be accessed by a double-click on the corresponding object. In this tab it's possible to configure watchdog behavior, clock synchronism, among other parameters, as described on section [Controller's CPU](#).

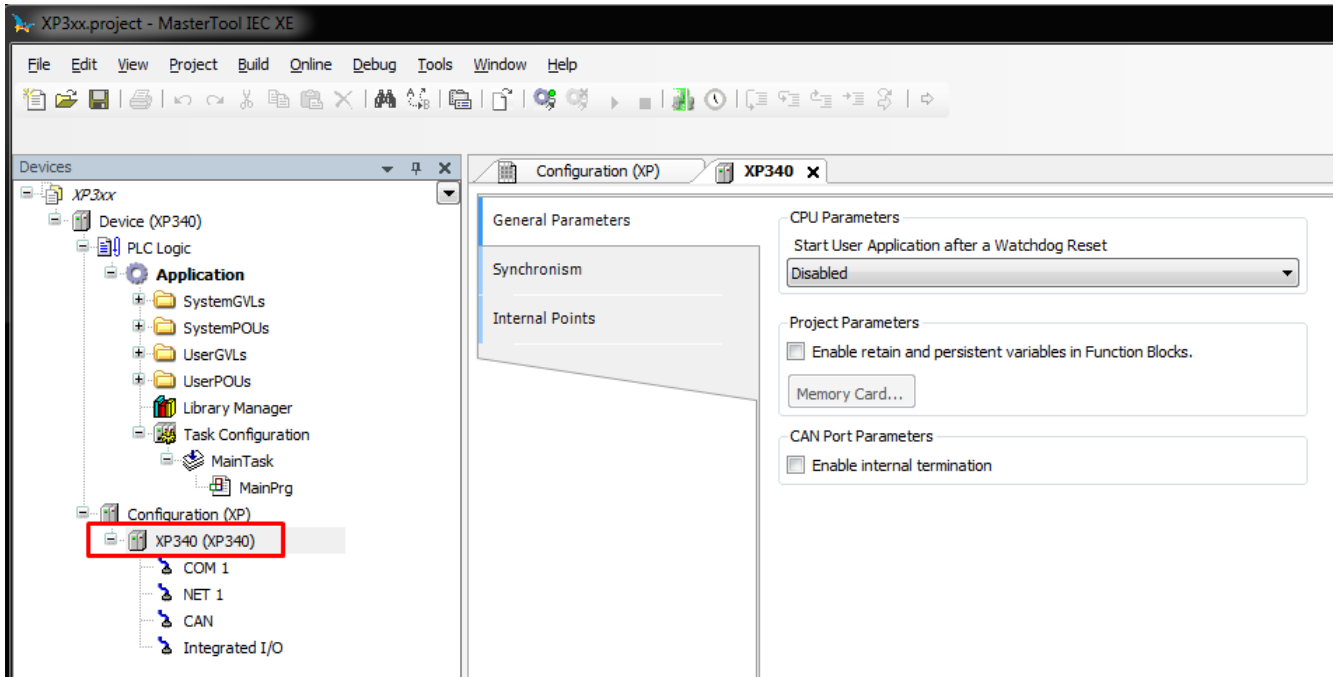


Figure 10: CPU Configuration

Besides that, by double-clicking on controller's NET 1 icon, it's possible to configure the Ethernet interface that will be used for communication between the controller and the software MasterTool IEC XE.

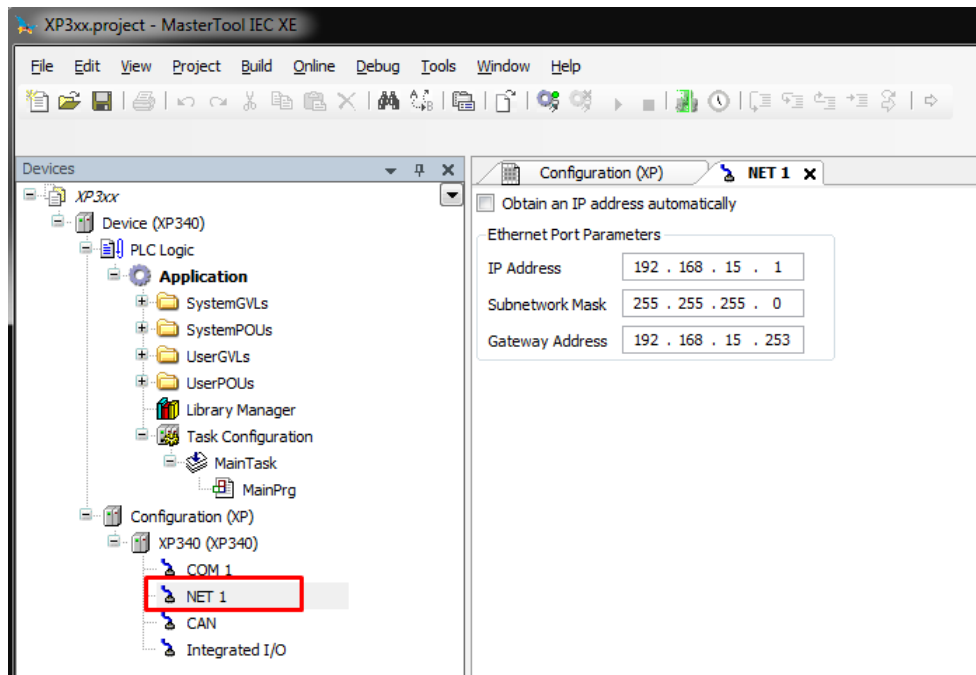


Figure 11: Configuring the Communication Port

The configuration defined on this tab will be applied to the device only when sending the application to the device (download), which is described further on sections [Finding the Device](#) and [Login](#).

Additionally, the device tree also offers the configuration of the integrated I/O available on Nexto Xpress controllers, as shown on the figure below. In this tab it is possible to configure digital inputs filters, the mode of each analog input, among other parameters.

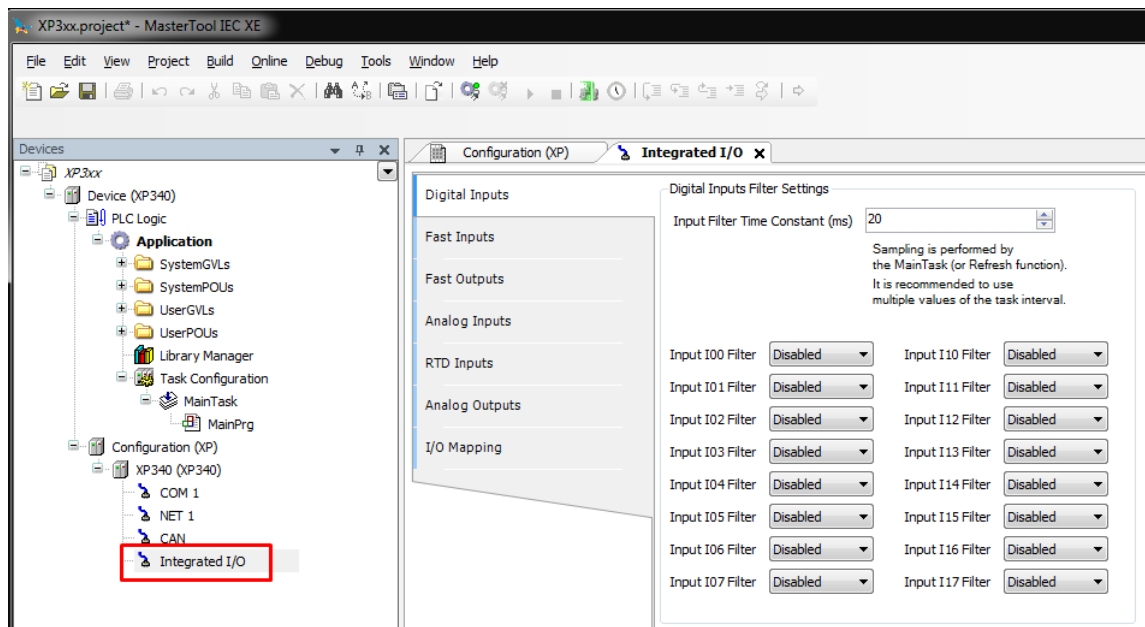


Figure 12: Configuring the Integrated I/O

4.4. Libraries

There are several programming tool resources which are available through libraries. Therefore, these libraries must be inserted in the project so its utilization becomes possible. The insertion procedure and more information about available libraries may be found in the MasterTool Programming Manual – MP399609.

4.5. Inserting a Protocol Instance

The Nexto Xpress controllers, as described on General Features table, offers several communication protocols. Except for the OPC communication, which have a different configuration procedure, the insertion of a protocol can be done by simply right-clicking on the desired communication interface, selecting to add the device and finally performing the configuration as shown in the [Communication Protocols](#) section. Below are presented some examples.

4.5.1. MODBUS Ethernet

The first step to configure the MODBUS Ethernet (Server in this example) is to include the instance in the desired NET (in this case, NET 1, as the XP3xx has only one Ethernet interface). Click on the NET with the mouse right button and select Add Device..., as shown on Figure 13:

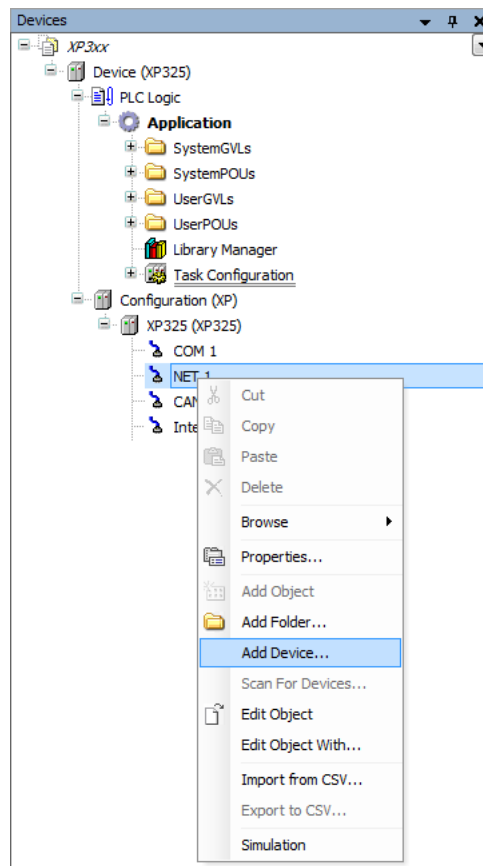


Figure 13: Adding the Instance

4. INITIAL PROGRAMMING

After that, the list of protocols will appear on the screen. Simply select MODBUS Symbol Server as described on the figure below:

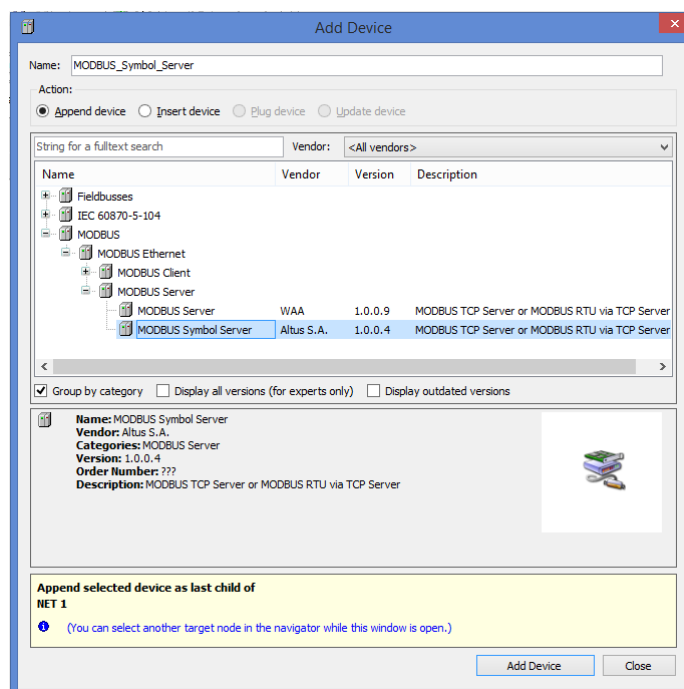


Figure 14: Selecting the Protocol

4.6. Finding the Device

To establish the communication between the controller and MasterTool IEC XE, first it's necessary to find and select the desired device. The configuration of this communication is located on the object *Device* on device tree, on *Communication Settings* tab. On this tab, after selecting the *Gateway* and clicking on button *Scan network*, the software MasterTool IEC XE performs a search for devices and shows the controllers found on the network of the Ethernet interface of the station where the tool is running.

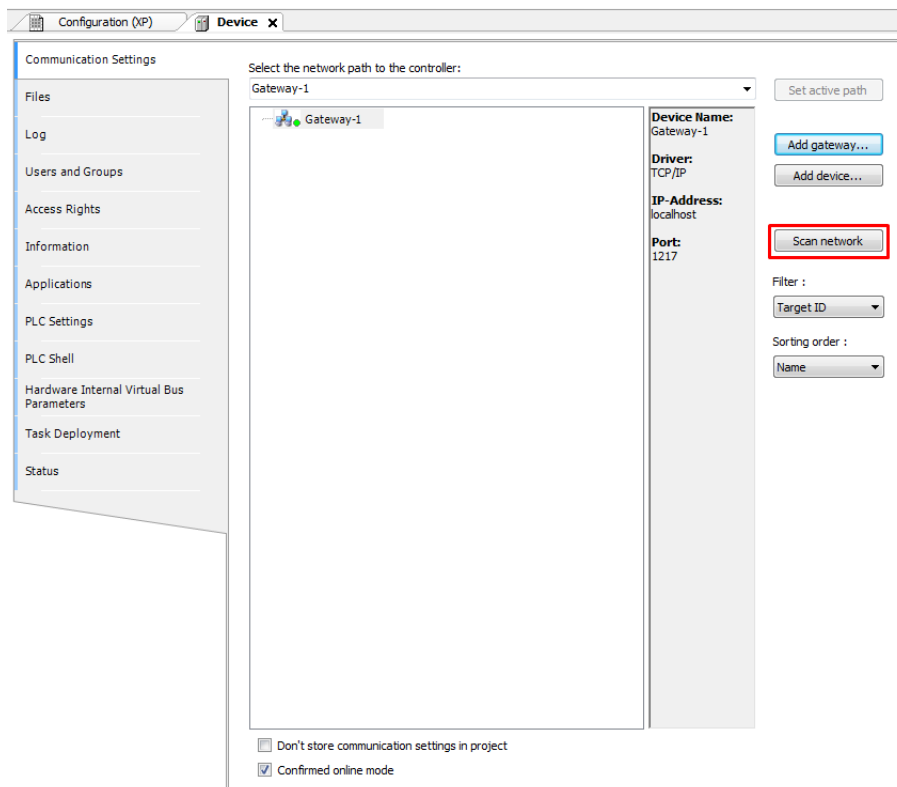


Figure 15: Finding the Device

If there is no gateway previously configured, it can be included by the button *Add gateway*, using the default IP address *localhost* to use the gateway resident on the station or changing the IP address to use the device internal gateway.

Next, the desired controller must be selected by clicking on *Set active path* clicked. This action selects the controller and informs the configuration software which controller shall be used to communicate and send the project.

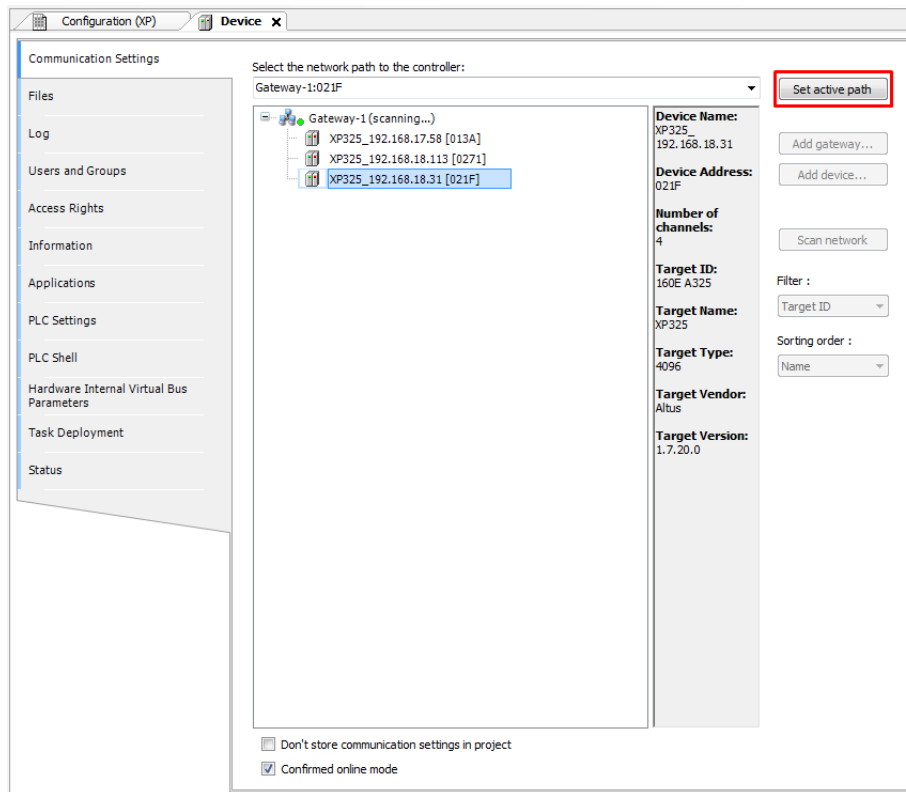


Figure 16: Selecting the controller

Additionally, the user can change the default name of the device that is displayed. For that, you must click the right mouse button on the desired device and select *Change Node Name*. After a name change, the device will not return to the default name under any circumstances.

In case the Ethernet configuration of the controller to be connected is in a different network from the Ethernet interface of the station, the software MasterTool IEC XE will not be able to find the device. In this case, it's recommended to use the command *Easy Connection* located on Online menu.

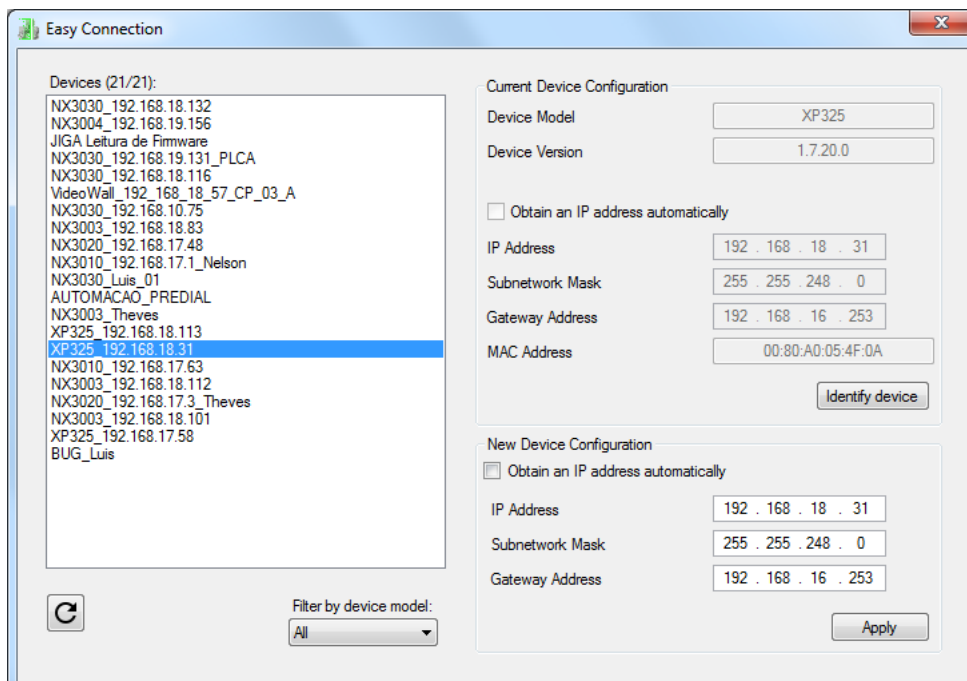


Figure 17: Easy Connection

This command performs a MAC level communication with the device, allowing to permanently change the configuration of the controller's Ethernet interface, independently of the IP configuration of the station and from the one previously configured on the device. So, with this command, it's possible to change the device configuration to put it on the same network of the Ethernet interface of the station where MasterTool IEC XE is running, allowing to find and select the device for the communication. The complete description of Easy Connection command can be found on User Manual of MasterTool IEC XE code MU299609.

4.7. Login

After compiling the application and fixing errors that might be found, it's time to send the project to the controller. To do this, simply click on *Login* command located on *Online* menu of MasterTool IEC XE as shown on the following figure. This operation may take a few seconds, depending on the size of the generated file.

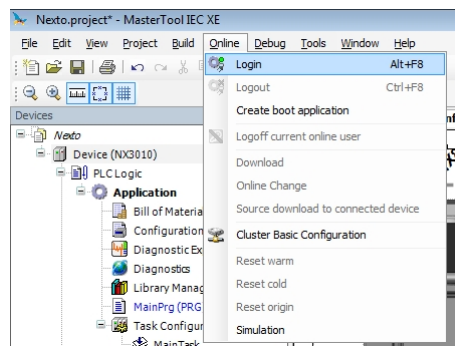


Figure 18: Sending the Project to the controller

After the command execution, some user interface messages may appear, which are presented due to differences between an old project and the new project been sent, or simply because there was a variation in some variable.

If the Ethernet configuration of the project is different from the device, the communication may be interrupted at the end of download process when the new configuration is applied on the device. So, the following warning message will be presented, asking the user to proceed or not with this operation:

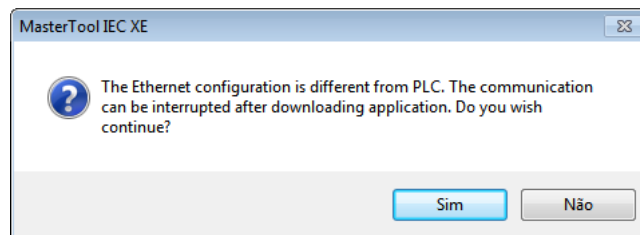


Figure 19: IP Configuration Warning

If there is no application on the controller, the following message will be presented.



Figure 20: No application on the device

If there is already an application on the controller, depending on the differences between the projects, the following options will be presented:

- Login with online change: execute the login and send the new project without stopping the current controller application (see [Run Mode](#) item), updating the changes when a new cycle is executed
- Login with download: execute the login and send the new project with the controller stopped (see [Stop Mode](#) item). When the application is initiated, the update will have been done already
- Login without any change: executes the login without sending the new project

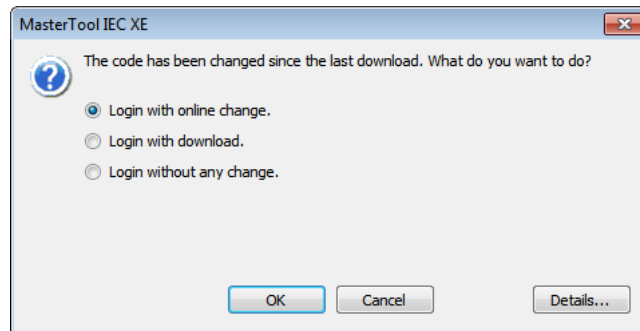


Figure 21: New application download

ATTENTION:

In the online changes is not permitted to associate symbolic variables mapping from a global variable list (GVL) and use these variables in another global variable list (GVL).

If the new application contains changes on the configuration, the online change will not be possible. In this case, the MasterTool IEC XE requests whether the login must be executed as download (stopping the application) or if the operation must be canceled, as shown on the following figure.

PS.: The button *Details...* shows the changes made in the application.

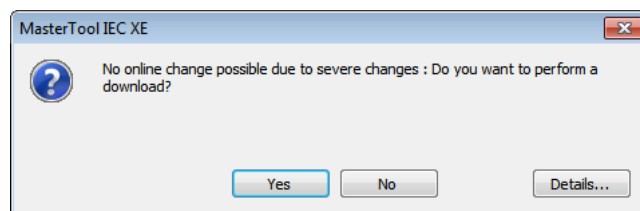


Figure 22: Configuration change

Finally, at the end of this process the MasterTool IEC XE offers the option to transfer (download) the source code to the internal memory of the device, as shown on the following figure:

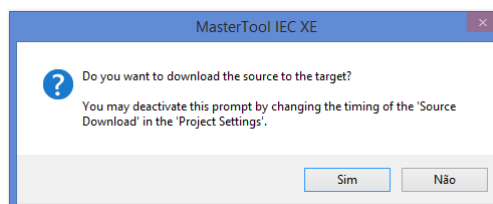


Figure 23: Source code download

Transferring the source code is fundamental to ensure the future restoration of the project and to perform modifications on the application that is loaded into the device.

4.8. Run Mode

Right after the project has been sent to the controller, the application will not be immediately executed (except for the case of an online change). For that to happen, the command *Start* must be executed. This way, the user can control the execution of the application sent to the controller, allowing to pre-configure initial values which will be used by the controller on the first execution cycle.

To execute this command, simply go to the Debug menu and select the option *Start*, as shown on Figure 24.

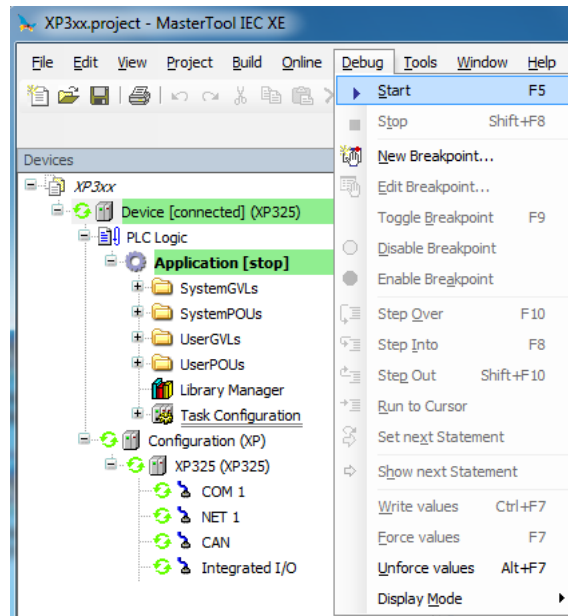


Figure 24: Starting the Application

Figure 25 shows the application in execution. In case the POU tab is selected, the created variables are listed on a monitoring window, in which the values can be visualized and forced by the user.

