



Lexium 32

Programming example
(without PLCopen Library)
Profibus/Profinet with Siemens
S7-1200/1500



Profibus / Profinet





Table of Contents

1	Introduction.....	3
1.1	Overview	3
1.2	Before You Begin	3
1.3	Start-Up and Test	5
1.4	Operations and Adjustments	6
2	Programming example overview	7
2.1	Required files and LXM32 versions	7
3	Hardware Configuration.....	8
3.1	Profibus Hardware configuration	8
3.2	Profinet Hardware configuration	8
3.3	HW Identifier of DriveProfileLexium 1	10
3.3.1	Profibus:.....	10
3.3.2	Profinet:.....	11
4	LXM32 Function Blocks.....	12
4.1	Create Axis Reference Structure	12
4.2	LXM32_Control	13
4.3	LXM32_Homing.....	15
4.4	LXM32_Jog	18
4.5	LXM32_ProfilePosition	20
4.6	LXM32_ProfilePorque	21
4.7	LXM32_ProfileVelocity	23
4.8	LXM32_ElectronicGear	24
4.9	LXM32_Parameter_Read_Write	26
4.10	LXM32_Touchprobe	27
4.11	LXM32_RMAC	28
4.12	LXM32_Reboot	30



1 Introduction

1.1 Overview

This chapter gives the introduction.

Contents of this chapter

This chapter contains the following topics:

Topic	Page
Before you begin	03
Start-Up and Test	05
Operations and Adjustments	06

1.2 Before You Begin

General

The products specified in this document have been tested under actual service conditions. Of course, your specific application requirements may be different from those assumed for this and any related examples described herein. In that case, you will have to adapt the information provided in this and other related documents to your particular needs. To do so, you will need to consult the specific product documentation of the hardware and/or software components that you may add or substitute for any examples specified in this documentation. Pay particular attention and conform to any safety information, different electrical requirements and normative standards that would apply to your adaptation.

© 2014 Schneider Electric. All rights reserved

WARNING

REGULATORY INCOMPATIBILITY

Be sure that all equipment applied and systems designed comply with all applicable local, regional and national regulations and standards

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

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material. A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved. Failure to observe this information can result in injury or equipment damage.



Expert Support Machine Solution

The use and application of the information contained herein require expertise in the design and programming of automated control systems. Only the user or integrator can be aware of all the conditions and factors present during installation and setup, operation, and maintenance of the machine or process, and can therefore determine the automation and associated equipment and the related safeties and interlocks which can be effectively and properly used. When selecting automation and control equipment, and any other related equipment or software, for a particular application, the user or integrator must also consider any applicable local, regional or national standards and/or regulations.

Some of the major software functions and/or hardware components used in the proposed architectures and examples described in this document cannot be substituted without significantly compromising the performance of your application. Further, any such substitutions or alterations may completely invalidate any proposed architectures, descriptions, examples, instructions, wiring diagrams and/or compatibilities between the various hardware components and software functions specified herein and in related documentation. You must be aware of the consequences of any modifications, additions or substitutions.

A residual risk, as defined by EN/ISO 12100-1, Article 5, will remain if

- it is necessary to modify the recommended logic and if the added or modified components are not properly integrated in the control circuit.
- you do not follow the required standards applicable to the operation of the machine, or if the adjustments to and the maintenance of the machine are not properly made (it is essential to strictly follow the prescribed machine maintenance schedule).
- the devices connected to any safety outputs do not have mechanically-linked contacts.

CAUTION

EQUIPMENT INCOMPATIBILITY

Read and thoroughly understand all device and software documentation before attempting any component substitutions or other changes related to the application examples provided in the document

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



1.3 Start-Up and Test

Before using electrical control and automation equipment after design and installation, the application and associated functional safety system must be subjected to a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such testing be made and that enough time is allowed to perform complete and satisfactory testing.

CAUTION

EQUIPMENT OPERATION HAZARD

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

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

Verify that the completed system, including the functional safety system, is free from all short circuits and grounds, except those grounds installed according to local regulations. If high-potential voltage testing is necessary, follow the recommendations in equipment documentation to help prevent injury or equipment damage.



1.4 Operations and Adjustments

General

Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly installed and operated.

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

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

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

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the examples and implementations suggested herein. It is sometimes possible to adjust the equipment incorrectly and this produce unsatisfactory or unsafe operation. Always use the manufacturer instructions as a guide to functional adjustments. Personnel who have access to these adjustments must be familiar with the equipment manufacturer instructions and the machinery used with the electrical equipment. Only those operational adjustments actually required by the machine operator should be accessible to the operator. Access to other controls should be restricted to help prevent unauthorized changes in operating characteristics.