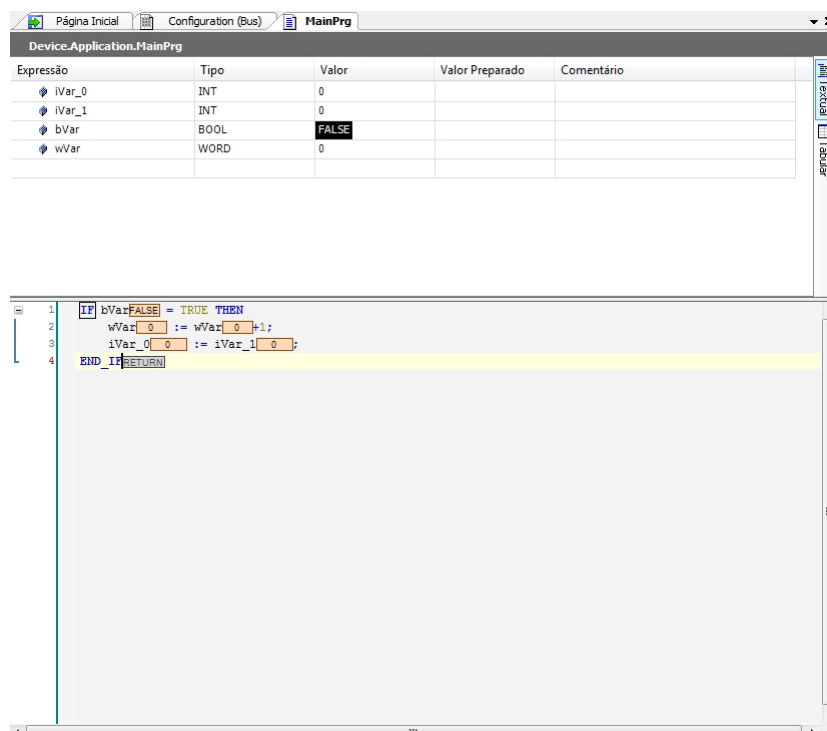


Figure 25: Program running

If the controller already have a boot application internally stored, it goes automatically to *Run Mode* when the device is powered on, with no need for an online command through MasterTool IEC XE.

4.9. Stop Mode

To stop the execution of the application, the user must execute the *Stop* command, available at the menu *Debug*, as shown on Figure 26.

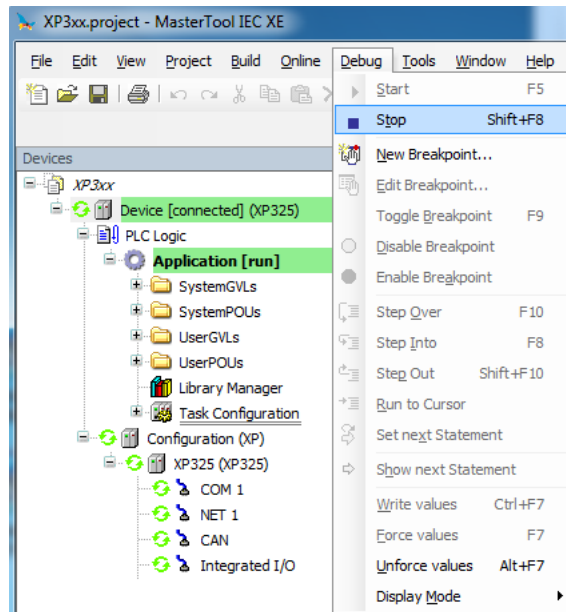


Figure 26: Stopping the Application

In case the controller is initialized without the stored application, it automatically goes to *Stop Mode*, as it happens when a software exception occurs.

4.10. Writing and Forcing Variables

After Logging into a PLC, the user can write or force values to a variable of the project.

The writing command (*CTRL + F7*) writes a value into a variable and this value could be overwritten by instructions executed in the application.

Moreover, the forced writing command (*F7*) writes a value into a variable without allowing this value to be changed until the forced variables be released.

It is important to highlight that, when using the MODBUS RTU Slave and the MODBUS Ethernet Server, and the *Read-only* option from the configured relations is not selected, the forced writing command (*F7*) must be done over the available variables in the monitoring window as the writing command (*CTRL + F7*) leaves the variables to be overwritten when new readings are done.

ATTENTION:

The variables forcing can be done only in Online mode.

Diagnostic variables cannot be forced, only written, because diagnostics are provided by the controller and will be overwritten by it.

ATTENTION:

When a controller is with forced variables and it is de-energized, the variables will lose the forcing in the next initialization.

The limit of forcing for all models of Nexto controllers is 128 variables.

4.11. Logout

To finalize the online communication with the controller, the command *Logout* must be executed, located in the *Online* menu, as shown on Figure 27.

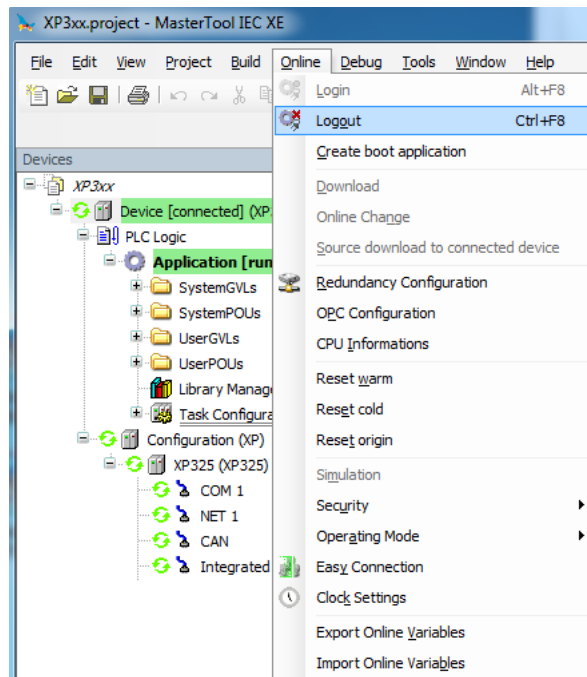


Figure 27: Ending the online communication with the controller

4.12. Project Upload

Nexto Xpress controllers are capable to store the source code of the application on the internal memory of the device, allowing future retrieval (*upload*) of the complete project and to modify the application.

To recover a project previously stored on the internal memory of the controller, the command located on *File* menu must be executed as shown on the following figure.

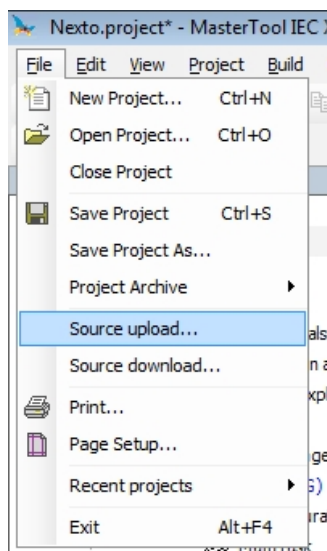


Figure 28: Project Upload Option

Next, just select the desired controller and click *OK* as shown on Figure 29.

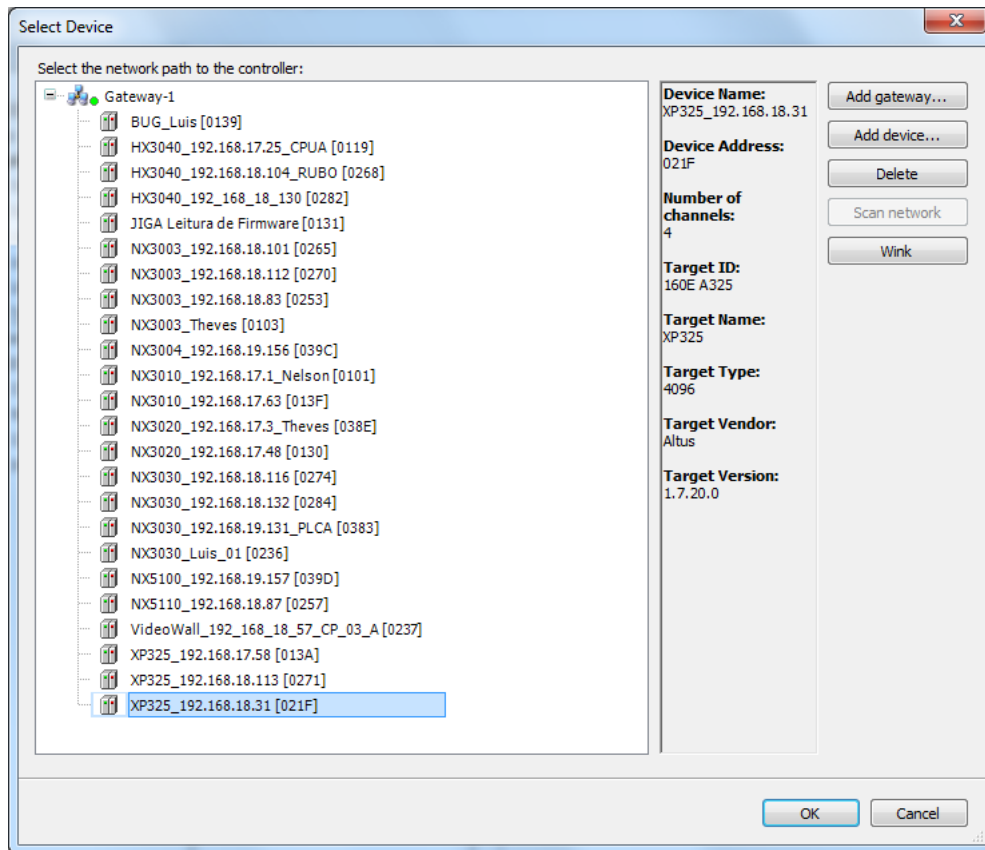


Figure 29: Selecting the controller

To ensure that the project loaded in the controller is identical and can be accessed in other workstations, consult the chapter **Projects Download/Login Method without Project Differences** at the MasterTool IEC XE User Manual MT8500 - MU299609.

ATTENTION:
The memory size area to store a project in the Nexto Xpress controller is defined on [General Features](#) table.

ATTENTION:
The Upload recovers the last project stored in the controller as described in the previous paragraphs. In case only the application was downloaded, without transferring its source code, it will not be possible for it to be recovered by the *Upload* procedure.

4.13. CPU Operating States

4.13.1. Run

When the controller is in Run mode, all application tasks are executed.

4.13.2. Stop

When a CPU is in Stop mode, all application tasks are stopped. The variable values in the tasks are kept with the current value and output points go to the safe state.

When a CPU goes to the Stop mode due to the download of an application, the variables in the application tasks will be lost except the persistent variables type.

4.13.3. Breakpoint

When a debugging mark is reached in a task, it is interrupted. All the active tasks in the application will not be interrupted, continuing their execution. With this feature, it's possible to go through and investigate the program flow step by step in Online mode according to the positions of the interruptions included through the editor.

For further information about the use of breakpoints, please consult the MasterTool IEC XE Utilization Manual - MU299609.

4.13.4. Exception

When a CPU is in *Exception* it indicates that some improper operation occurred in one of the application active tasks. The task which caused the *Exception* will be suspended and the other tasks will pass for the *Stop* mode. It is only possible to take off the tasks from this state and set them in execution again after a new CPU start condition. Therefore, only with a *Reset Warm*, *Reset Cold*, *Reset Origin* or a CPU restart puts the application again in *Run* mode.

4.13.5. Reset Warm

This command puts the CPU in *Stop* mode and initializes all the application tasks variables, except the persistent and retentive type variables. The variables initialized with a specific value will assume exactly this value, the other variables will assume the standard initialization value (zero).

4.13.6. Reset Cold

This command puts the CPU in *Stop* mode and initializes all the application tasks variables, except the persistent type variables. The variables initialized with a specific value will assume exactly this value, the other variables will assume the standard initialization value (zero).

4.13.7. Reset Origin

This command removes all the application tasks variables, including the persistent type variables and deletes the CPU application.

Notes:

Reset: When a Reset is executed, the breakpoints defined in the application are disabled.

Command: To execute the commands *Reset Warm*, *Reset Cold* or *Reset Origin*, is necessary to have MasterTool IEC XE in *Online* mode with the controller.

4.13.8. Reset Process Command (IEC 60870-5-104)

This process reset command can be solicited by IEC 60870-5-104 clients. After answer the client, the CPU start a rebooting process, as if being done an energizing cycle.

The standard IEC 60870-5-104 foresee a qualification value pass (0..255) with the process reset command, but this *parameter* is not considered by the CPU.

4.14. Programs (POUs) and Global Variable Lists (GVLs)

The project created by MasterTool IEC XE contains a set of program modules (POUs) and global variables lists that aims to facilitate the programming and utilization of the controller. The following sections describes the main elements that are part of this standard project structure.

4.14.1. MainPrg Program

The MainTask task is associated to one unique POU of program type, named MainPrg. The MainPrg program is created automatically and cannot be edited by user.

The MainPrg program code is the following, in ST language:

```
(*Main POU associated with MainTask that calls StartPrg,
UserPrg/ActivePrg and NonSkippedPrg.
This POU is blocked to edit.*)

PROGRAM MainPrg
VAR
    isFirstCycle : BOOL := TRUE;
END_VAR

IF isFirstCycle THEN
    StartPrg();
    isFirstCycle := FALSE;
ELSE
    UserPrg();
END_IF;
```

MainPrg call other two POU's of program type, named *StartPrg* and *UserPrg*. While the *UserPrg* is always called, the *StartPrg* is only called once in the PLC application start.

Differently from the *MainPrg* program, that must not be modified, the user can change the *StartPrg* and *UserPrg* programs. Initially, when the project is created from the wizard, these two programs are created *empty*, but the user might insert code in them.

4.14.2. StartPrg Program

In this POU the user might create logics, loops, start variables, etc. that will be executed only one time in the first PLC's cycle, before execute *UserPrg* POU by the first time. And not being called again during the project execution.

In case the user load a new application, or if the PLC gets powered off, as well as in *Reset Origin*, *Reset Cold* and *Reset Warm* conditions, this POU is going to be executed again.

4.14.3. UserPrg Program

In this POU the user must create the main application, responsible by its own process control. This POU is called by the main POU (MainPrg).

The user can also create additional POU's (programs, functions or function blocks), and called them or instance them inside UserPrg POU, to ends of its program instruction. Also it is possible to call functions and instance function blocks defined in libraries.

4.14.4. GVL IntegratedIO

The GVL *IntegratedIO* contains the variables correspondent to the physical input and output channels integrated into the controller.

The following picture shows an example of the presentation of this GVL when in *Online* mode.

Device.Application.IntegratedIO				
Expression	Type	Value	Prepared value	Address
[-] DigitalInputs	T_DIGITAL_INPUTS_1			
100	BOOL	FALSE		
101	BOOL	FALSE		
102	BOOL	FALSE		
103	BOOL	FALSE		
104	BOOL	FALSE		
105	BOOL	FALSE		
106	BOOL	FALSE		
107	BOOL	FALSE		
110	BOOL	FALSE		
111	BOOL	FALSE		
112	BOOL	FALSE		
113	BOOL	FALSE		
114	BOOL	FALSE		
115	BOOL	FALSE		
116	BOOL	FALSE		
117	BOOL	FALSE		
[-] DigitalOutputs	T_DIGITAL_OUTPUT...			
Q00	BOOL	FALSE		
Q01	BOOL	FALSE		
Q02	BOOL	FALSE		
Q03	BOOL	FALSE		
Q04	BOOL	FALSE		
Q05	BOOL	FALSE		
Q06	BOOL	FALSE		
Q07	BOOL	FALSE		
Q10	BOOL	FALSE		
Q11	BOOL	FALSE		
Q12	BOOL	FALSE		
Q13	BOOL	FALSE		
Q14	BOOL	FALSE		
Q15	BOOL	FALSE		
Q16	BOOL	FALSE		
Q17	BOOL	FALSE		
[+] FastInputs	T_FAST_INPUTS_1			
[+] FastOutputs	T_FAST_OUTPUTS_1			
[-] AnalogInputs	T_ANALOG_INPUTS_1			
AI0	INT	0		
AI1	INT	0		
AI2	INT	0		
AI3	INT	0		
AI4	INT	0		
[-] AnalogOutputs	T_ANALOG_OUTPUT...			
AO0	INT	0		
AO1	INT	0		
AO2	INT	0		
AO3	INT	0		
[-] RtdInputs	T_RTD_INPUTS_1			
RI0	INT	0		
RI1	INT	0		

Figure 30: IntegratedIO GVL in Online Mode

4.14.5. GVL System_Diagnostics

The *System_Diagnostics* GVL contains the diagnostic variables of the controller's CPU, communication and I/O interfaces. This GVL isn't editable and the variables are declared automatically with type specified by the device to which it belongs when it is added to the project.

ATTENTION:

In *System_Diagnostics* GVL, are also declared the diagnostic variables of the direct representation MODBUS Client/Master requisitions.

The following picture shows an example of the presentation of this GVL when in *Online* mode.

Device.Application.System_Diagnostics				
Expression	Type	Value	Prepared value	Address
DIG_XP325	T_DIAG_...			
tSummarized	T_DIAG_...			
bHardwareFailure	BIT	FALSE		
bSoftwareException	BIT	FALSE		
bCOM1ConfigError	BIT	FALSE		
bNET1ConfigError	BIT	FALSE		
bInvalidDateTime	BIT	FALSE		
bRuntimeReset	BIT	FALSE		
bRetentivityError	BIT	FALSE		
bIntegratedIODiagnostic	BIT	FALSE		
tDetailed	T_DIAG_...			
Target	T_DIAG_...			
Hardware	T_DIAG_...			
Exception	T_DIAG_...			
RetainInfo	T_DIAG_...			
Reset	T_DIAG_...			
Serial	T_DIAG_...			
CAN	T_DIAG_...			
USB	T_DIAG_...			
Ethernet	T_DIAG_...			
UserFiles	T_DIAG_...			
UserLogs	T_DIAG_...			
Application	T_DIAG_...			
ApplicationInfo	T_DIAG_...			
SNTP	T_DIAG_...			
IntegratedIO	T_DIAG_I...			
AnalogInputs	T_DIAG_...			
AnalogOutputs	T_DIAG_...			
RTDInputs	T_DIAG_...			

Figure 31: System_Diagnostics GVL in Online Mode

4.14.6. GVL Disables

The *Disables* GVL contains the MODBUS Master/Client by symbolic mapping requisition disabling variables. It is not mandatory, but it is recommended to use the automatic generation of these variables, which is done clicking in the button *Generate Disabling Variables* in device requisition tab. These variables are declared as type BOOL and follow the following structure:

Requisition disabling variables declaration:

```
[Device Name]_DISABLE_[Requisition Number] : BOOL;
```

Where:

Device name: Name that shows on TreeView to the MODBUS device.

Requisition Number: Requisition number that was declared on the MODBUS device requisition table following the sequence from up to down, starting on 0001.

Example:

Device.Application.Disables

```
VAR_GLOBAL
MODBUS_Device_DISABLE_0001 : BOOL;
MODBUS_Device_DISABLE_0002 : BOOL;
MODBUS_Device_DISABLE_0003 : BOOL;
MODBUS_Device_1_DISABLE_0001 : BOOL;
MODBUS_Device_1_DISABLE_0002 : BOOL;
END_VAR
```

The automatic generation through button *Generate Disabling Variables* only create variables, and don't remove automatically. This way, in case any relation is removed, its respective disabling variable must be removed manually.

The *Disables* GVL is editable, therefore the requisition disabling variables can be created manually without need of following the model created by the automatic declaration and can be used both ways at same time, but must always be of BOOL type. And it is need to take care to do not delete or change the automatic declared variables, cause them can being used for some MODBUS device. If the variable be deleted or changed then an error is going to be generated while the project is being compiled. To correct the automatically declared variable name, it must be followed the model exemplified above according to the device and the requisition to which they belong.

The following picture shows an example of the presentation of this GVL when in *Online* mode. If the variable values are TRUE it means that the requisition to which the variables belongs is disabled and the opposite is valid when the variable value is FALSE.









Device.Application.Disables			
Expression	Type	Value	Prepared
 MODBUS_Slave_1_DISABLE_0001	BOOL	FALSE	
 MODBUS_Slave_1_DISABLE_0002	BOOL	TRUE	
 MODBUS_Slave_1_DISABLE_0003	BOOL	FALSE	
 MODBUS_Slave_1_DISABLE_0004	BOOL	FALSE	
 MODBUS_Server_1_DISABLE_0001	BOOL	FALSE	
 MODBUS_Server_1_DISABLE_0002	BOOL	FALSE	
 MODBUS_Server_1_DISABLE_0003	BOOL	FALSE	
 MODBUS_Server_1_DISABLE_0004	BOOL	TRUE	

Figure 32: Disable GVL in Online Mode

4.14.7. GVL Qualities

The *Qualities* GVL contains the quality variable of the internal variables MODBUS Master/Client of symbolic mapping . It is not mandatory but is recommended to use these variables' automatic generation, what is done clicking on button *Generate Quality Variables* in the device mapping tab. These variables are declared as *LibDataTypes.QUALITY* type and follow the following structure:

Quality mapping variable declaration:

```
[Device Name]_QUALITY_[Mapping Number]: LibDataTypes.QUALITY;
```

Where:

Device Name: Name that appear at the Tree View to the device.

Mapping Number: Number of the mapping that was declared on the device mapping table, following the up to down sequence, starting with 0001.

ATTENTION:

It is not possible to associate quality variables to the direct representation MODBUS Master/Client drivers' mappings. Therefor it is recommended the use of symbolic mapping MODBUS drivers.

Examples: Device.Application.Qualities

```
VAR_GLOBAL
MODBUS_Device_QUALITY_0001: LibDataTypes.QUALITY;
MODBUS_Device_QUALITY_0002: LibDataTypes.QUALITY;
MODBUS_Device_QUALITY_0003: LibDataTypes.QUALITY;
END_VAR
```

The *Quality* GVL, is editable, therefore the mapping quality variables can be created manually without need to follow the automatic declaration model, and can be used both ways at same time. But must always be of *LibDataTypes.QUALITY* type and take care to don't delete or change a variable automatically declared, because they might being used by some device. If the variable be deleted or changed an error is going to be generated while the project is being compiled. To correct the automatically declared variable name, it must be followed the model exemplified above according to the device and the requisition to which they belong.

To the MODBUS communication devices the quality variables behave on the way showed at Table 37.

ATTENTION:

If a symbolic mapping MODBUS Client/Master driver's variable be mapped in Server IEC 60870-5-104 driver, it is necessary that the MODBUS mapping quality variables had been created to generate valid quality events to such Server IEC 60870-5-104 points. Case opposite, aren't going to be generated "bad" quality events to Server IEC60870-5-104 clients in the situations that MODBUS Master/Client can't communicate with its slaves/servers, by example.

The following picture shows an example of the presentation of this GVL when in *Online* mode.

Device.Application.Qualities				
Expression	Type	Value	Address	Comment
MODBUS_Slave_1_QUALITY_0001	LibDataTypes.QUALITY			
VALIDITY	QUALITY_VALIDITY	VALIDITY_GOOD		Quality validity
FLAGS	QUALITY_FLAGS			Quality flags
FLAG_OUT_OF_RANGE	BIT	FALSE		Bit 8
FLAG_INACCURATE	BIT	FALSE		Bit 9
FLAG_OLD_DATA	BIT	FALSE		Bit 10
FLAG_FAILURE	BIT	FALSE		Bit 11
FLAG_OPERATOR_BLOCKED	BIT	FALSE		Bit 12
FLAG_TEST	BIT	FALSE		Bit 13
FLAG_RESERVED_0	BIT	FALSE		Bit 14
FLAG_RESERVED_1	BIT	FALSE		Bit 15
FLAG_RESTART	BIT	FALSE		Bit 0
FLAG_COMM_FAIL	BIT	FALSE		Bit 1
FLAG_REMOTE_SUBSTITU...	BIT	FALSE		Bit 2
FLAG_LOCAL_SUBSTITUTED	BIT	FALSE		Bit 3
FLAG_FILTER	BIT	FALSE		Bit 4
FLAG_OVERFLOW	BIT	FALSE		Bit 5
FLAG_REFERENCE_ERROR	BIT	FALSE		Bit 6
FLAG_INCONSISTENT	BIT	FALSE		Bit 7
MODBUS_Slave_1_QUALITY_0002	LibDataTypes.QUALITY			
MODBUS_Slave_1_QUALITY_0003	LibDataTypes.QUALITY			
MODBUS_Slave_1_QUALITY_0004	LibDataTypes.QUALITY			
MODBUS_Server_1_QUALITY_0001	LibDataTypes.QUALITY			
MODBUS_Server_1_QUALITY_0002	LibDataTypes.QUALITY			
MODBUS_Server_1_QUALITY_0003	LibDataTypes.QUALITY			
VALIDITY	QUALITY_VALIDITY	VALIDITY_QUESTIONABLE		Quality validity
FLAGS	QUALITY_FLAGS			Quality flags
FLAG_OUT_OF_RANGE	BIT	FALSE		Bit 8
FLAG_INACCURATE	BIT	FALSE		Bit 9
FLAG_OLD_DATA	BIT	TRUE		Bit 10
FLAG_FAILURE	BIT	FALSE		Bit 11
FLAG_OPERATOR_BLOCKED	BIT	FALSE		Bit 12
FLAG_TEST	BIT	FALSE		Bit 13
FLAG_RESERVED_0	BIT	FALSE		Bit 14
FLAG_RESERVED_1	BIT	FALSE		Bit 15
FLAG_RESTART	BIT	FALSE		Bit 0
FLAG_COMM_FAIL	BIT	TRUE		Bit 1
FLAG_REMOTE_SUBSTITU...	BIT	FALSE		Bit 2
FLAG_LOCAL_SUBSTITUTED	BIT	FALSE		Bit 3
FLAG_FILTER	BIT	FALSE		Bit 4
FLAG_OVERFLOW	BIT	FALSE		Bit 5
FLAG_REFERENCE_ERROR	BIT	FALSE		Bit 6
FLAG_INCONSISTENT	BIT	FALSE		Bit 7
MODBUS_Server_1_QUALITY_0004	LibDataTypes.QUALITY			

Figure 33: GVL Qualities in Online Mode

4.14.8. GVL ReqDiagnostics

In *ReqDiagnostics GVL*, are declared the requisition diagnostics variables of symbolic mapping MODBUS Master/Client. It is nor mandatory, but recommended the use of these variables' automatic generation, what is done by clicking in the button *Generate Diagnostic Variables* in device requisitions tab. These variables declaration follow the following structure:

Requisition diagnostic variable declaration:

```
[Device Name]_REQDG_[Requisition Number]: [Variable Type];
```

Where:

Device Name: Name that appear at the TreeView to the device.

Mapping Number: Number of the mapping that was declared on the device mapping table, following the up to down sequence, starting with 0001.

Variable Type: NXMODBUS_DIAGNOSTIC_STRUCTS.

T_DIAG_MODBUS_RTU_MAPPING_1 to MODBUS Master and
NXMODBUS_DIAGNOSTIC_STRUCTS.

T_DIAG_MODBUS_ETH_MAPPING_1 to MODBUS Client.

ATTENTION:

The requisition diagnostics variables of direct mapping MODBUS Master/Client are declared at *System_Diagnostics* GVL.

Example:

Device.Application.ReqDiagnostics

```
VAR_GLOBAL
MODBUS_Device_REQDG_0001 : NXMODBUS_DIAGNOSTIC_STRUCTS.
                           T_DIAG_MODBUS_RTU_MAPPING_1;
MODBUS_Device_REQDG_0002 : NXMODBUS_DIAGNOSTIC_STRUCTS.
                           T_DIAG_MODBUS_RTU_MAPPING_1;
MODBUS_Device_REQDG_0003 : NXMODBUS_DIAGNOSTIC_STRUCTS.
                           T_DIAG_MODBUS_RTU_MAPPING_1;
MODBUS_Device_1_REQDG_0001 : NXMODBUS_DIAGNOSTIC_STRUCTS.
                             T_DIAG_MODBUS_ETH_MAPPING_1;
MODBUS_Device_1_REQDG_0002 : NXMODBUS_DIAGNOSTIC_STRUCTS.
                             T_DIAG_MODBUS_ETH_MAPPING_1;
END_VAR
```

The *ReqDiagnostics* GVL is editable, therefore the requisitions diagnostic variables can be manually created without need to follow the model created by the automatic declaration. Both ways can be used at same time, but the variables must always be of type referring to the device. And take care to don't delete or change a variable automatically declared, because they might being used by some device. If the variable be deleted or changed an error is going to be generated while the project is being compiled. To correct the automatically declared variable name, it must be followed the model exemplified above according to the device and the requisition to which they belong.