2 Programming example overview

This programming example can be used as an alternative solution to the PLCopen library.

The programming example contains function blocks to control a Lexium32M Profibus/Profinet.

2.1 Required files and LXM32 versions

LXM32M hardware revision RS03 and DOM (date of manufacturing) 2014 or newer

LXM32M Firmware PR 912.00 V1.22.00 or newer

LXM32M Profinet Module Software PR915.30 V1.00.11 or newer

LXM32M Profinet GSD file: GSDML-V2.31-Schneider-Lexium32-20151228.xml or newer

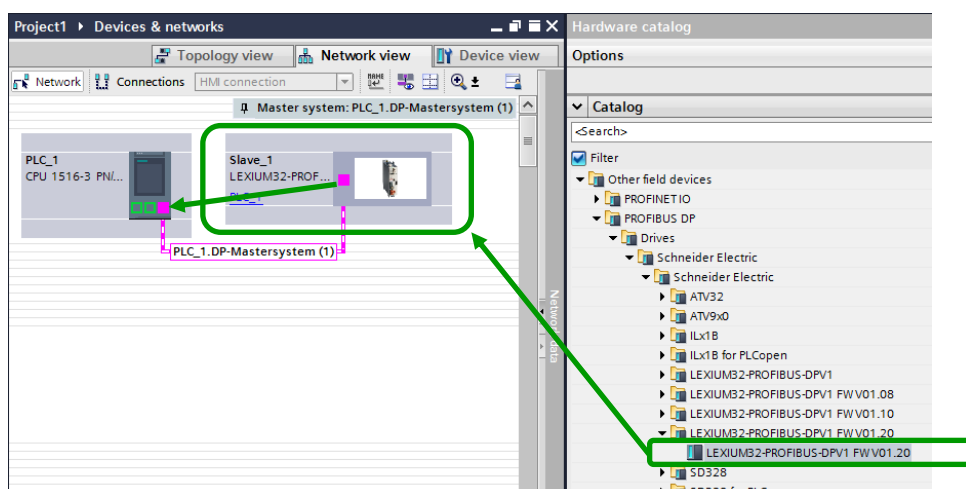
LXM32M ProfibusGSD file "SE120B9D.GSD" or newer



3 Hardware Configuration

3.1 Profibus Hardware configuration

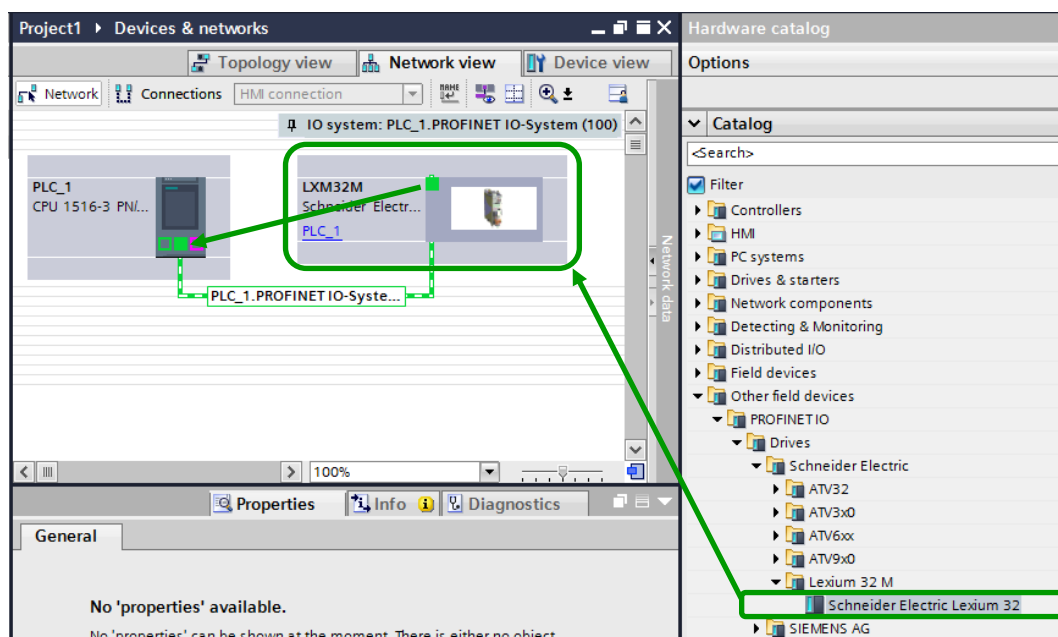
1. Insert the LXM32 Profibus Device to the Network view
2. Link the LXM32 PB interface to the PLC Master Interface



You have now linked the drive into the network as a Profibus Slave.