The following picture shows an example of the presentation of this GVL when in *Online* mode.

4. INITIAL PROGRAMMING

Device.Application.ReqDiagnostics		
Expression	Type	Value
MODBUS_Slave_1_REQDG_0001	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
byStatus	T_DIAG_MODBUS_RTU_MAPPING_STATUS	
eLastErrorCode	MASTER_ERROR_CODE	NO_ERROR
eLastExceptionCode	MODBUS_EXCEPTION	NO_EXCEPTION
byDiag_3_reserved	BYTE	0
wCommCounter	WORD	969
wCommErrorCounter	WORD	0
MODBUS_Slave_1_REQDG_0002	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
MODBUS_Slave_1_REQDG_0003	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
MODBUS_Slave_1_REQDG_0004	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
MODBUS_Server_1_REQDG_0001	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
MODBUS_Server_1_REQDG_0002	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
MODBUS_Server_1_REQDG_0003	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	
byStatus	T_DIAG_MODBUS_ETH_MAPPING_STATUS	
eLastErrorCode	MASTER_ERROR_CODE	ERR_CONNECTION_TIMEOUT
eLastExceptionCode	MODBUS_EXCEPTION	NO_EXCEPTION
byDiag_3_reserved	BYTE	0
wCommCounter	WORD	116
wCommErrorCounter	WORD	49
MODBUS_Server_1_REQDG_0004	NXMODBUS_DIAGNOSTIC_STRUCTS.T_DIAG_MODBUS...	

Figure 34: ReqDiagnostics GVL in Online Mode

5. Configuration

The Nexto Xpress controllers are configured and programmed through the MasterTool IEC XE software. The configuration defines the behavior and utilization modes for peripherals use and special features of the controller. The programming represents the application developed by the user, also known as *Application*.

5.1. Controller's CPU

5.1.1. General Parameters

The parameters related to the controller's CPU are located at project treeview on item XP3xx just below *Configuration*. Each item must be properly verified for the correct project execution. These parameters are described below:

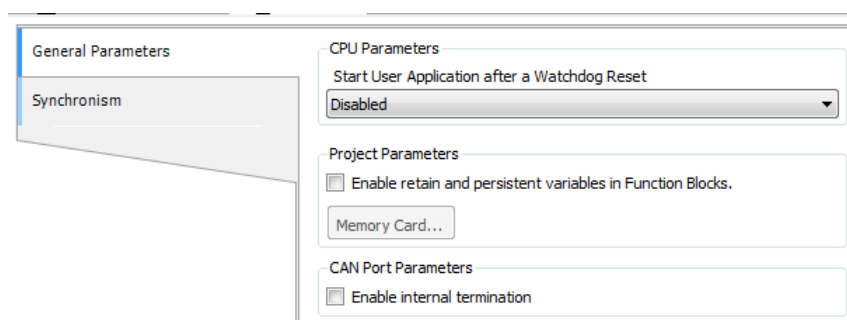


Figure 35: CPU configuration

Configuration	Description	Default	Options
CPU Parameters			
Start User Application After a Watchdog Reset	When enabled starts the user application after the hardware watchdog reset or through the Runtime restart, but keeps the diagnostics indication via LED DG and via variables	Disabled	Enabled Disabled
Enable retain and persistent variables in Function Blocks	Configuration to allow the use of retain and persistent variables on Function Blocks	Unmarked	Marked: allows the use of retain and persistent variables on Function Blocks. Unmarked: If this is done with this option unmarked, it may occur an exception error on startup.
Enable internal termination	When enabled use internal termination on CAN interface.	Unmarked	Marked: enabled internal termination on CAN interface. Unmarked: disabled internal termination on CAN interface.

Table 30: CPU Configuration

5.1.2. Time Synchronization

For the time synchronization, Nexto Xpress controllers use the SNTP (Simple Network Time Protocol) protocol or the synchronism through IEC 60870-5-104.

To use the time sync protocols, the user must set the following parameters at *Synchronism* tab located at *CPU configuration* on project treeview.

Figure 36: SNTP Configuration

Configuration	Description	Default	Options
Time zone (hh:mm)	Time zone of the user location. Hours and minutes can be inserted.	-3:00	-12:59 to +13:59
SNTP Service	Enables the SNTP service.	Disabled	Disabled or Enabled
Period for SNTP Synchronization (x1 sec)	Time interval of the synchronization requests (seconds).	60	1 to 255
Minimum Error Before Clock Update (x1 ms)	Offset value acceptable between the server and client (milliseconds).	100	1 to 65519
IP Address of the First SNTP Server	IP Address of the primary SNTP server.	192.168.15.10	1.0.0.1 to 223.255.255.254.
IP Address of the Second SNTP Server	IP Address of the secondary SNTP server.	192.168.15.11	1.0.0.1 to 223.255.255.254.

Table 31: SNTP Configurations

Notes:

SNTP Server: It is possible to define a preferential address and another secondary one in order to access a SNTP server and, therefore, to obtain a synchronism of time. If both fields are empty, the SNTP service will remain disabled.

Time zone: The time zone configuration is used to convert the local time into UTC and vice versa. While some sync sources use the local time (IEC 60870-5-104 protocol, *SetDateAndTime* Function), others use the UTC time (SNTP). The UTC time is usually used to stamp events (IEC 60870-5-104 protocol and MasterTool Device LOG), while the local time is used by another CPU's features (*GetDateAndTime* function).

It is allowed to enable more than one sync source on the project, however the device doesn't support the synchronism from more than one sync source during operation. Therefore there is implicitly defined a priority mechanism. The synchronism through SNTP is more priority than through IEC 60870-5-104 protocol. So, when both sources are enabled and SNTP server is present, it is going to be responsible for the CPU's clock sync, and any sync command from IEC 60870-5-104 is going to be denied.

5.1.2.1. IEC 60870-5-104

In case the synchronism is through IEC 60870-5-104 protocol, the user must enable the time sync at the protocol configuration screen to receive the clock synchronization. To set this option on the device, check the parameter *Enable Time Synchronization* available at the Application Layer section of Nexto Series CPUs User Manual code MU214605.

ATTENTION:

If the PLC receives a time sync command from the control center, and this option is disabled, an error answer will be returned to that command. But if this option is enabled then a success message will be returned to the control center, even that the sync command be discarded for there is another synchronism method active with higher priority.

This synchronism method should be used only as an auxiliary synchronism method, once the precision of the clock sync process depends a lot on delays and traffic on the network, as well as the processor load on the CPU, as this mechanism is treated by a low priority task.

5.1.2.2. SNTP

When enabled, the controller will behave as a SNTP client, which is, it will send requests of time synchronization to a SNTP/NTP server which can be in the local net or in the internet. The SNTP client works with a 1 ms resolution. The precision of the time sync through SNTP depends on the protocol configurations (*minimum error to clock update*) and the features of the Ethernet network where it is, if both client and server are in the same network (local) or in different networks (remote). Typically the precision is in tens of milliseconds order.

The controller sends the cyclic synchronization requests according to the time set in the SNTP Synchronization Period field. In the first synchronization attempt, just after the service start up, the request is for the first server set in the first server IP address. In case it does not respond, the requests are directed to the second server set in the second server IP address providing a redundancy of SNTP servers. In case the second server does not respond either, the same process of synchronization attempt is performed again but only after the Period of Synchronization having been passed. In other words, at every synchronization period the controller tries to connect once in each server, it tries the second server in case the first one does not respond. The waiting time for a response from the SNTP server is defined by default in 5 s and it cannot be modified.

If, after a synchronization, the difference between the current time of the controller and the one received by the server is higher than the value set in the *Minimum Error Before Clock Update* parameter, the controller time is updated.

SNTP uses the time in the UTC (Universal Time Coordinated) format, so the *Time zone* parameter needs to be set correctly so the time read by the SNTP will be properly converted to a local time.

The execution process of the SNTP client can be exemplified with the following steps:

1. Attempt of synchronization through the first server. In case the synchronization occurs successfully, the controller waits the time for a new synchronization (*Synchronization Period*) and will synchronize again with this server, using it as a primary server. In case of failure (the server does not respond in less than 5 s) step 2 is performed.
2. Attempt of synchronization through the second server. In case the synchronization occurs successfully, the controller waits the time for a new synchronization (*Synchronization Period*) and will try to synchronize with this server using the primary server. In case of failure (the server does not respond in less than 5 s) the time relative to the Synchronization Period is waited and step 1 is performed again.

As the waiting time for the response of the SNTP server is 5 s, the user must pay attention to lower than 10 s values for the *Synchronization Period*. In case the primary server does not respond, the time for the synchronization will be the minimum of 5 s (waiting for the primary server response and the synchronization attempt with secondary server). In case neither the primary server nor the secondary one responds, the synchronization time will be 10 s minimum (waiting for the two servers response and the new connection with first server attempt).

ATTENTION:

The SNTP Service depends on the user application only for its configuration. Therefore, this service will be performed even when the controller is in STOP or BREAKPOINT modes since there is an application in the controller with the SNTP client enabled and properly set.

5.1.2.3. Daylight Saving Time (DST)

The DST configuration must be done indirectly through the function *SetTimeZone*, which changes the time zone applied to the RTC. In the beginning of the DST, it has to be used a function to increase the time zone in one hour. At the end of the DST, it is used to decrease it in one hour.

For further information, see the section [RTC Clock](#) of this manual.

5.1.3. Internal Points

A communication point is storage on the CPU memory under form of two distinct variables. One represents the point's value (type BOOL, BYTE, WORD, etc. . .), while another, represents its quality (type QUALITY). Internal Points are those which the value and the quality are calculated internally by the user application, that is, they don't have an external origin like occur with points linked to IEDs (Communication drivers of type Master/Client).

This *Internal Points* configuration tab's function is to relate the variable which represents a point's value with the one which represents its quality. It must be used to relate value and quality variables internally created on the PLC program (as in a GVL), which ones typically will be afterlly mapped to a communication driver, of type *Server*, for communication with the control center.

ATTENTION:

If a value variable doesn't own a related quality variable, it will be reported as default a constant good quality (no significant indication) when the value variable is reported to a client or control center.

In this way, this tab purpose isn't to create or declare internal points. To do that, just declare value and/or quality variables in a GVL and map it on the communication driver.

The internal points configuration, shown in the figure below, follow the parameters described in the table below. It's possible to configure up to 5120 entries on *Internal Points* table.

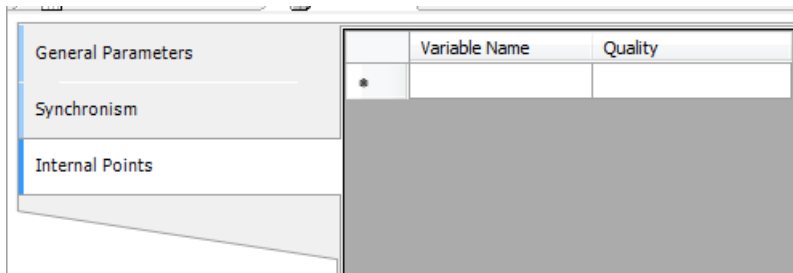


Figure 37: Internal Points Configuration Screen

Configuration	Description	Default	Options
Variable Name	Symbol variable which storage the internal point value.	-	Accept variables of type BOOL, WORD, DWORD, LWORD, INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL or DBP. The variable can be simple, array or array's element and can be part of a struct.
Quality	Symbol variable which storage the internal point quality.	-	QUALITY type variables (LibRtuStandard), which can be simple, array or array's element and can be part of a struct.

Table 32: Internal Points Configuration

The figure below show an example of two internal points configuration.

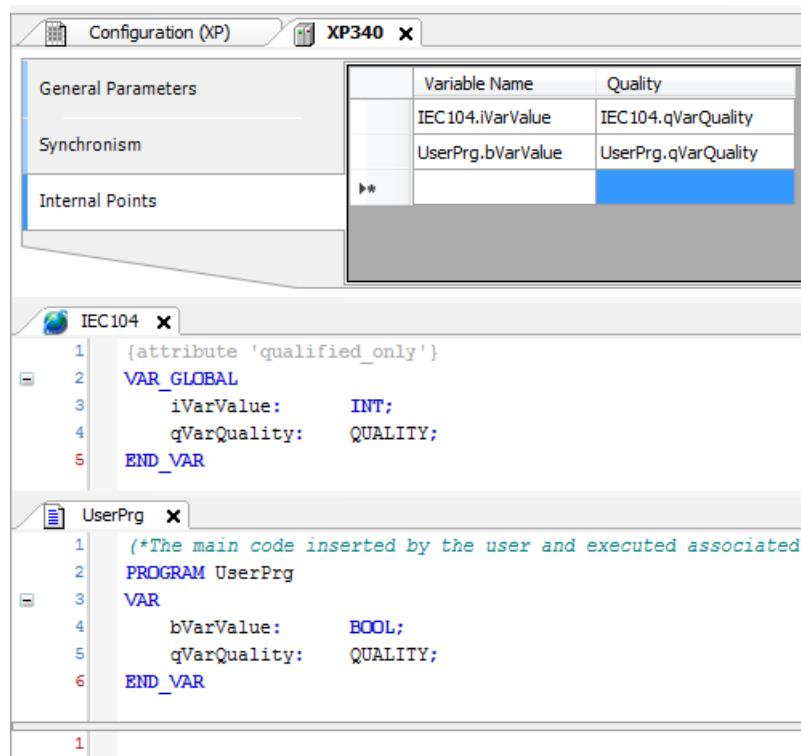


Figure 38: Internal Points Configuration Example

5.1.3.1. Quality Conversions

The internal point's quality is a trust level information about the value stored on that point. The quality may inform, for example, that the value stored is out of range, or yet that it is valid, but low trusted.

The Standards like IEC 104 have their own formats to representation of point's quality information. The Nexto Series, by its turn, have its own quality format (but quite similar to IEC 61850) called *Internal Quality*. This format is defined by type QUALITY (library *LibRtuStandard*) and it is used internally to quality storage, allowing to be done conversion between protocols without information loss.

The following tables define the protocols own formats conversion to internal format. Case it is necessary to consult the conversion between protocols, it is needed to analyze in two steps, looking each of the tables to internal format and after correlating them.

5.1.3.1.1. Internal Quality

This is the QUALITY structure. The table below shows detailed each of its components.

Bit	Name	Type	Description
0	FLAG_RESTART	BOOL	The RESTART flag indicates that the data haven't been updated by the field since the device's reset.
1	FLAG_COMM_FAIL	BOOL	Indicates there is a communication failure on the way between the data origin device and the reports device.
2	FLAG_REMOTE_SUBSTITUTED	BOOL	If TRUE the data values are overwritten in the remote communication devices.
3	FLAG_LOCAL_SUBSTITUTED	BOOL	If TRUE the data value is overwritten by the device which generated this flag. This behavior might occur due to a working in diagnostic or temporary due to human intervention.

Bit	Name	Type	Description
4	FLAG_FILTER	BOOL	Flag used to signalize and prevent the event communication channel overload. As oscillations (rapid changes) on the digital inputs.
5	FLAG_OVERFLOW	BOOL	This flag should indicates a quality problem, that the value, of the attribute to which the quality has been associated, is beyond representation.
6	FLAG_REFERENCE_ERROR	BOOL	This flag should identify that the value cannot be correct due to out of calibration reference.
7	FLAG_INCONSISTENT	BOOL	This flag should identify that an evaluation function has found an inconsistency.
8	FLAG_OUT_OF_RANGE	BOOL	This flag should indicates a quality problem that the attribute to which the quality has been associated is beyond the predefined values capacity.
9	FLAG_INACCURATE	BOOL	This flag should indicates that the value doesn't attend the declared precision of the source.
10	FLAG_OLD_DATA	BOOL	A value seems to be outdated. In case an update doesn't occur during a specific time period.
11	FLAG_FAILURE	BOOL	This flag should indicates that a watch function detected an internal or external failure.
12	FLAG_OPERATOR_BLOCKED	BOOL	Update blocked by operator.
13	FLAG_TEST	BOOL	This must be an additional identifier which can be used to classify a value being that a test value which won't be used to operational ends.
14-15	RESERVED	-	Reserved
16-17	VALIDITY	QUALITY_VALIDITY	0 – Good (Trustfull value, means that there is no abnormal conditions) 1 – Invalid (Value doesn't match the IED's value) 2 – Reserved (Reserved) 3 – Questionable (Present value might be not the same from the IED)

Table 33: QUALITY Structure

5. CONFIGURATION

5.1.3.1.2. IEC 60870-5-104 Conversion

The tables below show respectively the digital, analog and counters internal point's conversion to IEC 60870-5-104 of Nexto Series available to MT8500.

Internal -> IEC 60870-5-104 Digital		
Internal Quality		
Flags	VALIDITY	IEC 60870-5-104 Quality
FLAG_RESTART	ANY	NOT TOPICAL
FLAG_COMM_FAIL	ANY	NOT TOPICAL
FLAG_REMOTE_SUBSTITUTED	ANY	SUBSTITUTED
FLAG_LOCAL_SUBSTITUTED	ANY	SUBSTITUTED
FLAG_FILTER	ANY	-
FLAG_OVERFLOW	ANY	-
FLAG_REFERENCE_ERROR	ANY	-
FLAG_INCONSISTENT	ANY	-
FLAG_OUT_OF_RANGE	ANY	-
FLAG_INACCURATE	ANY	-
FLAG_OLD_DATA	ANY	NOT TOPICAL
FLAG_FAILURE	ANY	INVALID
FLAG_OPERATOR_BLOCKED	ANY	BLOCKED
FLAG_TEST	ANY	-
ANY	VALIDITY_INVALID	INVALID

Table 34: Digital Points Conversion Internal to IEC 60870-5-104

Internal -> IEC 60870-5-104 Analog		
Internal Quality		
Flags	VALIDITY	IEC 60870-5-104 Quality
FLAG_RESTART	ANY	NOT TOPICAL
FLAG_COMM_FAIL	ANY	NOT TOPICAL
FLAG_REMOTE_SUBSTITUTED	ANY	SUBSTITUTED
FLAG_LOCAL_SUBSTITUTED	ANY	SUBSTITUTED
FLAG_FILTER	ANY	-
FLAG_OVERFLOW	ANY	OVERFLOW
FLAG_REFERENCE_ERROR	ANY	INVALID
FLAG_INCONSISTENT	ANY	INVALID
FLAG_OUT_OF_RANGE	ANY	OVERFLOW
FLAG_INACCURATE	ANY	INVALID
FLAG_OLD_DATA	ANY	NOT TOPICAL
FLAG_FAILURE	ANY	INVALID
FLAG_OPERATOR_BLOCKED	ANY	BLOCKED
FLAG_TEST	ANY	-
ANY	VALIDITY_INVALID	INVALID

Table 35: Analog Points Conversion Internal to IEC 60870-5-104

Internal -> IEC 60870-5-104 Counters		
Internal Quality		
Flags	VALIDITY	IEC 60870-5-104 Quality
FLAG_RESTART	ANY	-
FLAG_COMM_FAIL	ANY	-
FLAG_REMOTE_SUBSTITUTED	ANY	-
FLAG_LOCAL_SUBSTITUTED	ANY	-
FLAG_FILTER	ANY	-
FLAG_OVERFLOW	ANY	OVERFLOW
FLAG_REFERENCE_ERROR	ANY	-
FLAG_INCONSISTENT	ANY	-
FLAG_OUT_OF_RANGE	ANY	-
FLAG_INACCURATE	ANY	-
FLAG_OLD_DATA	ANY	-
FLAG_FAILURE	ANY	INVALID
FLAG_OPERATOR_BLOCKED	ANY	-
FLAG_TEST	ANY	-
ANY	VALIDITY_INVALID	INVALID

Table 36: Counters Conversion Internal to IEC 60870-5-104

5.1.3.1.3. MODBUS Internal Quality

As the MODBUS standard don't specify quality types to each point, but for help on use of each point's communication diagnostic, MasterTool allows the quality variables mapping, through an internal own structure, to each MODBUS point. The table below describes the quality types that each MODBUS point can assume.

Resulting Quality	Resulting VALIDITY	Description
FLAG_RESTART	VALIDITY_INVALID	Initial value. The point was never updated.
-	VALIDITY_GOOD	Communication OK. The point is updated.
FLAG_COMM_FAIL AND FLAG_RESTART	VALIDITY_INVALID	Communication error. The point never was updated.
FLAG_COMM_FAIL AND FLAG_OLD_DATA	VALIDITY_QUESTIONABLE	An error has occurred but the point was updated and now has an old value.
FLAG_FAILURE AND FLAG_RESTART	VALIDITY_INVALID	It has received an exception response and the point kept its initial value.
FLAG_FAILURE AND FLAG_OLD_DATA	VALIDITY_QUESTIONABLE	It has received an exception response, but the point has a valid old value.
FLAG_RESTART AND FLAG_OLD_DATA	VALIDITY_QUESTIONABLE	Device stopped. The point has an old value.

Table 37: MODBUS Quality

5.2. Serial Interface

5.2.1. COM 1

The COM 1 interface is a RS-485 standard serial interface. It allows the point to point or network communication in the open protocols MODBUS RTU slave or MODBUS RTU master.

When using the MODBUS master / slave protocol, some of these parameters (such as Serial Mode, Data Bits, RX Threshold and Events Serial) are automatically adjusted by MasterTool tool for the correct operation of this protocol.

The parameters which must be configured for the proper functioning of the application are described below:

Configuration	Description	Default	Options
Serial Type	Serial channel configuration	RS-485	RS-485
Baud Rate	Serial communication port speed configuration	115200	2400, 4800, 9600, 19200, 38400, 57600, 115200 bps
Parity	Serial port parity configuration	None	Odd Even No parity
Data Bits	Sets the data bits quantity in each serial communication character	8	6, 7 and 8
Stop Bits	Sets the serial port stop bits	1	1 and 2
Serial Mode	Sets the serial port operation mode	Normal Mode	- Extended Mode: Extended operation mode which delivers information regarding the received data frame (see note on COM 1 section) - Normal Mode: Serial communication normal operation mode

Table 38: RS-485 Standard Serial Configurations

5.2.2. Advanced Configurations

The advanced configurations section allows to configure additional parameters of the serial port as described below:

Configuration	Description	Default	Options
UART RX Threshold	Bytes quantity which must be received for a new UART interruption to be generated. Low values make the TIMESTAMP more precise when the EXTENDED MODE is used and minimizes the overrun errors. However, values too low may cause several interruptions delaying the CPU.	8	1, 4, 8 and 14
RS-485 Termination	Enables the internal standard RS-485 termination. Must be enabled only if the controller is physically positioned at one of the extremities of the RS-485 network.	Enabled	Disabled or Enabled

Table 39: RS-485 Standard Serial Advanced Configurations

5.3. Ethernet Interface

5.3.1. NET 1

The NET 1 interface is composed by a RJ45 communication connector 10/100Base-TX standard. It allows the point to point or network communication with several protocols, for example: MODBUS, OPC UA, etc...

The parameters which must be configured for the proper functioning of the application are described below:

Configuration	Description	Default	Options
Obtain an IP address automatically	Enables the DHCP Client functionality on the device for automatic IP assignment.	Unmarked	Marked or Unmarked
IP Address	IP address of the controller in the Ethernet bus.	192.168.15.1	1.0.0.1 to 223.255.255.254
Sub network Mask	Subnet mask of the controller in the Ethernet bus.	255.255.255.0	128.0.0.0 to 255.255.255.252
Gateway Address	Controller Gateway address in the Ethernet bus.	192.168.15.253	1.0.0.1 to 223.255.255.254

Table 40: NET 1 Configuration

5.3.2. Reserved TCP Ports

The following TCP ports of the Ethernet interfaces, both local and remote, are used by CPUs services, so they are reserved and cannot be used by the user: 80, 161, 8080, 1217, 1740, 1741, 1742,1743 and 11740.

5.4. Controller Area Network Interface

5.4.1. CAN

The CAN interface allows point to point or network communication with other devices that have this interface using CANopen application protocol.

The parameters of CAN interface which must be configured for the proper functioning of the application are described below:

Configuration	Description	Default	Options
Network	CAN interface ID number	0	0 (fixed)
Baudrate	CAN Bus baudrate (kbit/s). The other devices must to use the same baudrate.	250	10, 20, 50, 100, 125, 250, 500, 800, 1000

Table 41: CAN Configuration

The parameters related to CANopen protocol are described on [Communication Protocols](#) section.

5.5. Integrated I/O

Nexto Xpress controllers have integrated I/O points, which allows it to interface with external devices like sensors, actuators, step motors, encoders, etc...

There are two objects on project treeview related to Integrated I/O, as shown on the figure below:

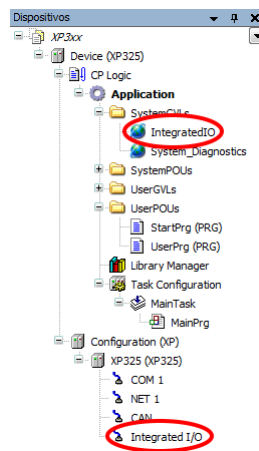


Figure 39: Integrated I/O objects on project treeview

One of these objects is the GVL *IntegratedIO*, which is created automatically by MasterTool IEC XE and contains a list of global symbolic variables that are directly mapped to the onboard inputs and outputs.

The other object is the connector *Integrated I/O*, which contains the configuration for each type of I/O point. These configurations will be detailed on next sections.

5.5.1. Digital Inputs

The parameters related to the *Digital Inputs* are located on the screen below (example from XP325), for both standard and fast inputs (when configured as normal digital inputs):

Figure 40: Digital Inputs Parameters

The table below shows the possible configuration values:

Configuration	Description	Default	Options
Filter time	Minimum time that an input must remain in a given state to confirm the state change	20 ms	2 to 255 ms
Filter	Enable/Disable filter for each input	Disabled	Enabled or Disabled

Table 42: Digital Inputs Parameters

Note:

Input Filter Time Constant: The filter sampling is performed on MainTask (or Refresh function), then it's recommended to use multiple values of the task interval.

5.5.2. Fast Inputs

The fast inputs are special input signals that can be used for special high-speed functions. These special physical inputs can be assigned to two types of logical elements: high-speed counters and external interruption. Each of these logical elements consumes a certain amount of fast inputs signals. For the high-speed counters unit, it depends on the selected mode (Up/Down, Quadrature, etc. . .), while each external interruption uses one fast input signal. The number of physical fast inputs, as well as the maximum number of high-speed counter and external interruption logical elements assignable for these inputs is described on [Technical Description](#) section.

The following table shows how each high-speed counter unit is assigned to the fast inputs signals:

High-Speed Counter	Counter Mode	Fast Inputs			
		I00	I01	I02	I03
Counter 0	Up/Down (A count, B direction) with zero	B	Z	A	-
	Quadrature 2X	A	B	-	-
	Quadrature 2X with zero	A	B	Z	-
	Quadrature 4X	A	B	-	-
	Quadrature 4X with zero	A	B	Z	-

Table 43: High-Speed Counters and Fast Inputs allocation

For each configuration described above, the remaining fast input signals (not used by the high-speed counter units) can be used as external interruption, respecting the maximum number of this kind of logical element specified on [Technical Description](#) section.

The configuration of high-speed counters and interruptions is located on the following screen:

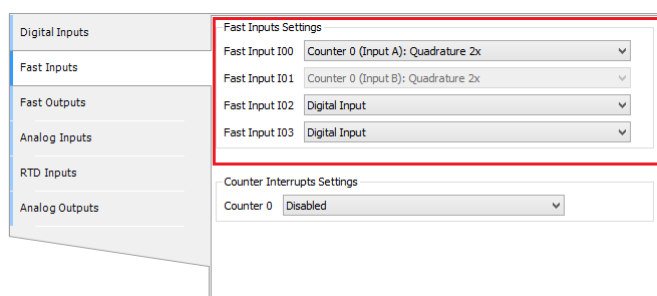


Figure 41: Fast Inputs settings

When selecting the function of a fast input, the following inputs are automatically assigned (locked for edition) according to the mode of the high-speed counter unit.

The table below shows the possible configuration values for each fast input:

Configuration	Description	Default	Options
Fast Input I00	Fast Input I00 configuration	Digital Input	Digital Input Counter 0 (Input B): Up/Down (A count, B direction) with zero Counter 0 (Input A): Quadrature 2X Counter 0 (Input A): Quadrature 2X with zero Counter 0 (Input A): Quadrature 4X Counter 0 (Input A): Quadrature 4X with zero
Fast Input I01	Fast Input I01 configuration	Digital Input	Digital Input Obs: This field will be set automatically when Fast Input I00 is configured as Up/Down or Quadrature Counter.
Fast Input I02	Fast Input I02 configuration	Digital Input	Digital Input Interruption (Rising Edge) Obs: This field will be set automatically when Fast Input I00 is configured as Up/Down or Quadrature Counter with zero.
Fast Input I03	Fast Input I03 configuration	Digital Input	Digital Input Interruption (Rising Edge)

Table 44: Fast Inputs Parameters

Even if a fast input is configured as a counter or interruption, it's digital value can still be read through *Integrate-dIo.DigitalInputs* variable. The next subsections give more details about the *Fast Inputs* configuration and programming.

5.5.2.1. High-Speed Counters

The high-speed counter units have multiple operating modes. The following table describes the details of each of these modes:

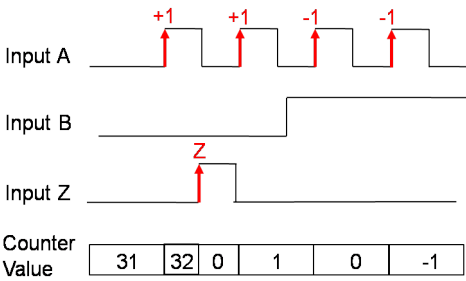
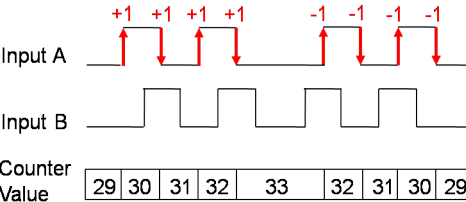
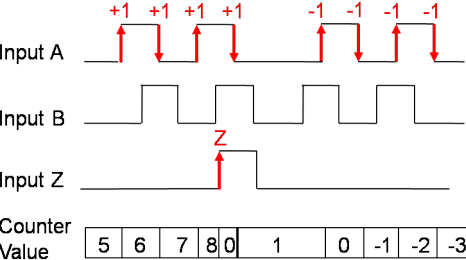
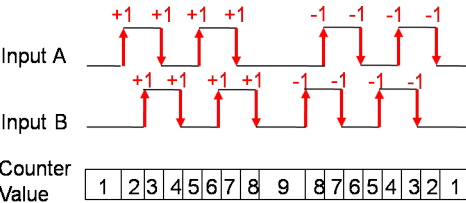
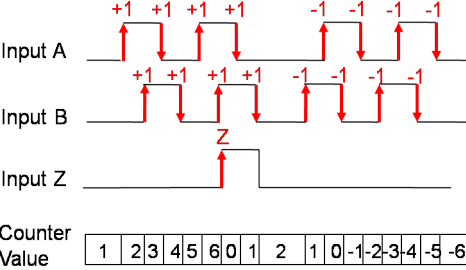
Counter Mode	Counting waveforms
Up/Down (A count, B direction) with zero	
Quadrature 2X	
Quadrature 2X with zero	
Quadrature 4X	
Quadrature 4X with zero	

Table 45: High-speed counter modes

The overall behavior is the same for all counters: when counting UP and the maximum positive value is reached, the next value will be the minimum negative value. The same thing happens for the opposite direction, so when counting DOWN and

the minimum negative value is reached, the next value will be the maximum positive value.

The user program can access the high-speed counters through the *FastInputs* symbolic structure, which is automatically created on *IntegratedIo* GVL. For each high-speed counter unit, there are 3 main areas as shown on the following figure:

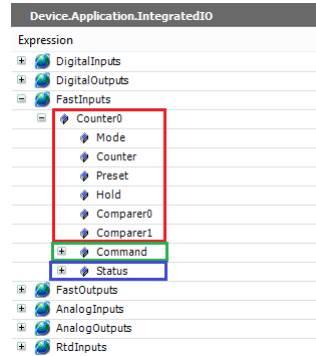


Figure 42: Counter structure

The table below describes the counter variables structure:

Variable	Description	Type	Allowed Values
Mode	Configured counter mode (read only)	ENUM_COUNTER_MODE	DISABLED UP_DOWN_A_COUNT_B_DIR_WITH_ZERO QUADRATURE_2X QUADRATURE_2X_WITH_ZERO QUADRATURE_4X QUADRATURE_4X_WITH_ZERO
Counter	Counter value	DINT	-2147483648 to 2147483647
Preset	Preset value	DINT	-2147483648 to 2147483647
Hold	Hold value	DINT	-2147483648 to 2147483647
Comparer0	Lower value of counter comparison	DINT	-2147483648 to 2147483647
Comparer1	Higher value of counter comparison	DINT	-2147483648 to 2147483647
Command	Counter commands structure	T_COUNTER_COMMAND	-
Status	Counter status structure	T_COUNTER_STATUS	-

Table 46: Counter structure variables

The command and status are structures of bits that allow the user program to control the counter operation. The following table describes the counter command structure.

Variable	Description	Type	Allowed Values
Stop	Stop the counter. The counter remains stopped while this bit is set	BIT	FALSE or TRUE
Reset	Reset the counter. The counter remains zeroed while this bit is set	BIT	FALSE or TRUE
Load	Load the preset value to the counter value. This operation is performed on rising edge of this bit	BIT	FALSE or TRUE

Variable	Description	Type	Allowed Values
Sample	Sample the counter storing its value in hold. This operation is performed on rising edge of this bit	BIT	FALSE or TRUE

Table 47: Counter command structure

The following table describes the counter status structure.

Variable	Description	Type	Allowed Values
Enabled	Counter is enabled	BIT	FALSE or TRUE
Direction	Counter direction (TRUE = Up, FALSE = Down)	BIT	FALSE or TRUE
EQComparer0	Counter value is equal to Comparer0	BIT	FALSE or TRUE
LTComparer0	Counter value is less than Comparer0	BIT	FALSE or TRUE
GTComparer0	Counter value is greater than Comparer0	BIT	FALSE or TRUE
EQComparer1	Counter value is equal to Comparer1	BIT	FALSE or TRUE
LTComparer1	Counter value is less than Comparer1	BIT	FALSE or TRUE
GTComparer1	Counter value is greater than Comparer1	BIT	FALSE or TRUE

Table 48: Counter status structure

Additionally to the *IntegratedIo* global variables, there is a function block from *LibIntegratedIo* library which allows to instantiate high-speed counter in POUs written in graphical languages (e.g Ladder Logic Diagram). This function block is, actually, a wrapper to the structured variables described before. The figure below shows the function block instantiated in a Ladder program.

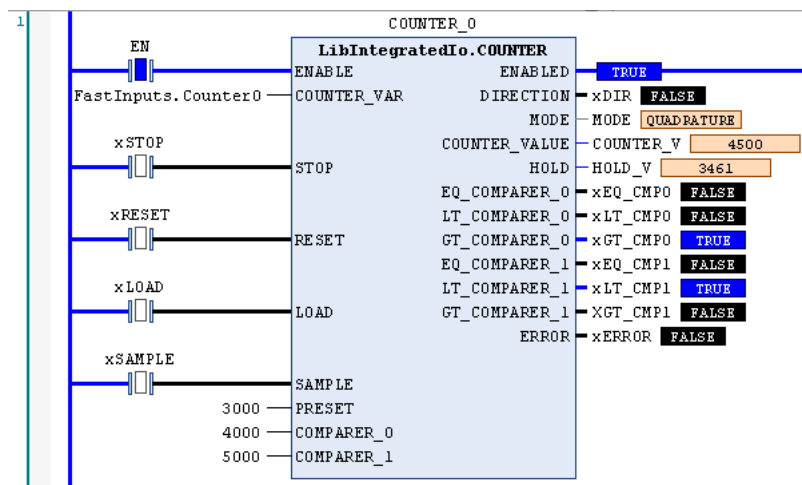


Figure 43: LibIntegratedIo.COUNTER function block

The table below describes the inputs and outputs variables of the function block.

Variable	Description	Type	Allowed Values
ENABLE	Enable the function block execution	BOOL	FALSE or TRUE
COUNTER_VAR	Counter variable	REFERENCE TO T_COUNTER	FastInputs.Counter0
STOP	Stop the counter	BOOL	FALSE or TRUE
RESET	Reset the counter	BOOL	FALSE or TRUE
LOAD	Load the preset value to the counter value	BOOL	FALSE or TRUE
SAMPLE	Sample counter storing its value in hold	BOOL	FALSE or TRUE
PRESET	Preset value	DINT	-2147483648 to 2147483647
COMPARER_0	Lower value of counter comparison	DINT	-2147483648 to 2147483647
COMPARER_1	Higher value of counter comparison	DINT	-2147483648 to 2147483647
ENABLED	Counter is enabled	BOOL	FALSE or TRUE
DIRECTION	Counter direction (TRUE = Up, FALSE = Down)	BOOL	FALSE or TRUE
Mode	Counter mode	ENUM_ COUNTER_MODE	DISABLED UP_DOWN_A_- COUNT_ B_DIR_- WITH_ZERO QUADRATURE_2X QUADRATURE_2X_- WITH_ZERO QUADRATURE_4X QUADRATURE_4X_- WITH_ZERO
COUNTER_VALUE	Counter value	DINT	-2147483648 to 2147483647
HOLD	Hold value	DINT	-2147483648 to 2147483647
EQ_COMPARER_0	Counter value is equal to Comparer0	BOOL	FALSE or TRUE
LT_COMPARER_0	Counter value is less than Comparer0	BOOL	FALSE or TRUE
GT_COMPARER_0	Counter value is greater than Comparer0	BOOL	FALSE or TRUE
EQ_COMPARER_1	Counter value is equal to Comparer1	BOOL	FALSE or TRUE
LT_COMPARER_1	Counter value is less than Comparer1	BOOL	FALSE or TRUE
GT_COMPARER_1	Counter value is greater than Comparer1	BOOL	FALSE or TRUE
ERROR	Error occurred in function block execution. Can be caused by invalid COUNTER_VAR or counter disabled.	BOOL	FALSE or TRUE

Table 49: LibIntegratedIo.COUNTER function block description

5.5.2.1.1. Counter Interrupts

The high-speed counter units have the ability to generate interrupts by comparison, i.e., when the counter reach a certain comparison value, an specific task will run and interrupt the main program execution. Each high-speed counter unit have two comparison values, called *Comparer0* and *Comparer1*, which are present on the corresponding global symbolic data structure or FunctionBlock as described on previous sections. The configuration of counter interrupt for each high-speed counter unit is located on the following screen:

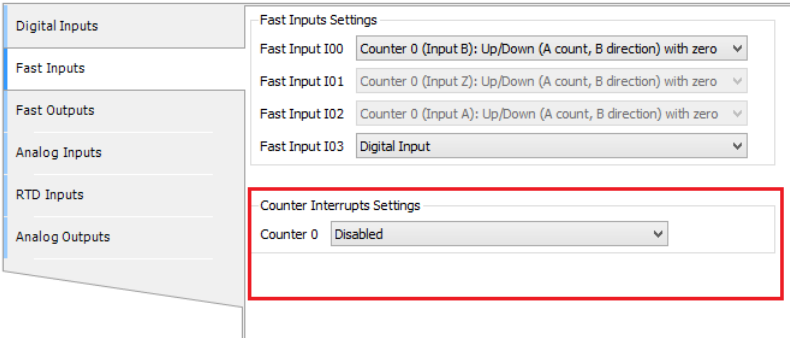


Figure 44: Counter interrupt settings

The table below shows the possible configuration values for the counter interrupt:

Configuration	Description	Default	Options
Counter 0	Counter0 comparator interrupt configuration	Disabled	Disabled Counter0InterruptTask: Counter equal to Comparer0 Obs: This configuration is available when the Counter0 is associated to some Fast Input.

Table 50: Counter interrupt parameters

The counter interrupt will generate an specific event. This event must trigger the execution of external event task, which must call an specific POU. For example, the comparison event generated for Counter 0 is called *COUNTER0_EVT*. So, an external event task called *Counter0InterruptTask* must be configured to be triggered by this event, and must call a POU called *Counter0InterruptPrg* which will contain the user program to be executed.

The figure below shows this configuration scenario in MasterTool IEC XE.

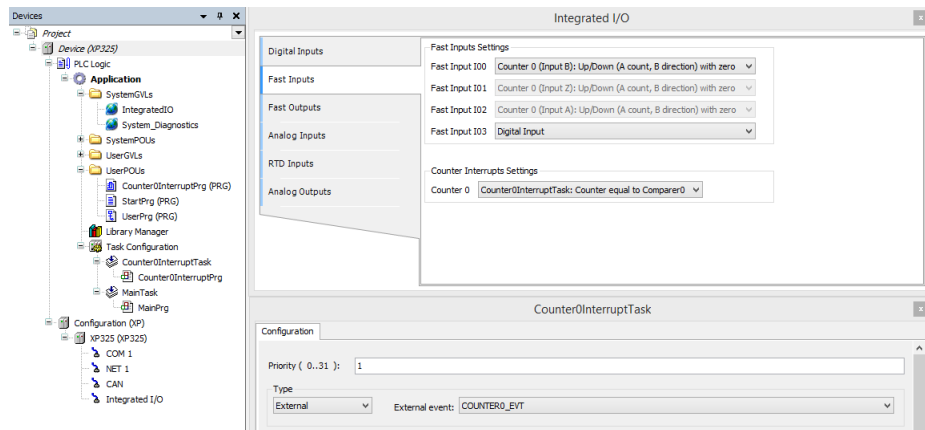


Figure 45: Counter Interrupt Settings

5.5.2.2. External Interruption

The fast inputs can be set as *Interruption (Rising Edge)* mode, which means that when a rising edge (0V to 24V transition) is performed on the input, an specific task will run and interrupt the main program execution.

Each external interruption will generate an specific event. This event must trigger the execution of external event task, which must call an specific POU. For example, the external interruption event generated for fast input I02 is called *FIN2_EVT*. So, an external event task called *FastInputI02InterruptTask* must be configured to be triggered by this event, and must call a POU called *FastInputI02InterruptPrg* which will contain the user program to be executed.

The figure below shows this configuration scenario in MasterTool IEC XE.

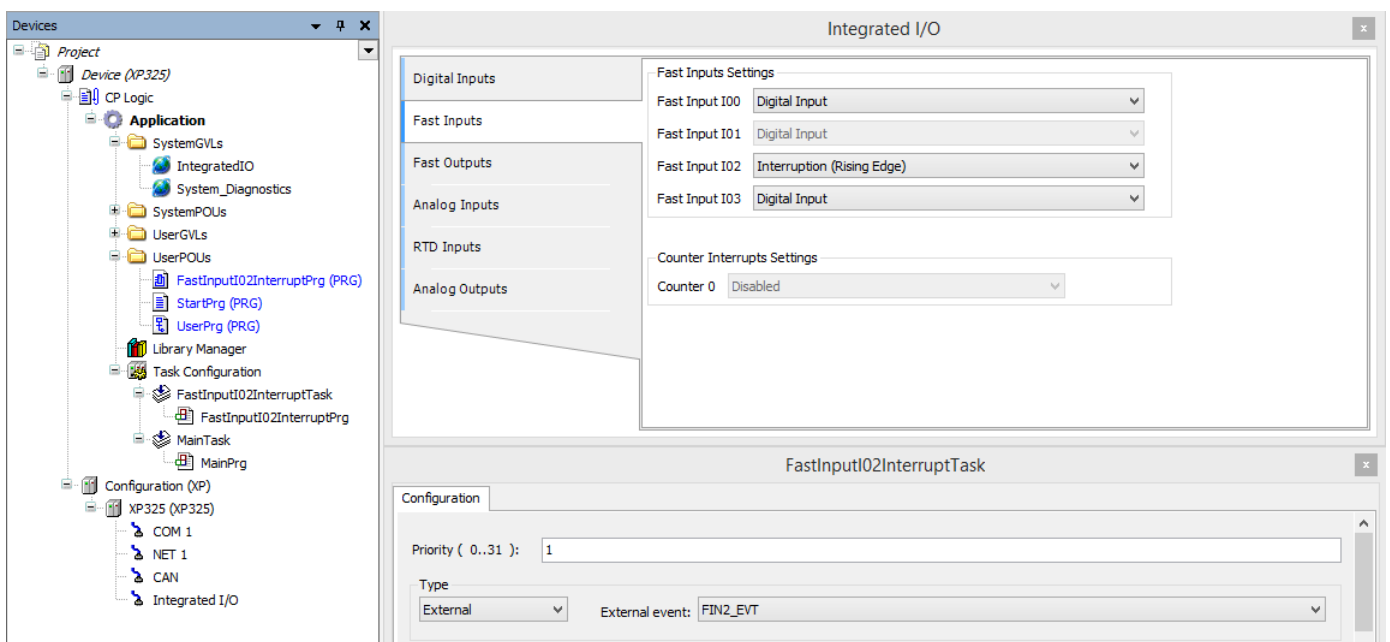


Figure 46: Fast Inputs Interruption Settings

ATTENTION:

The external interruption input have a 10ms time window filter to protect the controller against spurious transitions on the input signal. This window starts right after the occurrence of the interruption and, during this time, any other external interruption event will be discarded.

ATTENTION:

The external interruption does not supports reentrancy. If another interruption occurs (after the filter time) and its program execution is still not finished, this interruption will be discarded.

5.5.3. Fast Outputs

The fast outputs are special output signals that can be used for pulse generator outputs. These special physical outputs can be assigned to two types of logical elements: VFO/PWM (variable frequency/pulse width) and PTO (pulse train output). Each of these logical elements consumes one fast output signal each one. The number of physical fast outputs, as well as the maximum number of the VFO/PWM and PTO logical elements assignable to these outputs is described on [Technical Description](#) section.

The configuration of fast outputs is located on the following screen:

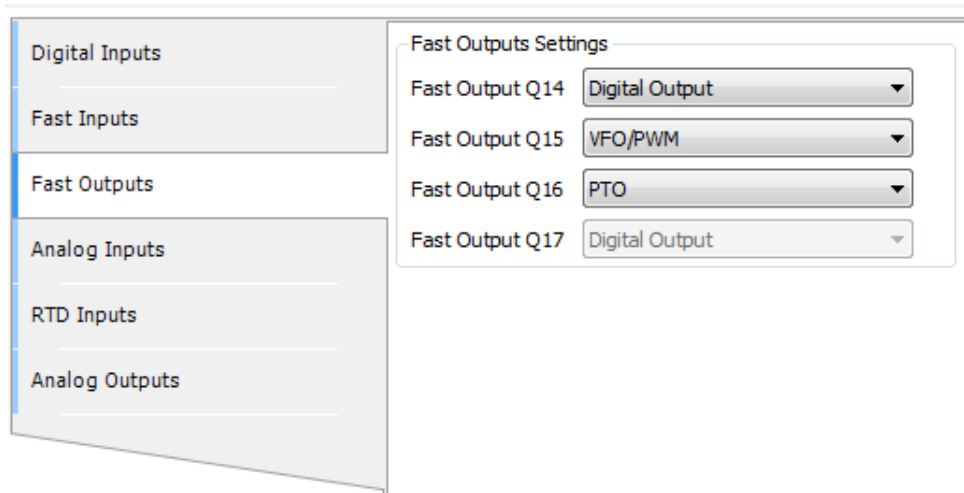


Figure 47: Fast Outputs Parameters

The table below shows the possible configuration values:

Configuration	Description	Default	Options
Fast Output Q14	Fast Output Q14 configuration.	Digital Output	Digital Output VFO/PWM PTO
Fast Output Q15	Fast Output Q15 configuration.	Digital Output	Digital Output VFO/PWM
Fast Output Q16	Fast Output Q16 configuration.	Digital Output	Digital Output VFO/PWM PTO
Fast Output Q17	Fast Output Q17 configuration.	Digital Output	Digital Output VFO/PWM

Table 51: Fast Outputs Parameters

The PTO function can be assigned only for Q14 and Q16. When the output is configured on this mode, the adjacent output (Q15 or Q17) will be forced to normal digital output mode.

As shown on the previous table, the fast outputs can be configured as normal digital output. In this case, its digital value can be set using the standard global variable *IntegratedIo.DigitalOutputs*.

When configured as VFO/PWM or PTO, the user program can control the fast outputs through the *FastOutputs* symbolic structure, which is automatically created on IntegratedIo GVL as shown on the following figure:

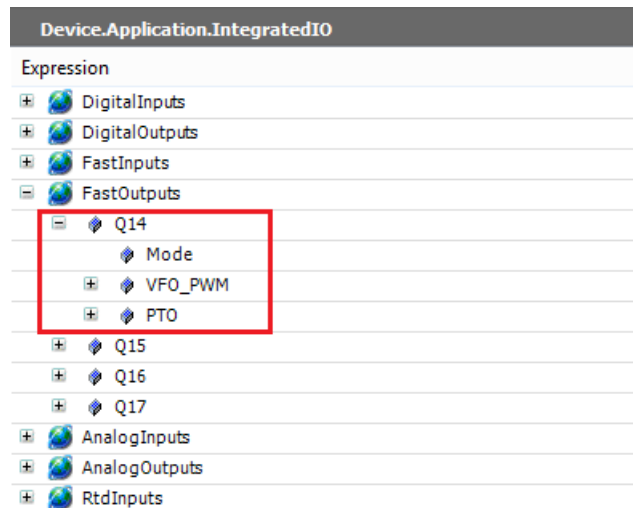


Figure 48: Fast Output structure

The table below describes the fast output variables structure:

Variable	Description	Type	Allowed Values
Mode	Fast output configured mode (read only)	ENUM_FAST_PUT_MODE	DIGITAL_OUTPUT PWM PTO
VFO_PWM	VFO/PWM structure. It contains a structure to control the fast output when it's configured as VFO/PWM.	T_VFO_PWM	-
PTO	PTO structure. It contains a structure to control the fast output when it's configured as PTO.	T_PTO	-

Table 52: Fast Output structure variables

The next subsections give more details about how to use these pulse generator functions, describing these structures for each mode.

5.5.3.1. VFO/PWM

The VFO/PWM (Variable Frequency Output / Pulse Width Modulator) is a pulse generator output mode where the frequency and duty cycle can be controlled by the user program. It's applicable, for example, to control the power transferred to an electric load or to control the angle of a servo motor. The principle of operation of VFO/PWM output is very simple, see the pulsed waveform that is shown in the figure below:

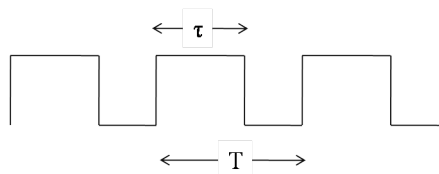


Figure 49: VFO/PWM waveform

The figure shows a pulsed waveform, where T is the period of the pulses and τ is the pulse width. Those are the pulse parameters which can be changed on VFO/PWM mode. The frequency is defined as the inverse of period, then:

$$f = \frac{1}{T}$$

The duty cycle is the reason between the pulse width and the period, then:

$$D = \frac{\tau}{T} 100\%$$

To control the VFO/PWM output, the user program must access the VFO_PWM variable of the fast output structure. The structure of VFO_PWM is shown on the table below:

Variable	Description	Type	Allowed Values
Frequency	Frequency in Hertz	UDINT	1 to 200000
DutyCycle	Duty Cycle in percent	USINT	0 to 100
Command	VFO/PWM commands structure	T_VFO_PWM_COMMAND	-
Status	VFO/PWM status structure	T_VFO_PWM_STATUS	-

Table 53: VFO_PWM variable structure

The table below shows the VFO_PWM commands structure.

Variable	Description	Type	Allowed Values
Enable	Enable VFO/PWM output	BIT	FALSE or TRUE

Table 54: VFO/PWM Command structure

The table below shows the VFO_PWM status structure.

Variable	Description	Type	Allowed Values
InvalidFrequency	Frequency value is invalid (out of range)	BIT	FALSE or TRUE
InvalidDutyCycle	Duty Cycle value is invalid (out of range)	BIT	FALSE or TRUE

Table 55: VFO/PWM Status structure

Once the Enable command is TRUE, the input parameters will be continuously checked and the status variables will be updated accordingly.

Additionally to the *IntegratedIo* global variables, there is a function block from *LibIntegratedIo* library which allows to instantiate VFO/PWM in POUs written in graphical languages (e.g Ladder Logic Diagram). This function block is, actually, a wrapper to the structured variables described before. The figure below shows the function block instantiated in a Ladder program.

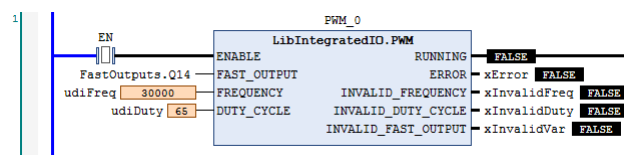


Figure 50: LibIntegratedIo.PWM function block

The table below describes the inputs and outputs variables of the function block.

Variable	Description	Type	Allowed Values
ENABLE	Enable the function block execution.	BOOL	FALSE or TRUE
FAST_OUTPUT	Fast Output Variable.	REFERENCE TO T_FAST _OUTPUT	FastOutputs.Q14 FastOutputs.Q15 FastOutputs.Q16 FastOutputs.Q17
FREQUENCY	Frequency in Hertz.	UDINT	1 to 200000
DUTY_CYCLE	Duty Cycle in percent.	USINT	0 to 100
RUNNING	VFO/PWM is being performed.	BOOL	FALSE or TRUE
ERROR	Error occurred in function block execution. The follow variables provide detailed information.	BOOL	FALSE or TRUE
INVALID_FREQUENCY	Frequency value is invalid (out of range).	BOOL	FALSE or TRUE
INVALID_DUTY_CYCLE	Duty Cycle value is invalid (out of range).	BOOL	FALSE or TRUE
INVALID_FAST_OUTPUT	FAST_OUTPUT was not assigned to the block or isn't configured as VFO/PWM.	BOOL	FALSE or TRUE

Table 56: LibIntegratedIo.PWM function block description

5.5.3.2. PTO

The PTO (Pulse Train Output) is a pulse generator mode. It's used, for example, to control step motors responsible for positioning of mechanisms with considerable inertia. For these cases, the rotation speed must increase slowly (acceleration) when the movement is starting and decrease slowly (deceleration) when the movement is stopping. These acceleration and deceleration are made on pulse train by increasing and decreasing the frequency of the pulses, maintaining the 50% of duty cycle.

There are a set of parameter that must be defined for a pulse train: Start frequency, operation frequency, stop frequency, acceleration profile, total number of pulses, number of pulses in acceleration step, number of pulses in deceleration step. The figure below shows, on Cartesian plane, the relation between the frequency of the pulses and time. The pulse train shown is called trapezoidal profile, because the acceleration and deceleration ramps produce a trapezium shape.

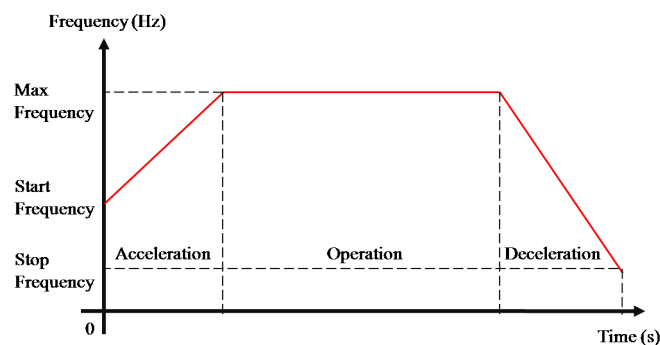


Figure 51: PTO with trapezoidal profile

For some applications it is more recommended to use the “S” profile, which acceleration and deceleration curves are similar

to “S” shape. The figure below shows this profile.

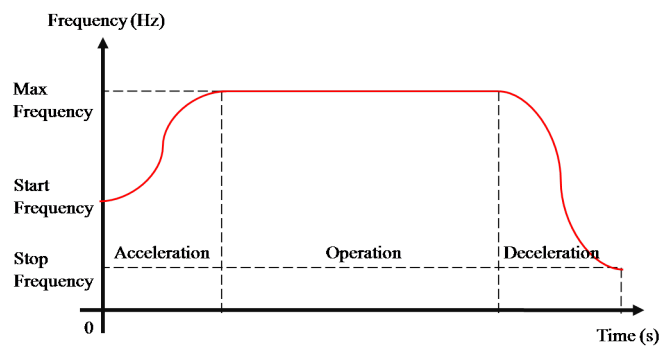


Figure 52: PTO with “S” profile

Besides the PTO parameters, there are status information and commands that the user program can use to control the output. Some important status information are the pulse counter (proportional to a position), the pulse train step (acceleration, operation, deceleration) and, even, if the output is working fine. The commands required to control PTO are to start the pulse train, to stop the pulse train and to stop the pulse train softly (soft stop). The soft stop command is very important, once can be used for emergency situations where the system can’t stop abruptly. The figures below shows how the soft stop command change the pulse train when it is performed. The dashed blue lines represents the PTO if the soft stop command is performed on acceleration and operation steps. The soft stop command on deceleration step has no effect, once the system is already stopping.

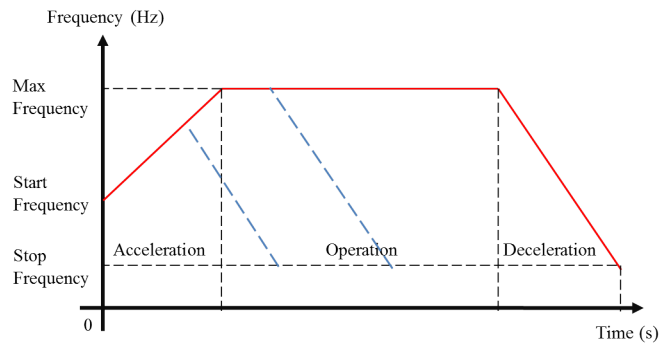


Figure 53: PTO Softstop on trapezoidal profile

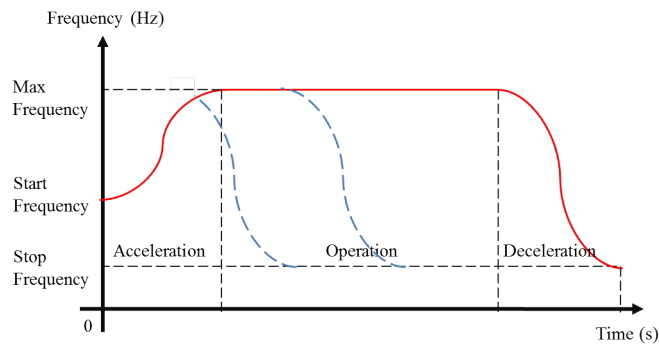


Figure 54: PTO Softstop on "S" profile

To control the PTO, the user program must access the PTO variable of the fast output structure. The structure of PTO is shown on the table below:

Variable	Description	Type	Allowed Values
StartFrequency	Start frequency in Hertz	UDINT	0 to 200000
StopFrequency	Stop frequency in Hertz	UDINT	0 to 200000
MaxFrequency	Maximum frequency in Hertz	UDINT	1 to 200000
AccelerationProfile	Acceleration profile (FALSE = Trapezoidal profile, TRUE = S profile)	BOOL	FALSE or TRUE
AccelerationPulses	Pulses in acceleration	UDINT	0 to (TotalPulses-DecelerationPulses-1)
DecelerationPulses	Pulses in deceleration	UDINT	0 to (TotalPulses-AccelerationPulses-1)
TotalPulses	Total number of pulses	UDINT	1 to 4294967295
PulsesCounter	Number of pulses generated for the current pulse train	UDINT	0 to 4294967295
Command	PTO commands structure	T_PTO_COMMAND	-
Status	PTO status structure	T_PTO_STATUS	-

Table 57: PTO variable structure

The table below shows the PTO commands structure.

Variable	Description	Type	Allowed Values
Start	Start the pulse train when this bit is set (rising edge)	BIT	FALSE or TRUE
Stop	Stop the pulse train when this bit is set (rising edge)	BIT	FALSE or TRUE
Softstop	Stop softly the pulse train when this bit is set (rising edge)	BIT	FALSE or TRUE

Table 58: PTO Command structure

5. CONFIGURATION

The table below shows the PTO status structure.

Variable	Description	Type	Allowed Values
Running	Pulse train is being performed	BIT	FALSE or TRUE
Acceleration	Acceleration step (from StartFrequency to MaxFrequency)	BIT	FALSE or TRUE
Deceleration	Deceleration step (from MaxFrequency to StopFrequency)	BIT	FALSE or TRUE
Operation	Operation Step (MaxFrequency)	BIT	FALSE or TRUE
Done	Pulse train has already been performed	BIT	FALSE or TRUE
InvalidFrequency	Frequency (start, stop or maximum) is invalid	BIT	FALSE or TRUE
InvalidPulses	Number of pulses (TotalPulses, Acceleration or Deceleration) is invalid	BIT	FALSE or TRUE

Table 59: PTO Status structure

Once the Start command is TRUE, the input parameters will be continuously checked and the status variables will be updated accordingly.

Additionally to the IntegratedIo global variables, there is a function block from *LibIntegratedIo* library which allows to instantiate PTO in POU's written in graphical languages (e.g Ladder Logic Diagram). This function block is, actually, a wrapper to the structured variables described before. The figure below shows the function block instantiated in a Ladder program.

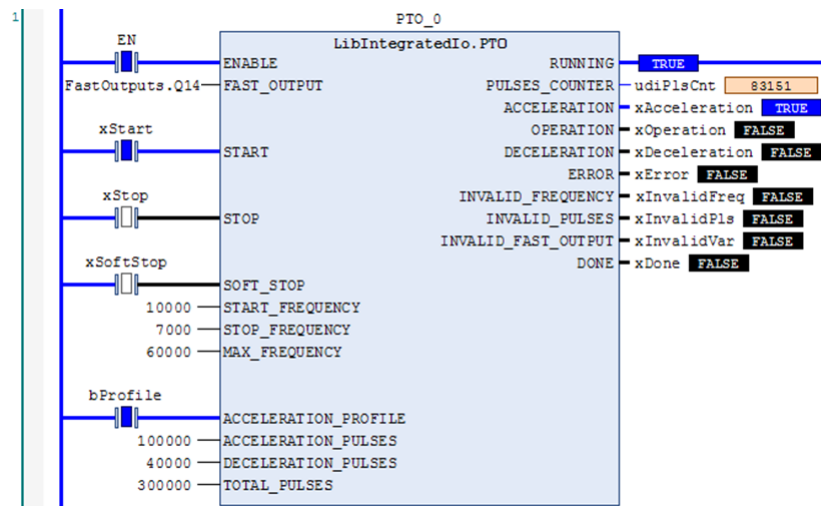


Figure 55: LibIntegratedIo.PTO function block

The table below describes the inputs and outputs variables of the function block.

Variable	Description	Type	Allowed Values
ENABLE	Enable the function block execution	BOOL	FALSE or TRUE
FAST_OUTPUT	Fast Output Variable	REFERENCE TO T_FAST _OUTPUT	FastOutputs.Q14 FastOutputs.Q15 FastOutputs.Q16 FastOutputs.Q17
START	Start the pulse train when this bit is set (rising edge)	BOOL	FALSE or TRUE
STOP	Stop the pulse train when this bit is set (rising edge)	BOOL	FALSE or TRUE
SOFT_STOP	Stop softly the pulse train when this bit is set (rising edge)	BOOL	FALSE or TRUE
START_FREQUENCY	Start frequency in Hertz	UDINT	1 to 200000
STOP_FREQUENCY	Stop frequency in Hertz	UDINT	1 to 200000
MAX_FREQUENCY	Maximum frequency in Hertz	UDINT	1 to 200000
ACCELERATION_PROFILE	Acceleration profile (FALSE = Trapezoidal profile, TRUE = S profile)	BOOL	FALSE or TRUE
ACCELERATION_PULSES	Pulses in acceleration	UDINT	0 to (TotalPulses- DecelerationPulses-1)
DECELERATION_PULSES	Pulses in deceleration	UDINT	0 to (TotalPulses- AccelerationPulses-1)
TOTAL_PULSES	Total number of pulses	UDINT	1 to 4294967295
RUNNING	Pulse train is being performed	BOOL	FALSE or TRUE
PULSES_COUNTER	Number of pulses generated for the current pulse train	UDINT	0 to 4294967295
ACCELERATION	Acceleration step (from StartFrequency to MaxFrequency)	BOOL	FALSE or TRUE
OPERATION	Operation Step (MaxFrequency)	BOOL	FALSE or TRUE
DECELERATION	DecelerationStep (from MaxFrequency to StopFrequency)	BOOL	FALSE or TRUE
ERROR	Error occurred in function block execution. The follow variables detail the error.	BOOL	FALSE or TRUE
INVALID_FREQUENCY	Frequency (start, stop or maximum) is invalid	BOOL	FALSE or TRUE
INVALID_PULSES	Number of pulses (acceleration or deceleration) is invalid	BOOL	FALSE or TRUE
INVALID_FAST_OUTPUT	FAST_OUTPUT was not assigned to the block or isn't configured as PTO.	BOOL	FALSE or TRUE
DONE	Pulse train has already been performed	BOOL	FALSE or TRUE

Table 60: LibIntegratedIo.PTO function block description

5.5.4. Analog Inputs

The parameters related to the Analog Inputs are shown below:

	Input Type	Digital Filter	Open Loop Value
Input AI0	Not Configured	Disabled	Disabled
Input AI1	Not Configured	Disabled	Disabled
Input AI2	Not Configured	Disabled	Disabled
Input AI3	Not Configured	Disabled	Disabled
Input AI4	Not Configured	Disabled	Disabled

Figure 56: Analog Inputs Parameters

The table below shows the possible configuration values:

Configuration	Description	Default	Options
Input Type	Selects the input type	Not configured	Not configured Voltage 0 - 10 Vdc Current 0 - 20 mA Current 4 - 20 mA
Digital Filter	Enable/Disable a 1st order low pass digital filter for each input	Disabled	Disabled 100 ms 1 s 10 s
Open Loop Value	Set value when in open loop condition (Only valid for 4 - 20 mA scale)	Disabled	Disabled 0 30000

Table 61: Analog Inputs Parameters

Notes:

Input Type: Be sure to use the proper pin on the terminal block correspondent to the selected type (voltage or current).

Open Loop Value: : Determines the behavior of the input variable when set to 4 - 20 mA scale and current less than 3 mA.

5.5.5. RTD Inputs

The parameters related to the RTD Inputs are shown below:

Figure 57: RTD Inputs Parameters

The table below shows the possible configuration values:

Configuration	Description	Default	Options
Temperature Unit	Selects the temperature unit	Degree Celsius	Degree Celsius Degree Fahrenheit
Input Type	Selects the input type	Not configured	Not configured 400 Ω 4000 Ω Pt100A Pt100E Pt1000A Pt1000E
Digital Filter	Enable/Disable a 1st order low pass digital filter for each input	Disabled	Disabled 100 ms 1 s 10 s

Table 62: RTD Inputs Parameters

The next table describes additional details about each input type:

Input type	Temperature Coefficient (α)	Measurement Band	Count	Resolution
400 Ω	-	0 to 400 Ω	0 to 4000	0.1 Ω
4000 Ω	-	0 to 4000 Ω	0 to 4000	1 Ω
Pt100E, Pt1000E	0,00385	-200 to 850 $^{\circ}\text{C}$ -328 to 1562 $^{\circ}\text{F}$	-2000 to 8500 -3280 to 15620	0.3 $^{\circ}\text{C}$ 0.6 $^{\circ}\text{F}$
Pt100A, Pt1000A	0,003916	-200 to 630 $^{\circ}\text{C}$ -328 to 1166 $^{\circ}\text{F}$	-2000 to 6300 -3280 to 11660	0.3 $^{\circ}\text{C}$ 0.6 $^{\circ}\text{F}$

Table 63: RTD Input Types

5.5.6. Analog Outputs

The parameters related to the Analog Outputs are shown below:

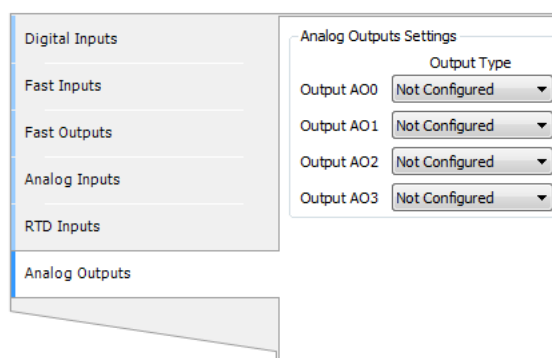


Figure 58: Analog Outputs Parameters

The table below shows the possible configuration values:

Configuration	Description	Default	Options
Output Type	Selects the output type	Not configured	Not configured Voltage 0 - 10 Vdc Current 0 - 20 mA Current 4 - 20 mA

Table 64: Analog Outputs Parameters

5.5.7. I/O Mapping

In the *I/O Mapping* tab, it is possible to configure the name and description for each input and output variable.

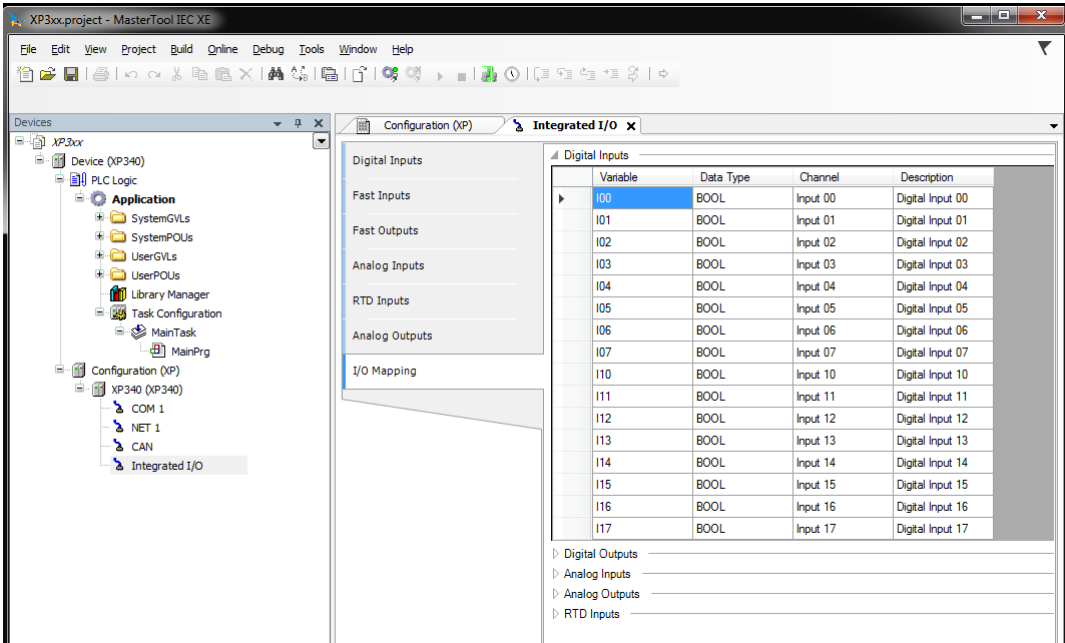


Figure 59: I/O tag mapping

5.6. USB Port

The USB Host port present on Nexto Xpress controllers allows to extend the controller's functionalities by using several types of USB dongles. Due to the wide range of USB devices available on the market (flash drives, Ethernet/Wifi adapters, 3G/4G modem, etc...), the support for each specific device is provided by a firmware update.

The management of USB devices is done through a dedicated section located on the *PLC Management* tab of the controller's system webpage as shown below:

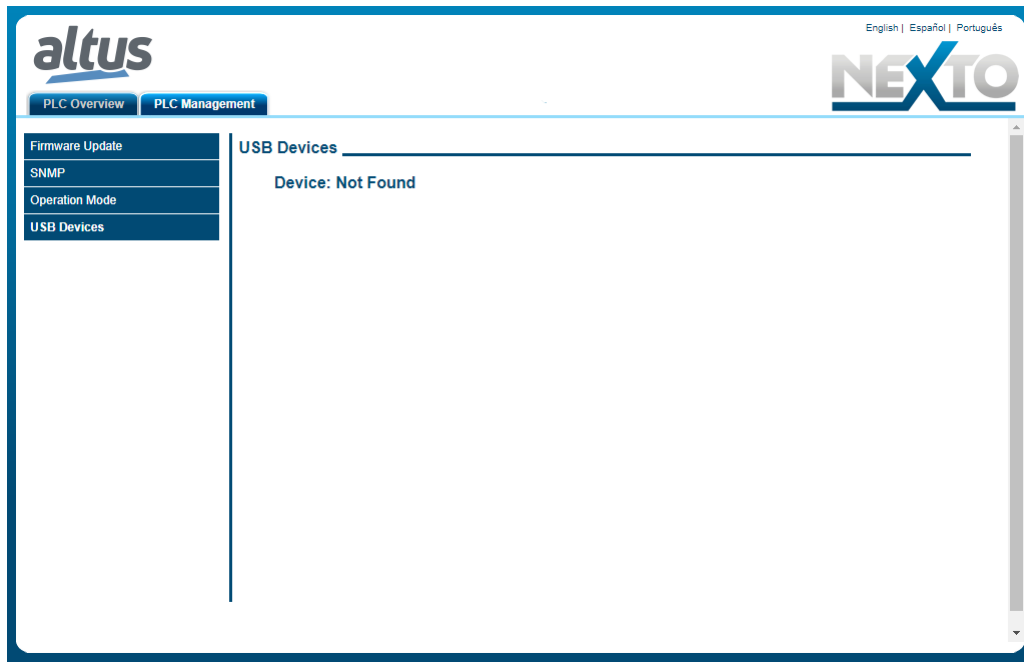


Figure 60: USB Devices Section

The content of this page changes dynamically according to the type of USB device that is connected. In the example above, there is no device connected. This page requires login, similar to the firmware update section. The default user and password are "admin", without quotation marks, for both fields.

The following sections describe all the types of USB devices currently supported. If an unsupported device is connected, the page will inform that device is unknown:

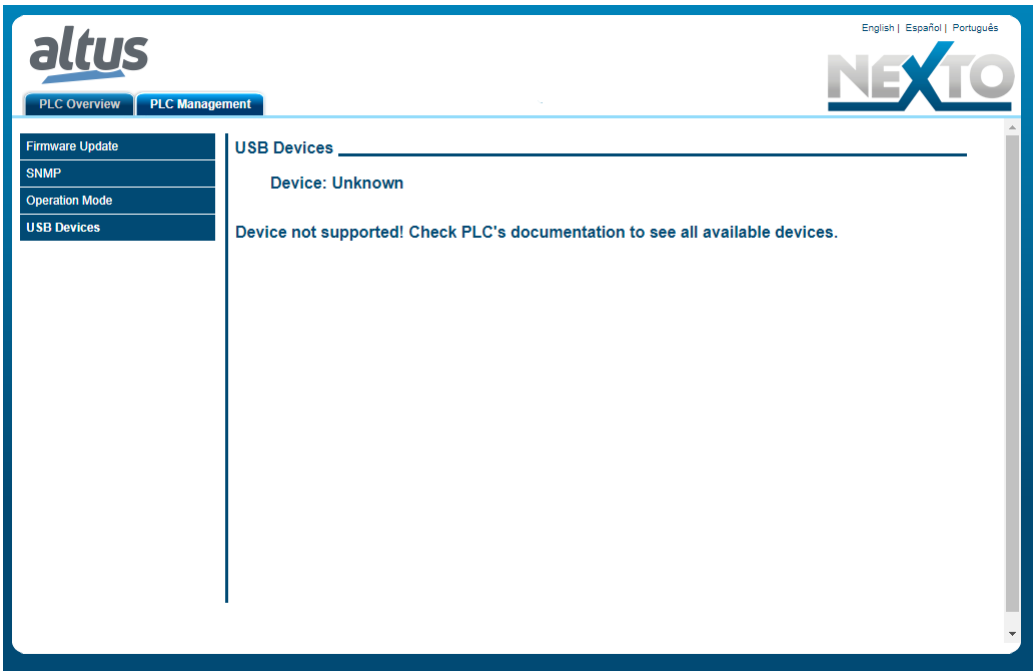


Figure 61: USB Devices - Unknown

5.6.1. Mass Storage Devices

5.6.1.1. General Storage

Mass storage devices can be used to expand the controller’s flash memory to store big amount of data, like on datalogger applications, for instance. To use a USB mass storage device, simply connect it to the USB port. After a few seconds, when the device is properly detected and mounted, the USB LED will turn on and the device information will appear on section *USB Devices* located at the tab *PLC Management* of the controller’s diagnostics webpage as shown below:

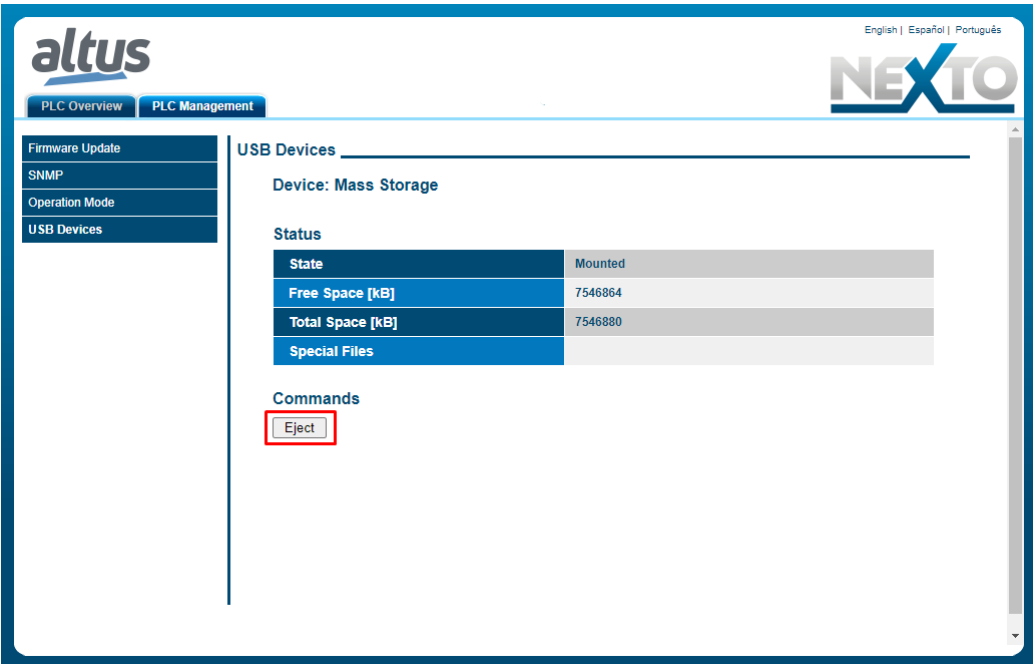


Figure 62: Mass Storage Device Information

The information shown on the status section of this page is also available in the symbolic variables diagnostics structure (see Section [Diagnostics via Variables](#)).

ATTENTION:
The USB mass storage device must be formatted as a FAT32 volume. Other filesystem formats are not supported.

The device can be ejected using the command provided on the *Commands* area of this page as indicated on the picture above.

After the device is properly detected and mounted, a new folder called *Mass_Storage* will appear on the controller's memory as shown on the picture below:

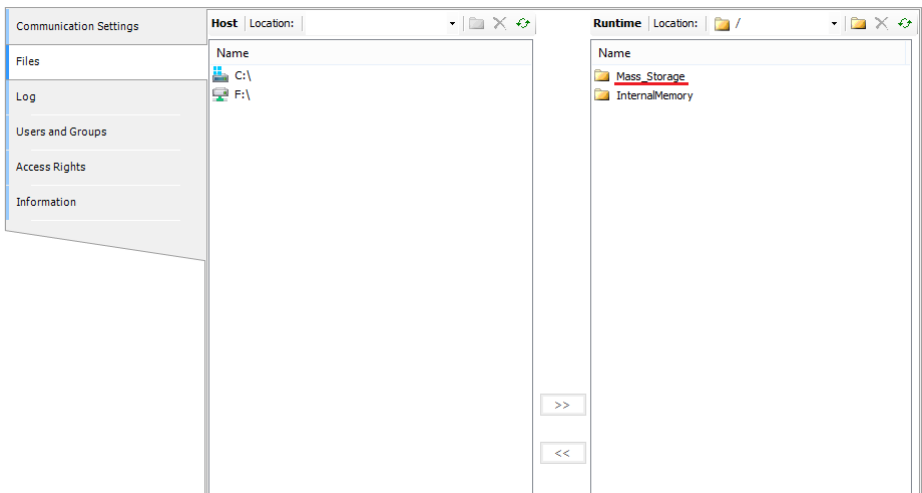


Figure 63: USB Mass Storage Folder

5.6.1.2. Not Loading the Application at Startup

The USB mass storage device can be used to prevent the controller from automatically loading the application after the power on. To do that, simply place an empty text file called "*dontbootapp.txt*" on the root folder of mass storage device. The presence of this file is informed in the *Special Files* field on controller's system webpage as shown below.

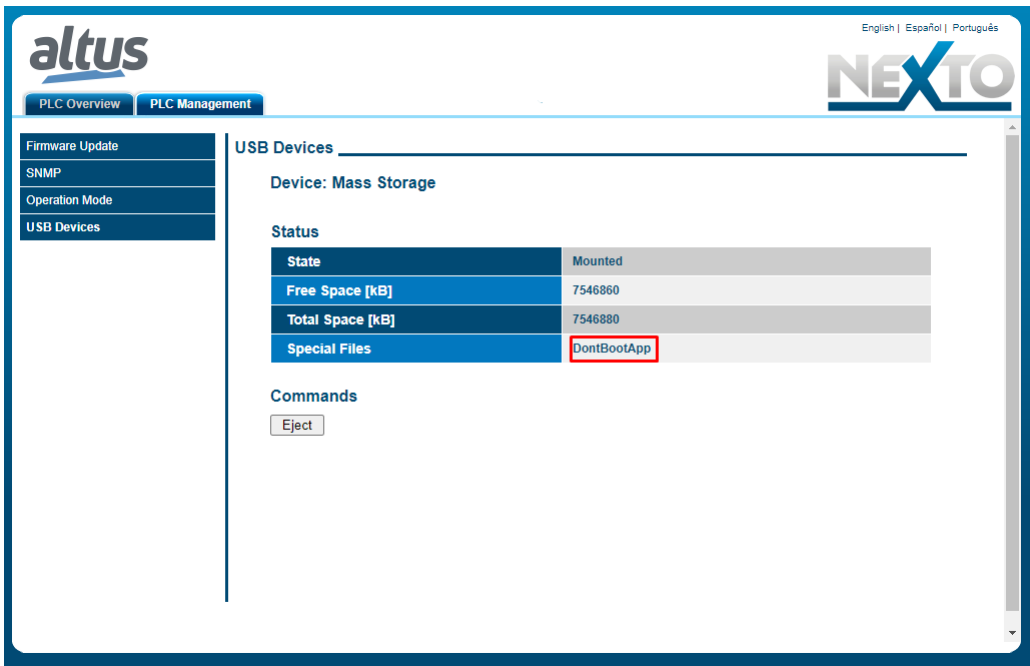


Figure 64: DontBootApplication in the Mass Storage Device

5.6.1.3. Transferring an Application from the USB device

The USB mass storage device can also be used to transfer an application to the controller. To do that, place the two files *Application.app* and *Application.crc* on the root folder of mass storage device (these files are created using MasterTool IEC XE executing the command *Online -> Create boot application* when offline). After the power on, if the controller detects the presence of these files on the USB mass storage device, the following sequence of actions will occur:

- The controller will start copy of the application from USB device to internal memory
- After finishing the copy process, the USB device will be ejected (USB LED will turn off)
- The new application will start (RUN) automatically (if "*dontbootapp.txt*" is not present)

The presence of the application is informed in the *Special Files* field on controller's system webpage as shown below:

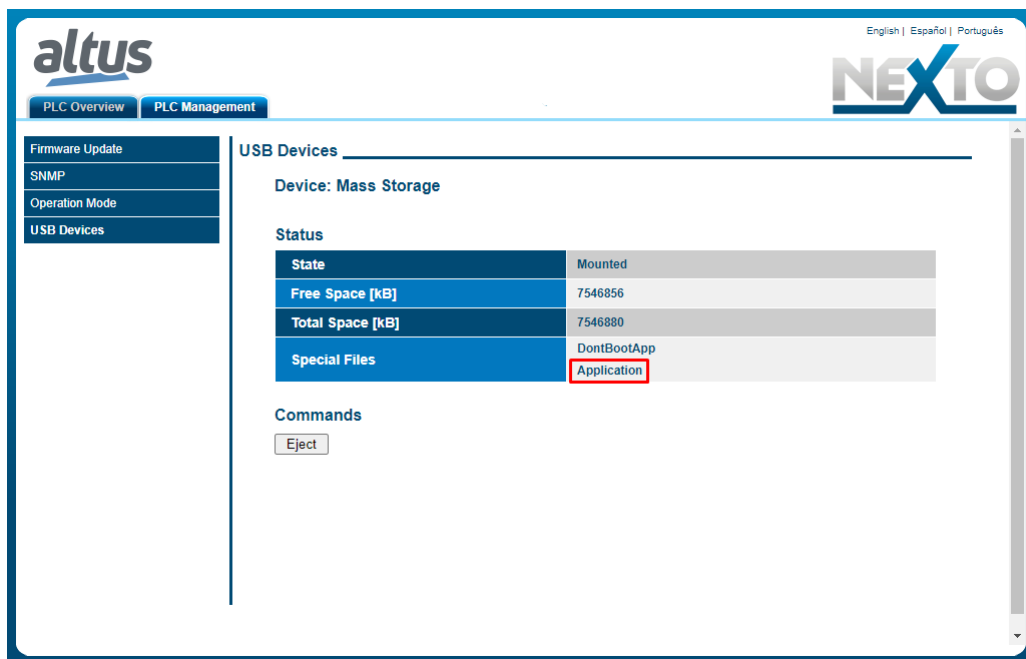


Figure 65: Application in the Mass Storage Device

Note that it is possible to have multiple *Special Files* in the same Mass Storage. In the example above, the PLC will transfer the new application to the internal memory but not load it on startup (hence, will not go to RUN).

5.6.2. USB to RS-232 Converters

Nexto Xpress allows to implement a RS-232 port using a USB to Serial converter. These converters are based on an internal controller chip. The following table shows the list of supported controllers:

Controller	Manufacturer
FT232	FTDI
PL2303	Prolific

Table 65: Supported USB to RS-232 converters

This port is intended to be used exclusively with the Serial communication function blocks provided by the *NextoSerial* library, allowing to implement a point-to-point communication with equipments that use simple protocols (non time critical) like *Radio modems*, *Barcode readers*, *RFID readers*, etc... Additionally, this kind of solution has the following limitations:

- **Baud Rate:** values lower than 4800 bps are not supported
- **Data Bits:** value "5" is not supported (only 6, 7 or 8)
- **Parity:** values "mark" and "space" are not supported (only Odd, Even and None)
- **Stop Bits:** value "1.5" is not supported (only 1 or 2)

After plugging the converter into the USB port, the USB LED may turn on indicating that the device was properly detected and mounted and the device information will appear on section *USB Devices* located at the tab *PLC Management* of the controller's diagnostics webpage as shown below:

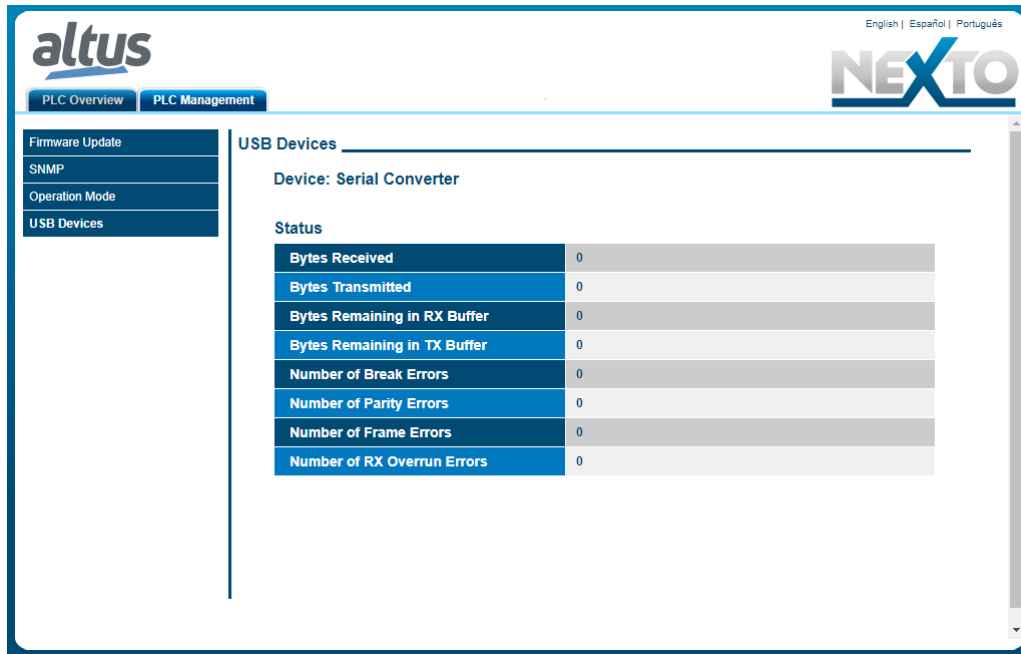


Figure 66: USB Devices - Serial Converter

The information shown on this page is also available in the symbolic variables diagnostics structure (see Section [Diagnostics via Variables](#)).

This additional serial port will be identified internally as *COM10*, and will not have a representation on the project treeview. From this point, this port can be used for communication using the *NextoSerial* functions similar to the native ports. For this kind of port, the handshake configuration is limited to *RS232_MANUAL* only (must be considered when configuring the port with *SERIAL_CFG* function).

5.6.3. Modem Devices

An USB Modem with a SIM chip can be used to connect the PLC to the internet using the cellular data network (telephone services, like sending SMS, are not implemented). This feature allows to use Nexto Xpress controllers to implement telemetry and IoT applications. There are basically two types modems: bridge and router. The following table shows the list of supported devices.

Model	Manufacturer	Type	Remarks
E303	Huawei	Bridge	-
E3276	Huawei	Bridge	-
E8372	Huawei	Router	Redirection of the configuration web page (button <i>Open Modem Configuration</i>) is not supported for this model. In this case, the modem configuration must be done externally by plugging it directly on a PC.

Table 66: Supported USB modems

The bridge modem is a non-managed device that implements a direct connection (pass through) to the mobile data network, so all the connection requests coming from the internet will reach the controller's operating system. For this kind of device, the configuration is performed through the controller's system web page as described further on this section.

On the other hand, the router modem is a managed device that implements a firewall with configurable network policies. For this kind of device, all the configuration is performed through a proprietary web page that is embedded into the device.

By default, the Modem blocks any kind of inbound connections (i.e, will not allow the remote access). To allow this, the user must configure the modem through the device embedded web page to open the TCP port related to the desired service (some manufacturers calls this feature as *Virtual Server* or *Port Forwarding*). The following table describes the number of the TCP port associated to the main services of the PLC.

Port	Service
80	Embedded system web page
8080	Embedded Webvisu web page
1217	MasterTool programming

Table 67: TCP ports of main PLC services

The management of the USB Modem functionality is done through the *USB Devices* page in the *PLC Management* tab of the controller's system webpage. Once the Modem device is properly detected and mounted, the USB LED will turn on and the device information will appear as shown below:

Figure 67: USB Modem Page Configuration

This area contains basically two sections: *Status* and *Configuration*.

The *Status* section is where all diagnostics related to the USB modem are displayed: configuration state (which does not depends on the device state), internet connection state and the modem IP address. These fields are updated automatically every time that a value is updated (does not require to reload the page manually). The same information is also provided on the symbolic variables diagnostics structure (see Section [Diagnostics via Variables](#)), which also contains the detailed description of the possible values for each field.

The *Configuration* section is where the user performs the modem configuration. For bridge modems, this section is used to enter the SIM chip information. For router modems, this section will show only a button called *Open Modem Configuration* that will redirect to the internal modem web page (check on the list of compatible devices if the redirection is supported, otherwise the modem must be configured externally on a PC). This section contains the following buttons:

- **Import:** loads the configuration from an external file
- **Export:** saves the current configuration on an external file
- **Reset:** erases the modem configuration from controller's memory
- **Apply:** write the configuration on the controller's memory

ATTENTION:

The provider APN and PIN code fields are mandatory for every SIM chip. If the provider informs these parameters, they shall be used. In other hand, it's known that several SIM chips simply doesn't care for the content of these fields, using internal predefined values. In this case, these fields of the web page can be left with the default values and the connection will proceed successfully. Values like "zero" for PIN Code and "empty" for the Provider APN are not allowed.

The operation principle of USB Modem functionality is rather simple. For bridge devices, once the device is properly detected and the SIM parameters configured, the controller starts a background process, which continuously controls the modem to keep it connected to the internet. This eliminates the need of any kind of manual intervention, so if the connection is dropped for some reason (bad reception, carrier outage, etc.), the controller will try to reconnect automatically. The status of this process can be observed by the Connection *Status* field. For router modems, the device contains a similar process that runs internally (independent from the controller) that is called *Auto Connect*. It generally comes enabled by default, but can be disabled through the modem web page.

One important aspect to consider is that, if the USB Modem is configured, the controller system will set it as the default gateway for all Ethernet communication. It means that, if the controller is simultaneously connected to a local network (NET1), which has also access to the internet, all the Ethernet messages addressed to external IPs will route through the USB Modem (and not through NET1). NET1 returns as the default gateway when the USB Modem was removed.

The configuration and operation sequence can be summarized into the following steps:

- Plug the device into the USB port. After some seconds, the *USB Devices* page will show the *Device* as "Modem". If not, the device might be unsupported or defective.
- (For bridge devices) Set the SIM parameters (*Provider APN* - Access Point Name and the chip *PIN code*) as informed by the SIM chip provider. After clicking on *Apply* button, the connection process will start. Once configured, it is not necessary to set this information anymore. It will be saved on the controller's memory.
- Watch the connection status on the corresponding field. If everything goes fine, after some seconds it will inform that the modem is connected to the internet, and an IP address will appear in its field.

Once connected to the internet, this communication channel can be used for several purposes. One typical use case is to implement telemetry solution using MQTT Client Function Block to publish data. Another use case is to access the controller remotely. In this case, it is necessary to know the modem IP address on the internet. However, this address is dynamic and changes on every connection process. One way to workaround this problem is to publish the IP address (available on the modem diagnostic variables) through MQTT.

With the Modem IP address, it is possible to perform remote access to the controller's system web page to view status and diagnostics (firmware update is not supported). Also, it allows to program the controller remotely using MasterTool IEC XE. For this, the gateway must be configured with the Modem IP address like shown on the following picture:

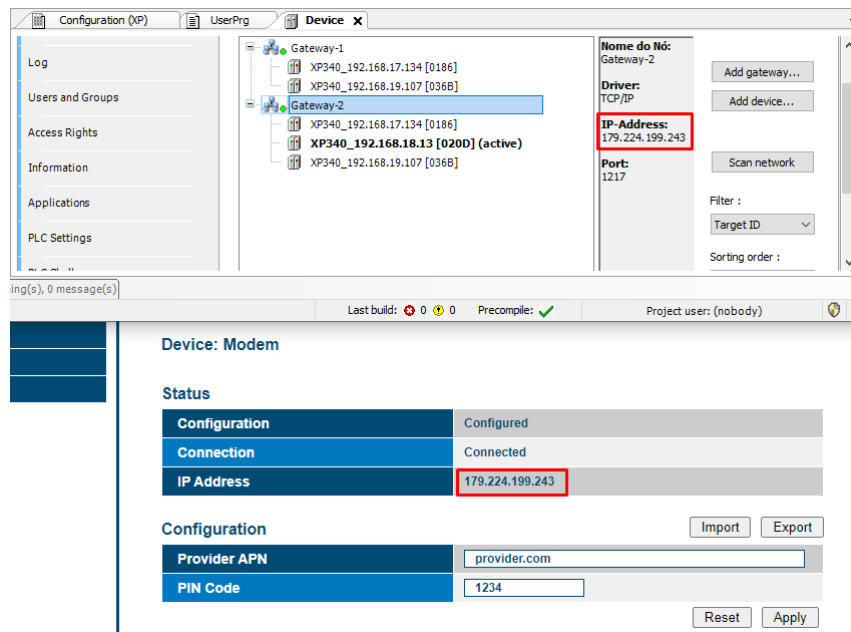


Figure 68: Configuring Gateway with Modem IP Address

ATTENTION:

For bridge devices, or router devices with external access enabled (port forwarding), once connected to the internet, anyone who knows the modem IP address will be able to access the controller remotely. So, for security reasons, it is EXTREMELY important and recommended to configure the User Rights on the controller to restrict the online operations of MasterTool IEC XE with login and password.

5.6.4. WiFi Adapters

An USB WiFi adapter can be used to connect the PLC to an existing WiFi network, creating a second network adapter that can be used for programming and communication. The following table shows the list of supported chipsets.

Chipset	Manufacturer	Example of comercial products
RTL8188EU	Realtek	TP-LINK model TL-WN725N LM Technologies model LM007
RT28xx	Ralink/Mediatek	D-Link model DWA-125
AR9271	Atheros/Qualcomm	TP-LINK model TL-WN721N

Table 68: Supported chipsets for USB WiFi adapters

The management of the WiFi Adapter functionality is done through the *USB Devices* page in the *PLC Management* tab of the controller's system webpage. Once the WiFi adapter device is properly detected and mounted, the USB LED will turn on and the device information will appear as shown below:

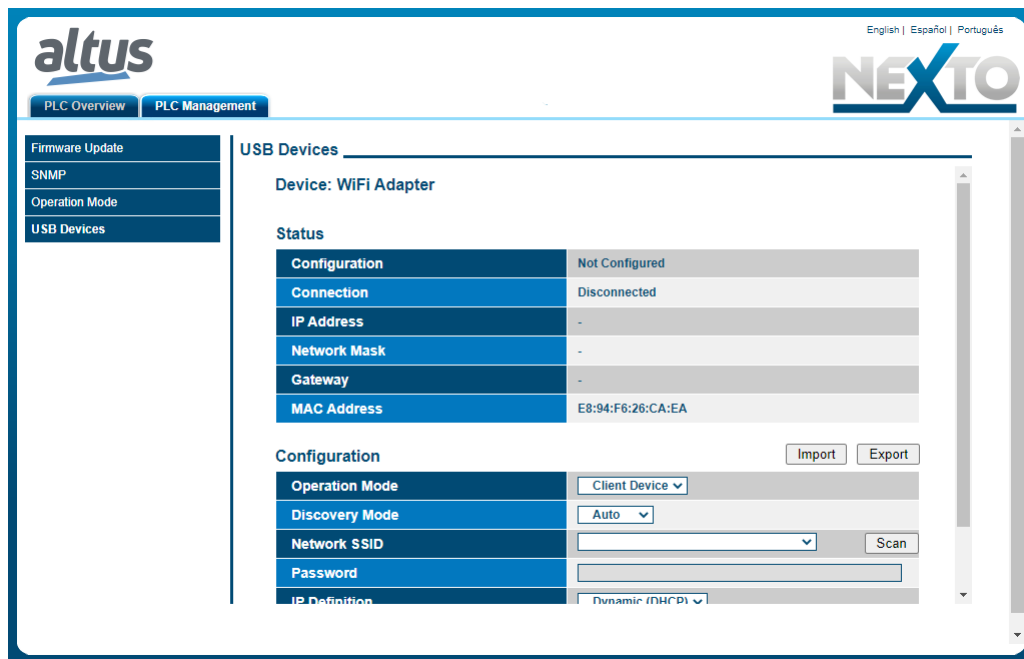


Figure 69: USB WiFi Adapter Page

This page contains basically two sections: *Status* and *Configuration*.

The *Status* section shows all diagnostics related to the WiFi adapter: configuration state, connection state, IP address, Netmask, Gateway and MAC address. These fields are updated as soon the values change. These informations are also provided in the symbolic variables diagnostics structure (see Section [Diagnostics via Variables](#)).

The *Configuration* section is composed by the following parameters:

- **Operation Mode:** defines how the WiFi adapter will operate (currently, only client mode is supported).
- **Discovery Mode:** defines what is the method to set the WiFi network. If selected as "Auto", the "Scan" button must be used to choose the wanted network. If selected as "Manual", the name of the SSID and Security Type must be entered manually.
- **Network SSID:** when discovery mode is set to "Auto", this field will show the available networks found on the scan process sorted from the best to the worst signal level (up to down). When discovery mode is set to "Manual" this field must be populated with the SSID of the wanted network.
- **Security Type:** this field is only available when the discovery mode is set to "Manual" (the "Auto" mode automatically selects the security type provided by the scanned network). This field defines the type of security used in the WiFi network, which can be "Public" or "WPA2-Personal".
- **Password:** this where you need to enter with the WiFi network password. The field will be automatically blocked if the Security Type is set as "Public" or the scanned network chosen does not use a security protocol.

- **IP Definition:** defines if the WiFi adapter will set the IP address dynamically (assigned by the network DHCP Server) or statically (where the user needs to enter the IP settings manually).
- **IP Address, Network Mask and Gateway:** only available when the IP definition is set as "Static". These fields will be used to configure the network parameters of the WiFi adapter.
- **Default Gateway:** this field defines what network interface will be used as a Gateway to access the Internet. It is possible to choose the "WiFi Adapter" or the "Local Ethernet" for this function.

ATTENTION:

For proper operation, the WiFi adapter network (defined by IP and Mask) must be different from the one configured for NET1.

The screenshot shows the 'altus NEXTO' web interface for PLC Management. The left sidebar has a menu with 'Firmware Update', 'SNMP', 'Operation Mode', and 'USB Devices'. The main content area is titled 'USB Devices' and contains a 'Configuration' section. This section has several fields: 'Operation Mode' (dropdown set to 'Client Device'), 'Discovery Mode' (dropdown set to 'Manual'), 'Network SSID' (text input), 'Security Type' (dropdown set to 'Public'), 'Password' (text input), 'IP Definition' (dropdown set to 'Static'), 'IP Address' (text input), 'Network Mask' (text input), 'Gateway' (text input), and 'Default Gateway' (dropdown set to 'Local Ethernet'). There are 'Import' and 'Export' buttons at the top right of the configuration section, and 'Reset' and 'Apply' buttons at the bottom right.

Figure 70: USB WiFi Adapter Configuration

Besides that, the *Configuration* section contains the following buttons:

- **Import:** loads a configuration file
- **Export:** download a file with the current configuration
- **Reset:** returns the configuration to the default
- **Apply:** apply the current configuration

The operation principle of USB WiFi adapter is similar to the USB Modem. Once the device is properly detected and the network parameters are configured, the controller will always try to keep connected to the WiFi network. The status of this process can be observed by the Connection Status field.

ATTENTION:

If the Default Gateway was set as "WiFi Adapter", the Gateway of the NET1 diagnostics of the MasterTool will show zero (0.0.0.0). Otherwise, if it was set as "Local Ethernet", the WiFi adapter Gateway will be zero.

The configuration and operation sequence can be summarized into the following steps:

- Plug the device into the USB port. After some seconds, the *USB Devices* page will show the *Device* as "WiFi Adapter". If not, the device might be unsupported or defective.
- Set the network configuration. After clicking on Apply button, the connection process will start. Once configured, it is not necessary to set this information anymore. It will be saved on the controller's memory.
- Watch the connection status on the corresponding field. If everything goes fine, after some seconds it will inform that the adapter is connected to the WiFi network.

The following picture shows the page of a controller connected to a WiFi network:

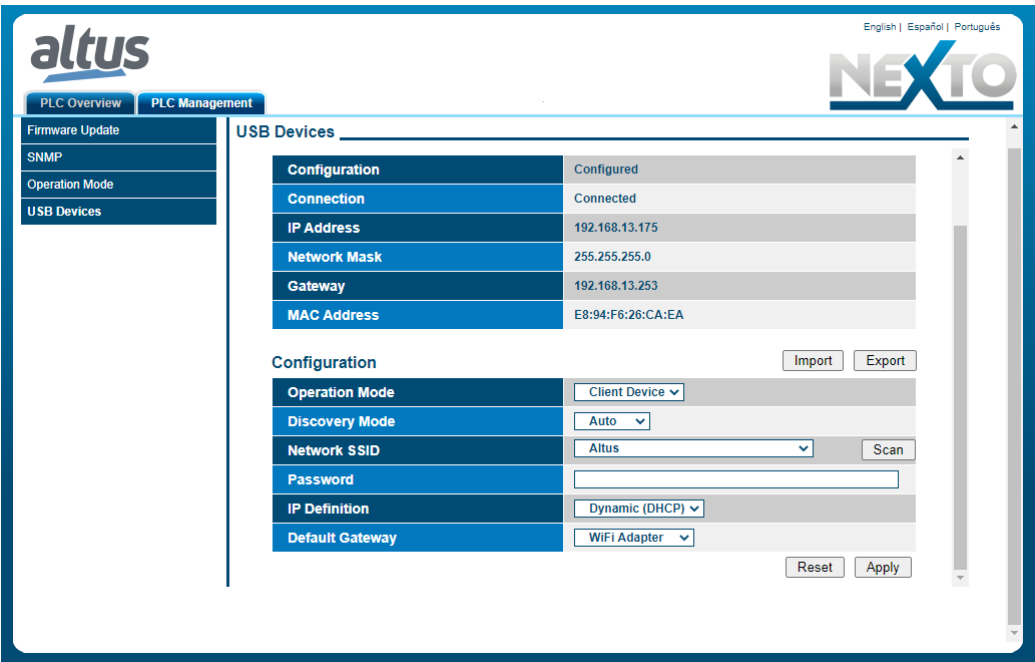


Figure 71: USB WiFi Adapter Connected to a Network

Once connected to the WiFi network, this communication channel can be used for several purposes. To program the PLC with MasterTool IEC XE, the gateway must be configured with the IP address assigned to the WiFi adapter (similar to the USB Modem, Figure 68). This IP address can also be used to access the controller’s system web page, where it is possible to perform a firmware update, which is not available when using the USB Modem. Additionally, this communication channel can also be used with the MQTT client Function Block to report data to an external broker on the internet (in this case, the WiFi adapter as Default Gateway).

5.7. Communication Protocols

Nexto Xpress controllers offers several communication protocols, including MODBUS (exclusively Symbolic), OPC UA and other. The following table describes the configuration limits:

	XP300	XP315	XP325	XP340
Mapped Points	20480			
Mappings	5120			
Requests	512			
NETs – Client or Server instances	4			
COM (n) – Master or Slave instances	1			
Control Centers	-	-	-	3

Table 69: Protocol Limits per CPU

Notes:

Mapped Points: Each variable or item of a given data type is assumed to be a mapping. The same is considered for each position of the ARRAY type. This means that if a simple variable is declared, it will be considered a mapping and if an ARRAY type is declared, the count will be equal to the size of the declared ARRAY. The amount of mappings increments by one when there is a simple type of variable being declared independent of the size of the given type. Then, mapping a variable of INT type (16-bit) in a Holding Register of symbolic Modbus drivers or a variable of type LINT (64-bit) in four Holding Register of symbolic Modbus drivers is accounted for as just a mapping.

Mappings: A mapping is a relationship between an application intern variable and an application protocol object. The limit value for the project mappings corresponds to the sum of all the mappings made within the instances of communication protocols and their respective devices.

Requests: The sum of requests for communication protocols, declared on the devices, cannot exceed the maximum number of requests supported by the CPU.

Control Centers: *Control Center* is all client device connected to the CPU through protocol IEC 60870-5-104. This field informs the maximum of client devices of control center type supported by the CPU. Correspond to the sum of all client devices of communication protocol Server IEC 60870-5-104 (does not include master or clients from MODBUS RTU Slave and MODBUS Server protocols)

The following additional information can be found on this same section of Nexto Series CPUs User Manual code MU214605:

- Detailed limits of MODBUS Symbol protocol
- Detailed limits of IEC 60870-5-104 Server protocol
- Protocol Behavior x CPU State
- Double Points
- CPU's Events Queue
- Interception of Commands Coming from the Control Center

5.7.1. MODBUS RTU MASTER

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.2. MODBUS RTU SLAVE

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.3. MODBUS ETHERNET

Nexto Xpress controllers supports only the newest MODBUS Symbolic Client/Server drivers. These protocols are the same ones available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.4. OPC DA Server

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.5. OPC UA Server

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.6. EtherNet/IP

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.7. IEC 60870-5-104 Server

This protocol is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.7.8. CANopen Manager

CANopen is a protocol based on CAN bus which provides fast I/O update (around 5 ms for a 1000 kbit/s network with a few slaves) with a simple twisted pair physical bus infrastructure.

The CANopen Manager (master) is responsible for controlling the slave devices, managing their operation state and exchanging I/O and other service data. By default, the CANopen manager protocol activities (bus cycle) are executed on the context of MainTask, keeping it synchronous with the execution of application code.

The configuration of CANopen network is performed with the support of EDS files, which describes the I/O data and service objects (PDO and SDO) of the slave and must be provided by the device manufacturer.

Additionally, an application library called CiA405 is provided with FunctionBlocks which allows to perform several specific actions like changing the slave state (NMT), receiving emergency object, querying the slave state and performing SDO read/write commands. The complete description of CiA405 library can be found on *Online Help (F1)* of MasterTool IEC XE.

ATTENTION:

- Only one CANopen Manager instance per project is allowed
- Although CANopen specification allows up to 127 nodes (including Manager), applications with Nexto Xpress must not exceed a maximum of 64 slave devices.

A special care must be taken considering the physical bus length and the selected baudrate. The following table shows the maximum bus length that can be used safely with a given baudrate:

Baudrate	Maximum bus length
1000 kbit/s	25 m
800 kbit/s	50 m
500 kbit/s	100 m
250 kbit/s	250 m
125 kbit/s	500 m
100 kbit/s	700 m
< 50 kbit/s	1000 m

Table 70: Baudrate vs Bus Length

5.7.8.1. Installing and inserting CANopen Devices

The configuration of a CANopen network uses the same standard procedure of other fieldbuses configuration on MasterTool IEC XE.

To add a CANopen Manager, right-click on the CAN interface and select *Add Device*. Expand the items until finding *CANopen_Manager* device and click on the *Add Device* button. The *CANopen Manager* device will appear below the CAN interface as shown on the following picture:

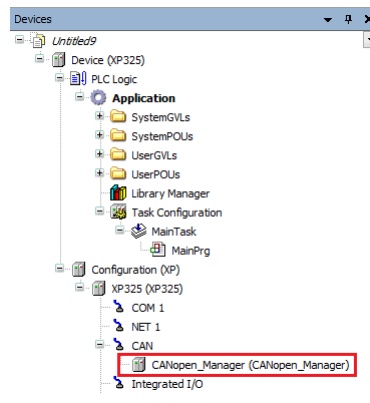


Figure 72: Adding CANopen Manager

To add a CANopen slave device, first you need to install it on the *Device Repository*. To do that, go to *Tools -> Device Repository* and install the device EDS file.

After that, right-click on the *CANopen_Manager* device and click on *Add Device*. Search the devices you desire and click on *Add Device* button like shown on the following picture:

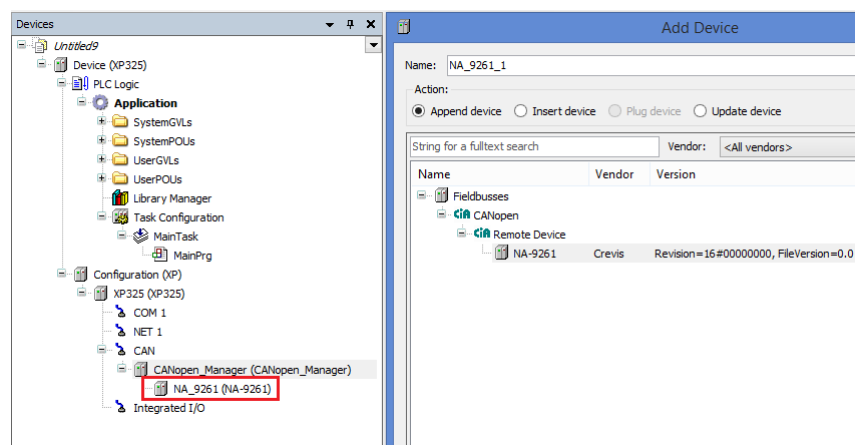


Figure 73: Adding CANopen Slave Device

5.7.8.2. CANopen Manager Configuration

The CANopen Manager comes with a ready-to-use configuration (default values). Typically, it is just needed to set the correct baudrate and slave address to have a network running.

The main parameters of CANopen manager are located at *General* tab:

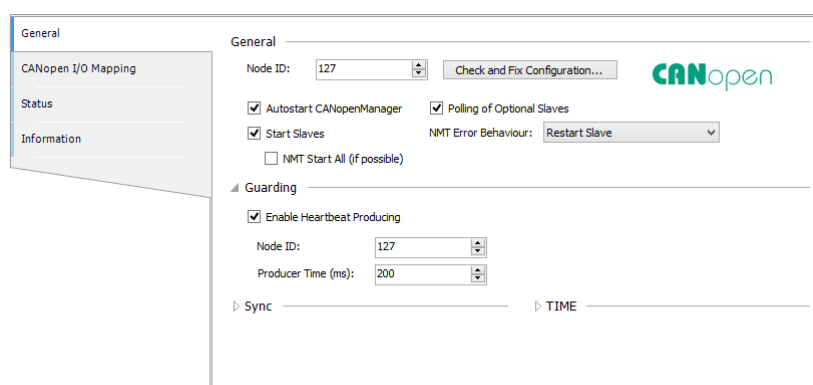


Figure 74: CANopen Manager general parameters

The detailed description of CANopen Manager general parameters can be found on section *Device Editors -> CANopen* of MasterTool IEC XE Online Help (F1).

Additionally, the tab *CANopen I/O Mapping* allows to change the bus cycle task:

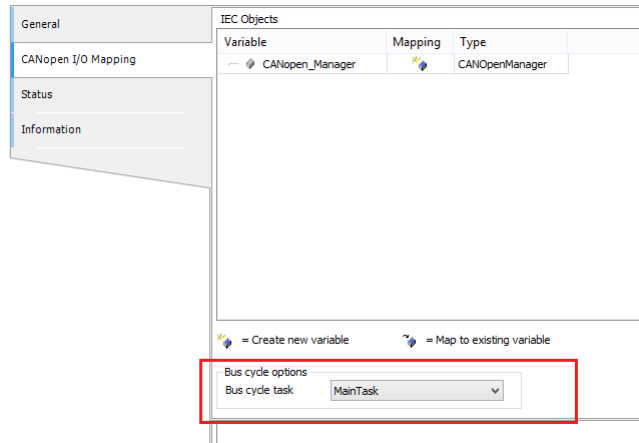


Figure 75: CANopen Manager bus cycle task setting

By default, the bus cycle task is configured to use the MainTask. This is the recommended setting for most of the applications. Changing this setting is only required on a very specific scenario which requires the implementation of a time-critical control loop using CANopen I/O (5ms lets say) that can not be performed on MainTask due to heavy application code.

5.7.8.3. CANopen Slave Configuration

The configuration of CANopen Remote Devices (Slaves) is separated in the first four tabs shown on the following picture:

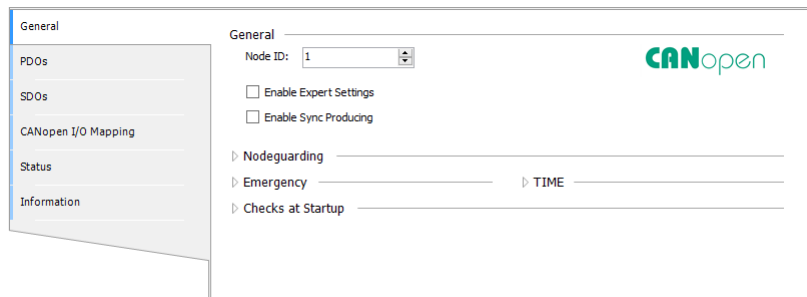


Figure 76: CANopen Slave parameters

The *General* tab contains the slave address (Node ID), Nodeguarding and Emergency object settings.

The *PDO* tab contains the configuration of process data (I/O data) that will be exchanged.

The *SDO* tab contains the SDO objects which can be selected to be accessed by SDO read/write FunctionBlock provided by CiA405 library.

The detailed description of CANopen Slave parameters can be found on section "*Device Editors -> CANopen*" of MasterTool IEC XE Online Help (F1).

5.8. Remote I/O Mode

Nexto Xpress controllers have a remote operation mode, which is used as I/O expansion. This expansion is based in CANopen protocol. When the controller is in remote mode, it isn't a standard PLC, operating only as a remote slave. To configure your Xpress as a remote I/O expansion, access the product Web page *PLC Management*, in the *Operation Mode* tab.

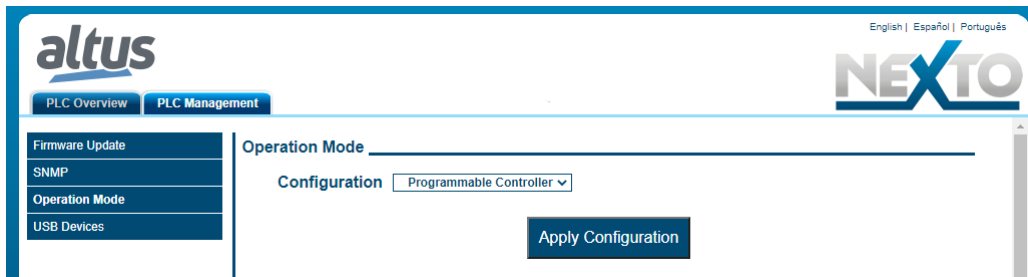


Figure 77: Remote I/O Configuration Screen

In this tab, it's possible to choose the controller operation mode through the combo box. This is available only when the controller is in STOP. Use the *Apply Configuration* button to change to the desired mode. The Xpress will reboot and configure the new operation mode. The available options are:

- **Programmable Controller:** default controller function, which can be programmed according to user needs.
- **CANopen Slave:** remote I/O expansion function, where the controller becomes a CANopen slave, which can communicate to other controllers through CANopen Manager.

ATTENTION:

The remote operation mode uses an application developed only for I/O expansion, which runs in a 5 ms MainTask cycle. It's not possible change or download an application in this mode.

When in remote operation mode, some features of the controller will be modified. The controller can't be found by the MasterTool. However, it's possible to find the device via *Easy Connection*, even change its IP, without erase the application. Besides that, in the *Firmware Update* tab on the Web page, the *Erase Application* option is unavailable.

5.8.1. CANopen Slave

To use the expansion mode as a CANopen Slave, first, change the *Operation Mode* to *CANopen Slave*, in *Operation Mode* on the Web page. Next, make the configurations of the *CANopen Slave*: configure the network (*IP Address*, *Network Mask* and *Gateway*); configure the CANopen parameters (*Node ID*, *Baudrate* and *Termination*); and the I/O configuration (according to the controller). These settings are similar to a typical application.

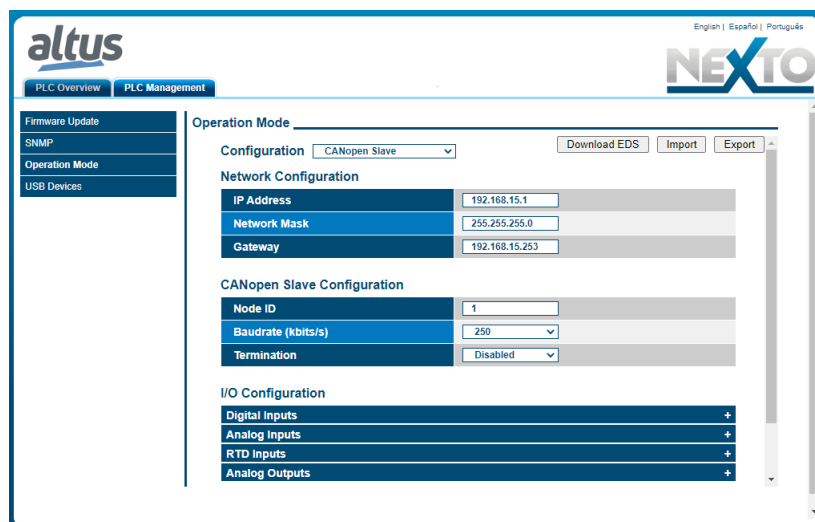


Figure 78: CANopen Slave Remote Configuration Screen

Click in the items with the + on the right to expand the configuration panel. All parameters shown in the *I/O Configuration* are the same mentioned in the [Integrated I/O](#) section. While the CANopen Slave Configuration parameters are the same of those in the [CANopen Manager](#) section.

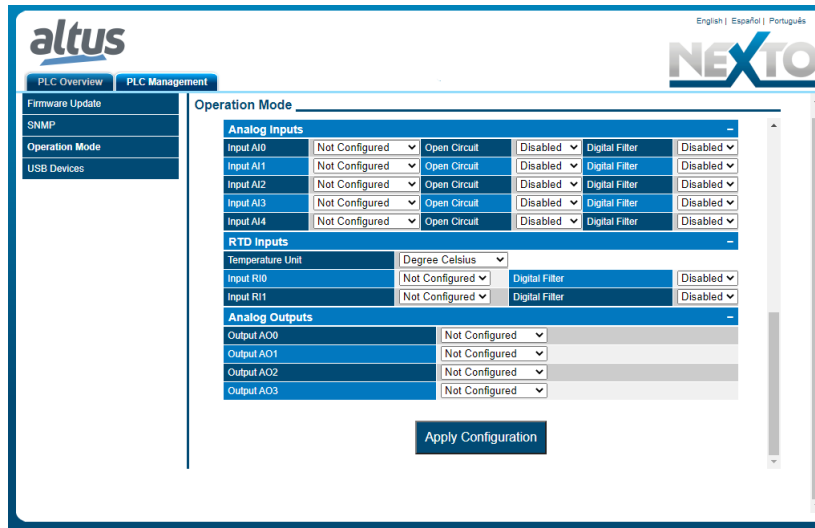


Figure 79: Expanded I/O Configuration Screen

After the configuration step, it's possible to use the *Export Configuration* button to download a file called *WebRemote-Configuration.config*. This file contains all parameters configured in the *Remote I/O* Configuration screen. You can use this file to load the configuration through the *Import Configuration* button. Besides that, you can download the CANopen Slave Electronic Data Sheet (EDS) file directly by the *Download EDS* button in the Web.

When the configuration is done, click in *Apply Configuration* to reboot the controller with the new settings. The Web page will automatically reload to the configured IP. The mode change can be confirmed by the CANopen Slave in the Operation Mode field, in the *PLC Overview* tab.

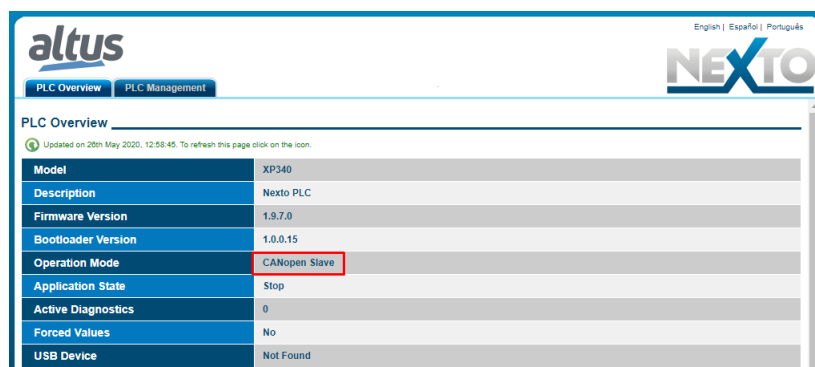


Figure 80: Operation Mode in the PLC Overview Screen

Therefore, it's possible to use a controller with the CANopen Manager feature (e.g. XP325) to access the CANopen Slave I/O. See the [CANopen Manager](#) section in this document to learn how use this feature. The CANopen Slave Remote PDOs are organized as follow:

Variable Name	Representation	Variable Type
Digital_Outputs_1	Q0 Group	USINT - 8 bits
Digital_Outputs_2	Q1 Group	USINT - 8 bits
Analog_Outputs_1	AO0	INT - 16 bits
Analog_Outputs_2	AO1	INT - 16 bits
Analog_Outputs_3	AO2	INT - 16 bits

Variable Name	Representation	Variable Type
Analog_Outputs_4	AO3	INT - 16 bits
Digital_Inputs_1	I0 Group	USINT - 8 bits
Digital_Inputs_2	I1 Group	USINT - 8 bits
Analog_Inputs_1	AI0	INT - 16 bits
Analog_Inputs_2	AI1	INT - 16 bits
Analog_Inputs_3	AI2	INT - 16 bits
Analog_Inputs_4	AI3	INT - 16 bits
Analog_Inputs_5	AI4	INT - 16 bits
RTD_Inputs_1	RI0	INT - 16 bits
RTD_Inputs_2	RI1	INT - 16 bits
Diagnostics_Analog_Inputs_1	AI0 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Inputs_2	AI1 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Inputs_3	AI2 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Inputs_4	AI3 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Inputs_5	AI4 Diagnostics	USINT - 8 bits
Diagnostics_RTD_Inputs_1	RI0 Diagnostics	USINT - 8 bits
Diagnostics_RTD_Inputs_2	RI1 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Outputs_1	AO0 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Outputs_2	AO1 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Outputs_3	AO2 Diagnostics	USINT - 8 bits
Diagnostics_Analog_Outputs_4	AO3 Diagnostics	USINT - 8 bits

Table 71: CANopen Slave Remote PDOs Organization

Digital I/Os are accessed by groups. They use a byte, where each bit is an digital I/O (e.g. I00 is the less significant bit - LSB - and I07, the most significant bit - MSB). Analog I/Os are transmitted/received directly through an integer. And the diagnostics of each analog I/O are received in a byte, according to the following tables:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	bOpenLoop	bOverRange	bInputNotEnable

Table 72: AIx Diagnostics

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	bUnderRange	bOverRange	bInputNotEnable

Table 73: RIx Diagnostics

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	bShortCircuit	bOpenLoop	bOutputNotEnable

Table 74: AOx Diagnostics

ATTENTION:
PDOs can't be edited or removed from the CANopen Slave. It's not possible to create your own CANopen slave device.

How the CANopen Slave is not accessible by the user via MasterTool, the RUN and STOP state of the application are controlled by the CANopen slave operation state. To put the CANopen Slave in RUN, it's necessary to set the state to *Operational* (green symbol next to the device). To put it in STOP, you need to use the NMT Function Block of the CiA405 library - see

Online Help (F1) - to change the CANopen slave operation state (recommended). Or, you can remove the CAN connector of the remote controller. The CAN LED can keep blinking once it indicates the message transmission and reception, not the operation state of the CANopen protocol.

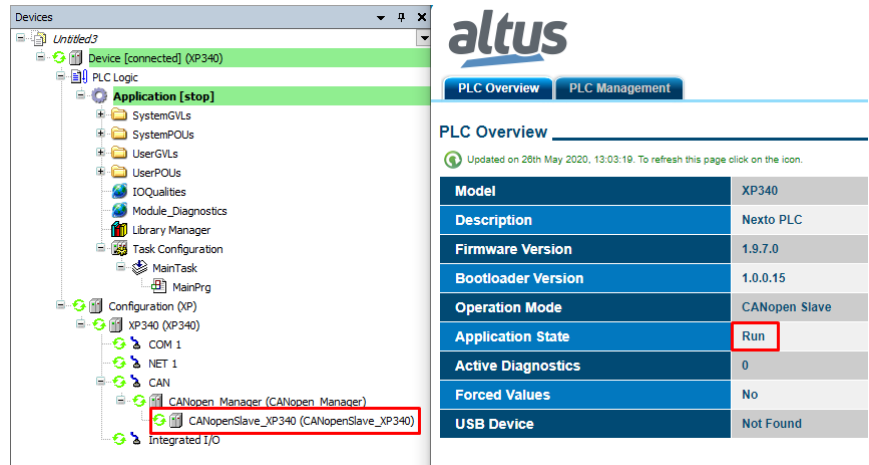


Figure 81: CANopen Slave in Operational - RUN

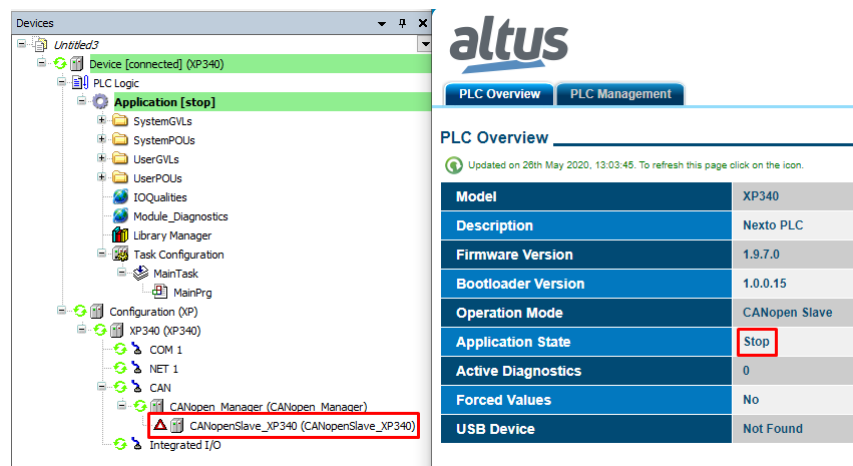


Figure 82: CANopen Slave in Pre-Operational - STOP

5.9. User Web Pages

Also called *Web Visualization*, or simply *Webvisu*, this feature allows to implement a simplified SCADA embedded into the PLC. The Visualization screens are developed on the same environment of the PLC application using MasterTool IEC XE. Once the application is downloaded, the PLC starts a web server hosting this special web page.

The complete information about this functionality can be found on Help of MasterTool IEC XE.

5.10. SNMP

5.10.1. Introduction

SNMP (Simple Network Management Protocol) is a protocol widely used by network administrators to provide important information and diagnostic equipment present in a given Ethernet network.

This protocol uses the concept of agent and manager, in which the manager sends read requests or write certain objects to the agent. Through a MIB (Management Information Base) the manager is aware of existing objects in the agent, and thus can make requests of these objects, respecting the read permissions or writing the same.

MIB is a collection of information organized hierarchically in which each object of this tree is called OID (Object Identifier). For all equipments with SNMP, it is mandatory to support MIB-II, which have key information for managing Ethernet networks.

5.10.2. SNMP in Nexto Xpress Controllers

The Nexto Xpress controllers behaves as agents in SNMP communication, with support for protocols SNMPv1, SNMPv2c, SNMPv3 and support the MIB-II, where required objects are described in RFC-1213. The information provided by the SNMP cannot be manipulated or accessed through the user application, requiring an external SNMP manager to perform access. The following table describes the objects available in Nexto Xpress controllers.

OID	Name	Description
1.3.6.1.2.1.1	System	Contains name, description, location and other equipment identification information.
1.3.6.1.2.1.2	Interfaces	Contains information of the machine's network interfaces. The ifTable table (OID 1.3.6.1.2.1.2.2) has the indexes 6 and 7 available, which can be viewed by the network interfaces statistics NET 1 and NET 2, respectively, of the Nexto series CPUs.
1.3.6.1.2.1.3	At	Contains information of the last required connections to the agent.
1.3.6.1.2.1.4	IP	Contains statistical connections using IP protocol.
1.3.6.1.2.1.5	ICMP	Contains statistical connections using ICMP protocol.
1.3.6.1.2.1.6	TCP	Contains statistical connections using TCP protocol.
1.3.6.1.2.1.7	UDP	Contains statistical connections using UDP protocol.
1.3.6.1.2.1.11	SNMP	Contains statistical connections using SNMP protocol.
1.3.6.1.2.1.31	ifMib	Extension to Interfaces, OID 1.3.6.1.2.1.2

Table 75: MIB II Objects – Nexto Xpress SNMP Agent

By default, the SNMP agent is activated, i.e., the service is initialized at the time the controller is started. The access to the agent information is via the Ethernet interface, TCP port 161. The following figure shows an example of an SNMP manager reading some values.

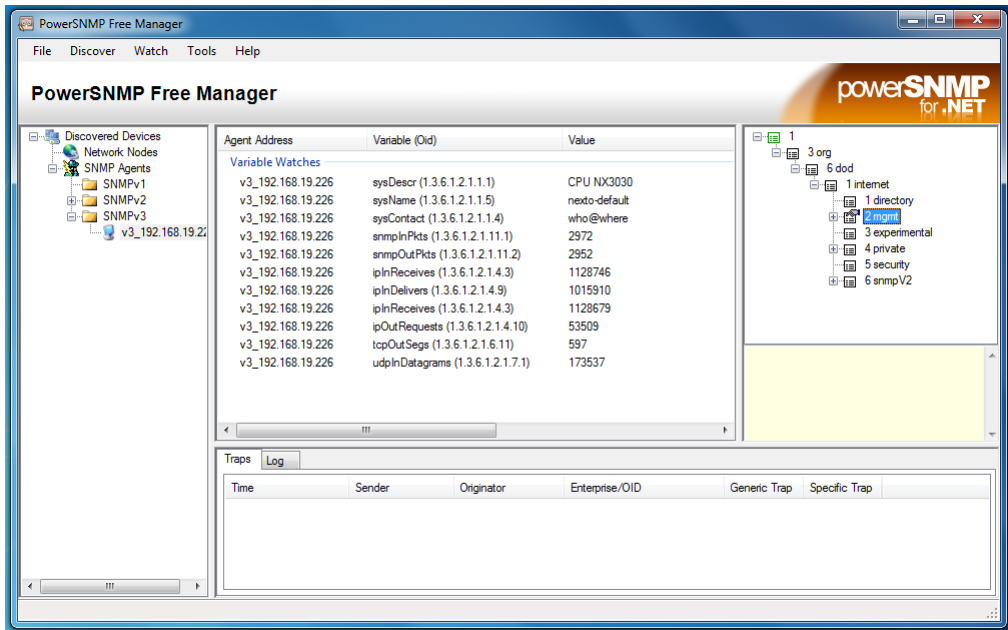


Figure 83: SNMP Manager Example

For SNMPv3, in which there is user authentication and password to requests via SNMP protocol, is provided a standard user described in the [User and SNMP Communities](#) section.

If you want to disable the service, change the SNMPv3 user or communities for SNMPv1 / v2c predefined, you must access the controller's web page as described on the following section.

5.10.3. Configuration

SNMP settings can be changed through the controller's web page, in the PLC Management tab. To access the settings, you must first log in, as shown in the figure below.

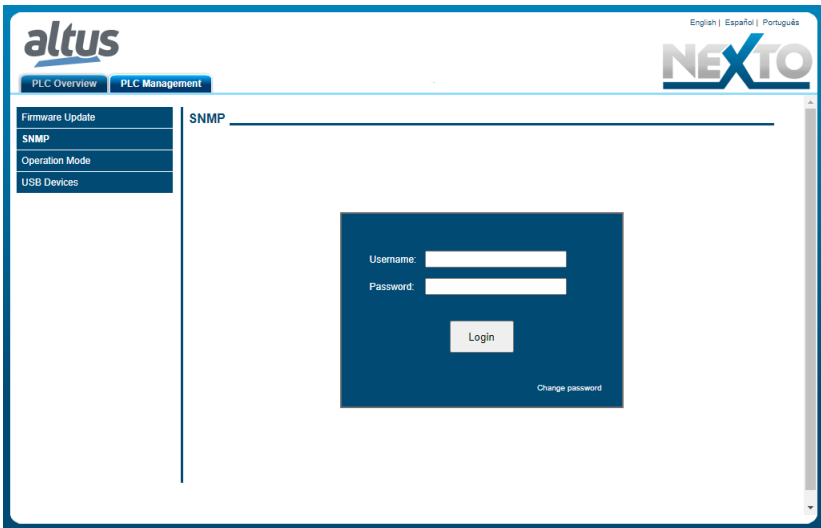


Figure 84: SNMP Login screen

After a successful login, the current state of the service (enabled or disabled) as well as the user information SNMPv3 and communities for SNMPv1 / v2c can be viewed.

The user can enable or disable the service via a checkbox at the top of the screen.

It's also possible to change the SNMPv3 information by clicking the *Change* button just below the user information. Will open a form where you must complete the old username and password, and the new username and password. The other user information SNMPv3 cannot be changed.

To change the data of SNMPv1/v2c communities, the process is similar, just click the *Change* button below the information community. A new screen will open where the new data to the rocommunity fields and rwcommunity will be inserted. If you fail any of the fields blank, their community will be disabled. That way, if the user leaves the two fields blank, access to the SNMP agent will only be possible through SNMPv3.

If the user wants to return to the default settings, it must be manually reconfigure the same according to the [User and SNMP Communities](#) section. Therefore, all current SNMP configurations will be kept in the firmware update process. These options are shown on the figure below.

Figure 85: SNMP status configuration screen

ATTENTION:
The user and password to login on the SNMP settings web page and to access the agent via SNMP protocol are the same.

5.10.4. User and SNMP Communities

To access the SNMPv1 / v2c of the Nexto Xpress controllers, there are two communities, according to following table.

Communities	Default String	Type
rocommunity	Public	Only read
rwcommunity	Private	Read and Write

Table 76: SNMP v1/v2c Default Communities info

It's possible to access SNMP v3 using default user, see table below:

User	Type	Authentication Protocol	Authentication Password	Private Protocol	Private word	Pass-
administrator	rwuser	MD5	administrator	-	-	

Table 77: SNMP v3 User info

For all settings of communities, user and password, some limits must be respected, as described on the following table:

Configurable item	Minimum Size	Max Size	Allowed Characters
rocommunity	-	30	[0-9][a-z][A-Z]@\$*._.
rwcommunity	-	30	[0-9][a-z][A-Z]@\$*._.
V3 User	-	30	[0-9][a-z][A-Z]@\$*._.
Password v3	8	30	[0-9][a-z][A-Z]@\$*._.

Table 78: SNMP settings limits

5.11. RTC Clock

Nexto Xpress controllers have an internal clock that can be used through the *NextoStandard.lib* library. This functionality is the same one available on other Nexto Series devices, which is described on this same section of Nexto Series CPUs User Manual code MU214605.

5.12. Function Blocks and Functions

The Function Blocks and Functions available for Nexto Xpress controllers are the same ones provided for Nexto Series CPUs, which are described on this same section of Nexto Series CPUs User Manual code MU214605.

6. Maintenance

6.1. Diagnostics

Nexto Xpress controllers permit many ways to visualize the diagnostics generated by the system, which are:

- [Diagnostics via LED](#)
- [Diagnostics via WEB](#)
- [Diagnostics via Variables](#)
- [Diagnostics via Function Blocks](#)

The first one is purely visual, generated through two LEDs placed on the front panel (PWR and DG). The next feature is the graphic visualization in a WEB page. The diagnostics are also provided as global symbolic variables to be used on the user application, for instance, being presented in a supervisory system. The last ones present specific conditions of the system functioning.

These diagnostics function is to point possible system installation or configuration problems, and communication network problems or deficiency.

6.1.1. Diagnostics via LED

Nexto Xpress controllers have a power (PWR) and a diagnostic indication (DG) LEDs. The following table shows the meaning of each state and its respective descriptions:

PWR	DG	Description	Causes	Priority
Off	Off	Not used	No power supply or Hardware problem	-
On	Off	Controller is booting	-	-
On	On	CPU is in RUN state, and there are no active diagnostics	-	5 (low)
On	Blinking 1x	CPU is in STOP state or no application loaded	-	2
On	Blinking 2x	There are active diagnostics	-	3
On	Blinking 3x	Data forcing	Some memory area is being forced by the user through MasterTool IEC XE	4
On	Blinking 4x	Hardware error	Internal hardware error	1
On	Blinking 5x	Power Failure	External power supply voltage is lower than acceptable threshold	0 (high)

Table 79: Description of the Diagnostic LEDs States

6.1.2. Diagnostics via WEB

As already known on Nexto Series, Nexto Xpress provides access to the system diagnostics and operation states through a WEB page.

The utilization, and dynamics, is very intuitive and facilitates the user operations. The use of a supervisory system can be replaced when it is restricted to system status verification.

To access the controller WEB page, it is just to use a standard navigator (Internet Explorer 7 or superior, Mozilla Firefox 3.0 or superior and Google Chrome 8 or superior) and type, on the address bar, the controller IP address (e.g. Ex.: <http://192.168.15.1>). First, the controller information is presented, according to figure below:

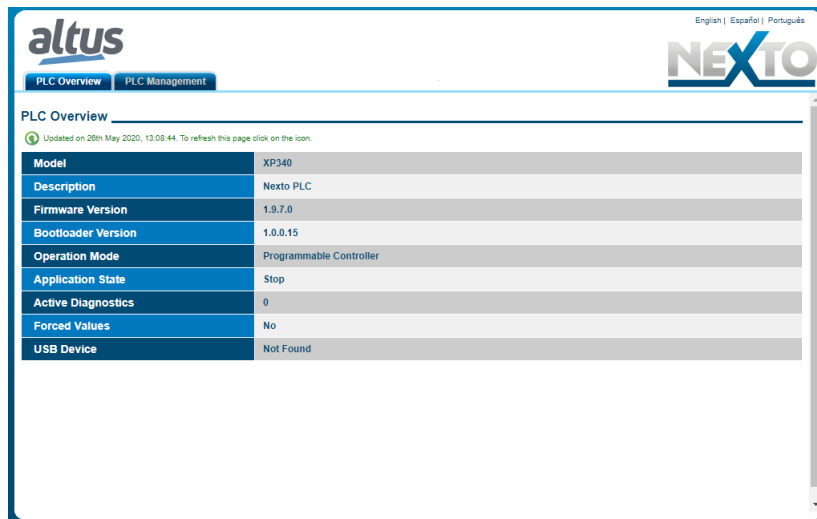


Figure 86: Initial Screen

The user can choose from three language options: Portuguese, English and Spanish. The desired language is selected through the upper right menu. Additionally, the management tab has other features like *Firmware Update* and *SNMP*.

Firmware Update tab is restricted to the user, that is, only for internal use of Altus. In cases where the update is performed remotely (via a radio or satellite connection for example), the minimum speed of the link must be 128Kbps.

6.1.3. Diagnostics via Variables

Nexto Xpress controllers offers a set of global symbolic variables, which provides several diagnostics information related to the hardware and software. These symbolic data structures are automatically created by the MasterTool IEC XE.

6.1.3.1. Summarized Diagnostics

The following table shows the meaning of summarized diagnostics:

DG_XP3xx.tSummarized.*	Type	Description
bHardwareFailure	BOOL	TRUE – Controller has internal hardware failure.
		FALSE – The hardware is working properly.
bSoftwareException	BOOL	TRUE – One or more exceptions generated by the software.
		FALSE – No exceptions generated in the software.
bCOM1ConfigError	BOOL	TRUE – Error during/after the COM 1 serial interface configuration.
		FALSE – Correct COM 1 serial interface configuration.
bNET1ConfigError	BOOL	TRUE – Error during/after the NET 1 Ethernet interface configuration.
		FALSE – Correct NET 1 Ethernet interface configuration.
bInvalidDateTime	BOOL	TRUE – Invalid date/hour.
		FALSE – Correct date/hour.
bRuntimeReset	BOOL	TRUE – The RTS (Runtime System) has been restarted at least once. This diagnostics is only cleared during the system restart.
		FALSE – The RTS (Runtime System) is operating normally.
bRetentivityError	BOOL	TRUE - Error occurred while saving the retentive data.
		FALSE – Valid data in the retentive memory during start up.
bIntegratedIODiagnostic	BOOL	TRUE - There is some diagnostic in the Integrated I/O (see detailed)
		FALSE – No diagnostic in the Integrated I/O
bUSBDiagnostic	BOOL	TRUE - There is some diagnostic in the USB (see detailed)
		FALSE – No diagnostic in the USB

Table 80: Summarized Diagnostics

Notes:

Hardware Failure: In case the Hardware Failure diagnostic is true, the controller must be sent to Altus Technical Assistance, as it has problems in the RTC or other hardware resources.

Software Exception: In case the software exception diagnostic is true, the user must verify his application to guarantee it is not accessing the memory wrongly. If the problem remains, the Altus Technical Support sector must be consulted. The software exception codes are described next in the controller's detailed diagnostics table.

Retentivity Error: The PLC writes data to retentive memory every 5 seconds at runtime. When this bit is TRUE, the most probable root cause is a hardware error on retentive memory. In this case, the CPU must be sent to Altus Technical Assistance. *Reset Cold* and *Reset Origin* commands triggered by MasterTool does not cause the indication of this diagnostic.

6.1.3.2. Detailed Diagnostics

The tables below contains Nexto Xpress controllers' detailed diagnostics. It is important to have in mind the observations below before consulting them:

- Visualization of the Diagnostics Structures: The Diagnostics Structures added to the Project can be seen at the item *Library manager* of MasterTool IEC XE tree view. There, it is possible to see all datatypes defined in the structure
- Counters: All controller diagnostics counters return to zero when their limit value is exceeded

DG_XP3xx.tDetailed.*		Type	Description
Target.*	dwCPUModel	ENUM (BYTE)	Controller model, ex: MODEL_XP325
	abyCPUVersion	BYTE ARRAY(4)	Firmware version
	abyBootloaderVersion	BYTE ARRAY(4)	Bootloader version
Hardware.*	bRTCFailure	BOOL	The main processor is unable to communicate with the RTC hardware.
	bIntegratedIoFailure	BOOL	The main processor is unable to communicate with the integrated I/O hardware.
Exception.*	wExceptionCode	WORD	Exception code generated by the RTS.
	byProcessorLoad	BYTE	Level, in percentage (%), of charge in the processor.
WebVisualization.*	byConnectedClients	BYTE	Clients number connected to the WebVisualization.
RetainInfo.*	byCPUInitStatus	BYTE	Controller startup status: 01: Hot start 02: Warm Start 03: Cold Start PS.: These variables are restarted in all startup.
	wCPUColdStartCounter	WORD	Counter of cold startups: It is increased when system starts up with bRetentivityError condition, and not due to the command of Reset Cold from MasterTool IEC XE (0 to 65535).
	wCPUWarmStartCounter	WORD	Counter of hot startups: It is increased during a normal start up, and not due the command of Reset Warm from MasterTool IEC XE (0 to 65535).
	wRTSResetCounter	WORD	Counter of reset performed by the RTS - Runtime System (0 to 65535).
	wWritesCounter	WORD	Counter of writes on retentive memory.
Reset.*	bBrownOutReset	BOOL	Last reset caused by failure in power supply.
	bWatchdogReset	BOOL	Last reset caused by internal watchdog error.
Serial. COM1.*	byProtocol	ENUM (BYTE)	Selected protocol in COM 1: NO_PROTOCOL (0): No protocol MODBUS_RTU_MASTER (1): MODBUS RTU Master MODBUS_RTU_SLAVE (2): MODBUS RTU Slave OTHER_PROTOCOL (3): Other protocol
	dwRXBytes	DWORD	Counter of characters received from COM 1 (0 to 4294967295).
	dwTXBytes	DWORD	Counter of characters transmitted from COM 1 (0 to 4294967295).

DG_XP3xx.tDetailed.*		Type	Description
	wRXPendingBytes	WORD	Number of characters left in the reading buffer in COM 1 (0 to 4095).
	wTXPendingBytes	WORD	Number of characters left in the transmission buffer in COM 1 (0 to 1023).
	wBreakErrorCounter	WORD	These counters are restarted in the following conditions: - Energizing - COM 1 serial port configuration - Removal of RX and TX queues PS.: When the controller is set Without Parity, the parity errors counter is not incremented in case it receives a different parity. In this case, an error of frame is indicated. The maximum value of each counter is 65535.
	wParityErrorCounter	WORD	
	wFrameErrorCounter	WORD	
	wRXOverrunCounter	WORD	
CAN.*	bBusAlarm	BOOL	The bus has a critical error and is shut-down.
	byBusState	ENUM (BYTE)	Informs the status of the device: UNKNOWN: impossible to get the network state. ERR_FREE: no occurrence of CAN bus errors. ACTIVE: only few CAN bus errors (below warning level). WARNING: occurrence of some CAN bus errors (above warning level). PASSIVE: too many CAN bus errors (above error level). BUSOFF: the node is shutdown (errors exceeded the admissible maximum).
	udiTxCounter	UDINT	Number of packets Tx changed in the PLC CAN bus.
	udiRxCounter	UDINT	Number of packets Rx changed in the PLC CAN bus.
	udiTxErrorCounter	UDINT	Number of packets Tx with errors in the PLC CAN bus.
	udiRxErrorCounter	UDINT	Number of packets Rx with errors in the PLC CAN bus.
	udiLostCounter	UDINT	Number of lost packets in the PLC CAN bus.
	byUSBDevice	ENUM (BYTE)	Type of the device connected to the USB port: NO_DEVICE UNKNOWN_DEVICE MASS_STORAGE_DEVICE SERIAL_CONVERTER_DEVICE MODEM_DEVICE WIFI_ADAPTER_DEVICE
	bOvercurrent	BOOL	The device connected on the USB port is draining more current than supported
	tMassStorage. byMountState	ENUM (BYTE)	Informs the status of the device: MOUNTED UNMOUNTED

DG_XP3xx.tDetailed.*	Type	Description
USB.*	tMassStorage. dwFreeSpaceKb	DWORD Informs the free space on the mass storage device.
	tMassStorage. dwTotalSizeKb	DWORD Informs the total size of the mass storage device.
	tSerialConverter. byProtocol	ENUM (BYTE) Selected protocol in COM 10: NO_PROTOCOL (0): No protocol
	tSerialConverter. dwRXBytes	DWORD Counter of characters received from COM 10 (0 to 4294967295).
	tSerialConverter. dwTXBytes	DWORD Counter of characters transmitted from COM 10 (0 to 4294967295).
	tSerialConverter. wRXPendingBytes	WORD Number of characters left in the reading buffer in COM 10 (0 to 4095).
	tSerialConverter. wTXPendingBytes	WORD Number of characters left in the transmission buffer in COM 10 (0 to 1023).
	tSerialConverter. wBreakErrorCounter	WORD These counters are restarted in the following conditions: - Energizing - COM 10 serial port configuration - Removal of RX and TX queues PS.: When the controller is set Without Parity, the parity errors counter is not incremented in case it receives a different parity. In this case, an error of frame is indicated. The maximum value of each counter is 65535.
	tSerialConverter. wParityErrorCounter	
	tSerialConverter. wFrameErrorCounter	
	tSerialConverter. wRXOverrunCounter	
	tModem. bConfigured	BOOL Indicates that the modem was configured in the Web page.
	tModem. byConnectionState	ENUM (BYTE) The modem connection state: DISCONNECTED (1): modem is not connected or configured. CONNECTING (2): device configured, trying to connect to the Internet. FAILED_RETRYING (3): connection failed, the modem will try to connect again. CONNECTED (4): modem connected and up, the IP is available in the szIP diagnostic.
	tModem. szIP	STRING String with IP address used by the modem.
	tWifiAdapter. bConfigured	BOOL Indicates that the WiFi adapter was configured in the Web page.
	tWifiAdapter. byConnectionState	ENUM (BYTE) The WiFi adapter connection state: DISCONNECTED (1): WiFi is not connected and/or configured. CONNECTING (2): device configured, trying to connect to WiFi network. FAILED_RETRYING (3): connection failed, the WiFi will try to connect again. This can occur due to wrong password or the network is not available.

DG_XP3xx.tDetailed.*		Type	Description
			CONNECTED (4): WiFi adapter connected to the network.
	tWifiAdapter.szIP	STRING	String with the IP used in the WiFi network.
	tWifiAdapter.szMask	STRING	String with the WiFi network mask.
	tWifiAdapter.szGateway	STRING	String with the WiFi network Gateway.
	tWifiAdapter.szMAC	STRING	String with WiFi adapter's MAC address (exclusive for the device).
Ethernet. NET1.*	bLinkDown	BOOL	Indicates the link state in the interface.
	wProtocol	WORD	Selected protocol in NET 1: 00: Without protocol
	wProtocol. bMODBUS_RTU_ETH_Client	BOOL	MODBUS RTU Client via TCP
	wProtocol. bMODBUS_ETH_Client	BOOL	MODBUS TCP Client
	wProtocol. bMODBUS_RTU_ETH_Server	BOOL	MODBUS RTU Server via TCP
	wProtocol. bMODBUS_ETH_Server	BOOL	MODBUS TCP Server
	szIP	STRING(15)	Port IP Address
	szMask	STRING(15)	Port Subnet Mask
	szGateway	STRING(15)	Port Gateway Address
	szMAC	STRING(15)	Port MAC Address
	abyIP	BYTE ARRAY(4)	Port IP Address
	abyMask	BYTE ARRAY(4)	Port Subnet Mask
	abyGateway	BYTE ARRAY(4)	Port Gateway Address
	abyMAC	BYTE ARRAY(6)	Port MAC Address
	dwPacketsSent	DWORD	Counter of packets sent through the interface (0 to 4294967295).
	dwPacketsReceived	DWORD	Counter of packets received through the interface (0 to 4294967295).
	dwBytesSent	DWORD	Counter of bytes sent through the interface (0 to 4294967295).
	dwBytesReceived	DWORD	Counter of bytes received through the port (0 to 4294967295).
	dwTXDropErrors	DWORD	Counter of connection losses in the transmission through the interface (0 to 4294967295).
	dwTXCollisionErrors	DWORD	Counter of collision errors in the transmission through the interface (0 to 4294967295).
	dwRXDropErrors	DWORD	Counter of connection losses in the reception through the interface (0 to 4294967295).
	dwRXFrameErrors	DWORD	Counter of frame errors in the reception through the interface (0 to 4294967295).

DG_XP3xx.tDetailed.*		Type	Description
UserFiles.*	byMounted	BYTE	Indicates if the memory used for recording the user files is able to receive data.
	dwFreeSpacekB	DWORD	Free memory space for user files (Kbytes).
	dwTotalSizekB	DWORD	Storage capacity of the memory of user files (Kbytes).
UserLogs.*	byMounted	BYTE	Status of memory in which the user logs are inserted.
	wFreeSpacekB	WORD	Free memory space of user logs (Kbytes)
	wTotalSizekB	WORD	Storage capacity of the memory of user logs (Kbytes).
Application.*	byCPUState	ENUM (BYTE)	<p>Informs the controller's CPU operation state:</p> <p>RUN (1): The application is in execution (Run Mode).</p> <p>STOP (3): The application is stopped (Stop Mode).</p>
	bForcedIOs	BOOL	Some memory area is being forced by the user through MasterTool.
Application Info.*	dwApplicationCRC	DWORD	32 bits CRC of Application. When the application is modified and sent to the controller, a new CRC is generated.
SNTP.*	bServiceEnabled	BOOL	SNTP Service enabled.
	byActiveTimeServer	ENUM (BYTE)	<p>Indicates which server is active:</p> <p>NO_TIME_SERVER (0): None active server.</p> <p>PRIMARY_TIME_SERVER (1): Active Primary Server.</p> <p>SECONDARY_TIME_SERVER (2): Active Secondary Server.</p>
	wPrimaryServerDownCount	WORD	Counter of times in which the primary server is unavailable (0 to 65535).
	wSecondaryServerDownCount	WORD	Counter of times in which the secondary server is unavailable (0 to 65535).
	dwRTCTimeUpdatedCount	DWORD	Counter of times the RTC was updated by the SNTP service (0 to 4294967295).
	byLastUpdateSuccessful	ENUM (BYTE)	<p>Last update status:</p> <p>NOT_UPDATED (0): Not updated.</p> <p>UPDATE_FAILED (1): Failure.</p> <p>UPDATE_SUCCESSFUL (2): Successful.</p>
	byLastUpdateTimeServer	ENUM (BYTE)	<p>Server used in the last update:</p> <p>NO_TIME_SERVER (0): None update.</p> <p>PRIMARY_TIME_SERVER (1): Primary Server.</p> <p>SECONDARY_TIME_SERVER (2): Secondary Server.</p>
	sLastUpdateTime	EXTENDED_DATE_AND_TIME	Date and time of the last sync time via SNTP.
	sLastUpdateTime. byDayOfMonth	BYTE	
	sLastUpdateTime. byMonth	BYTE	
	sLastUpdateTime. wYear	WORD	

DG_XP3xx.tDetailed.*		Type	Description
	sLastUpdateTime. byHours	BYTE	
	sLastUpdateTime. byMinutes	BYTE	
	sLastUpdateTime. bySeconds	BYTE	
	sLastUpdateTime. byMilliseconds	WORD	
IntegratedIO.*	AnalogInputs. tAnalogInput_xx. bInputNotEnable	BOOL	The input channel is not enabled on the configuration.
	AnalogInputs. tAnalogInput_xx. bOverRange	BOOL	The input signal level is above the maximum value defined for the selected input type.
	AnalogInputs. tAnalogInput_xx. bOpenLoop	BOOL	The input signal level is below minimum (only for 4-20mA mode).
	AnalogOutputs. tAnalogOutput_xx. bOutputNotEnable	BOOL	The output channel is not enabled on the configuration.
	AnalogOutputs. tAnalogOutput_xx. bOpenLoop	BOOL	The impedance of the load connected to the output channel is above the maximum accepted (only for current output mode).
	AnalogOutputs. tAnalogOutput_xx. bShortCircuit	BOOL	The impedance of the load connected to the output channel is below the minimum accepted (only for voltage output mode).
	RTDInputs. tRtdInput_xx. bInputNotEnable	BOOL	The input channel is not enabled on the configuration.
	RTDInputs. tRtdInput_xx. bOverRange	BOOL	The resistance is above the maximum value defined for the selected type.
	RTDInputs. tRtdInput_xx. bUnderRange	BOOL	The resistance is below the minimum value defined for the selected type (only for temperature sensors).

Table 81: Detailed Diagnostics Description

Notes:

Exception Code: The exception codes generated by the RTS (Runtime System) is presented below:

Code	Description	Code	Description
0x0000	There is no exception code.	0x0051	Access violation.
0x0010	Watchdog time of the expired IEC task (Software Watchdog).	0x0052	Privileged instruction.
0x0012	I/O configuration error.	0x0053	Page failure.
0x0013	Check-up errors after program download.	0x0054	Stack overflow.
0x0014	Fieldbus error.	0x0055	Invalid disposition.
0x0015	I/O updating error.	0x0056	Invalid maneuver.
0x0016	Cycle time (execution) exceeded.	0x0057	Protected page.

Code	Description	Code	Description
0x0017	Program online updating too long	0x0058	Double failure.
0x0018	Unsolved external references.	0x0059	Invalid OpCode.
0x0019	Download rejected.	0x0100	Data type misalignment.
0x001A	Project unloaded, as the retentive variables cannot be reallocated.	0x0101	Arrays limit exceeded.
0x001B	Project unloaded and deleted.	0x0102	Division by zero.
0x001C	Out of memory stack.	0x0103	Overflow.
0x001D	Corrupted retentive memory; cannot be mapped.	0x0104	Cannot be continued.
0x001E	Project can be loaded but it causes a break later on.	0x0105	Watchdog in the processor load of all IEC task detected.
0x0021	Target of startup application does not match to the current target.	0x0150	FPU: Not specified error.
0x0022	Scheduled tasks error... IEC task configuration failure. Application working with wrong target. Illegal instruction.	0x0151	FPU: Abnormal operand.
		0x0152	FPU: Division by zero.
0x0023	Downloaded file Check-up error.	0x0153	FPU: Inexact result.
0x0024	Mismatch between the retentive identity and the current boot project program identity	0x0154	FPU: Invalid operation.
0x0025	IEC task configuration failure.	0x0155	FPU: Overflow.
0x0026	Application is running with the wrong target.	0x0156	FPU: Stack verification.
0x0050	Illegal instruction.	0x0157	FPU: Underflow.

Table 82: Exception Codes

Brownout Reset: The brownout reset diagnostic is only true when the power supply exceeds the minimum limit required in its technical features, remaining in low voltage, without suffering any interruption. The controller identifies the voltage break and indicates the power supply failure diagnostic. When the voltage is reestablished, the controller is restarted automatically and indicates the brownout reset diagnostic.

Parity Error Counter: When the serial COM 1 is configured Without Parity, this error counter won't be incremented when it receives a message with a different parity. In this case, a frame error will be indicated.

User Partition: The user partition is a memory area reserved for the storage of data in the CPU. For example: files with PDF extension, files with DOC extension and other data.

RTD Inputs: the table below describes the behavior of over and under range diagnostics according to the input type selected:

Diagnostics	0 to 400 Ω Scale		0 to 4000 Ω Scale		Sensors of Platinum type (Pt) $\alpha = 0.00385$		Sensors of Platinum type (Pt) $\alpha = 0.003916$	
	Resist.	Count	Resist.	Count	Temp.	Count	Temp.	Count
Over range	>420 Ω (420 to 404.1 Ω)	4200 (4200 to 4041)	>4200 Ω (4200 to 4041 Ω)	>4200 (4200 to 4041)	>850 $^{\circ}\text{C}$	8500	>630 $^{\circ}\text{C}$	6300
No diagnostics	0 to 404 Ω	0 to 4040	0 to 4040 Ω	0 to 4040	-200 to 850 $^{\circ}\text{C}$	-2000 to 8500	-200 to 630 $^{\circ}\text{C}$	-2000 to 6300
Under range	-	-	-	-	<-200 $^{\circ}\text{C}$	-2000	<-200 $^{\circ}\text{C}$	-2000

Table 83: RTD Input Diagnostics

6.1.4. Diagnostics via Function Blocks

The Function Blocks for advanced diagnostics available for Nexto Xpress controllers are the same ones provided for Nexto Series CPUs, which are described on this same section of Nexto Series CPUs User Manual code MU214605.

6.2. Preventive Maintenance

- It must be verified, each year, if the interconnection cables are connected firmly, without dust accumulation, mainly the protection devices
- In environments subjected to excessive contamination, the equipment must be periodically cleaned from dust, debris, etc.
- The TVS diodes used for transient protection caused by atmospheric discharges must be periodically inspected, as they might be damaged or destroyed in case the absorbed energy is above limit. In many cases, the failure may not be visual. In critical applications, is recommendable the periodic replacement of the TVS diodes, even if they do not show visual signals of failure
- Connector block tightness and cleanness every six months