3.2 Profinet Hardware configuration

1. Insert the LXM32 Profinet Device to the Network view
2. Link the LXM32 Profinet interface to the PLC Master Interface



You have now linked the drive into the network as a Profinet IO-Device.



Setup PROFINET station name and IP address (Properties of LXM32 Device)

The screenshot displays the Schneider Electric software interface for configuring an LXM32 device. The top navigation pane shows the project structure: Project1 > PLC_1 [CPU 1516-3 PN/DP] > Distributed I/O > PROFINET IO-System (100): PN/IE_1 > lxm32-axis-01.

The main window is divided into two sections. The top section, titled "Device overview", shows a table of modules:

Module	Rack	Slot
lxm32-axis-01	0	0
X1	0	0 X1
Drive Profile Lexium 1_1	0	1
	0	2
	0	3
	0	4
	0	5
	0	6
	0	7
	0	8
	0	9

The bottom section, titled "lxm32-axis-01 [Module]", shows the configuration options for the selected module. The "General" tab is active, and the "PROFINET interface [X1]" is selected. The "Name" field is highlighted with a green box and contains the value "lxm32-axis-01".

The "Ethernet addresses" tab is also shown, with the "Interface networked with" section displaying "Subnet: PN/IE_1". The "IP protocol" section is checked, and the "Set IP address in the project" option is selected, with the "IP address" field highlighted by a green box and containing the value "192 . 168 . 0 . 2".



3.3 HW Identifier of DriveProfileLexium 1

The PLCopen open library does only use the IO data structure of DriveProfileLexium1. DriveProfileLexium1 is implemented in the default configuration in Profibus and Profinet.

3.3.1 Profibus:

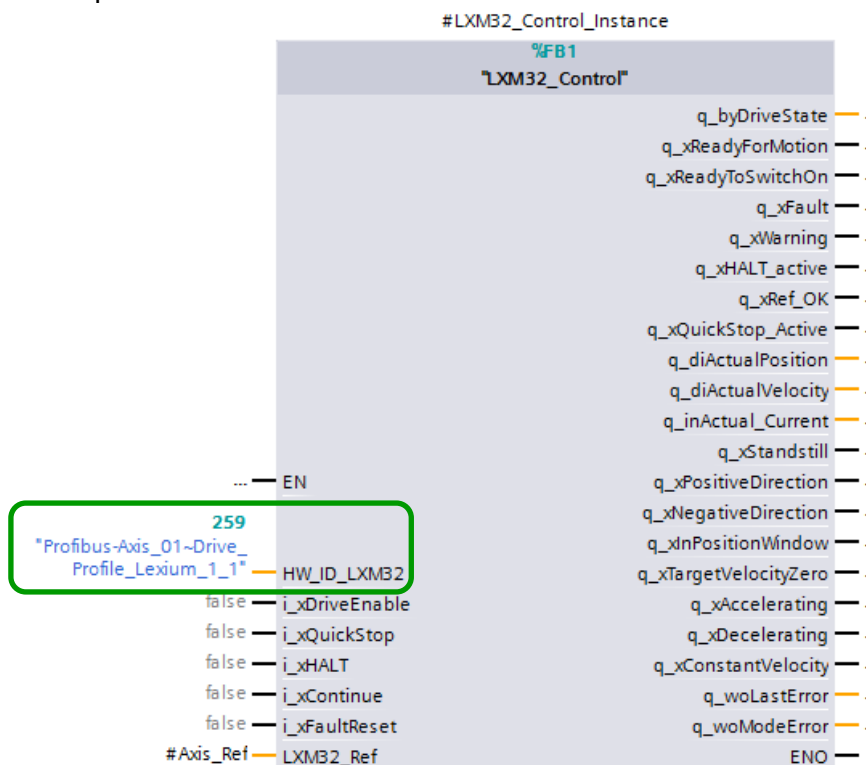
The screenshot shows the 'Device overview' table with the following data:

Module	Rack	Slot	I address	Q address	Type
Profibus-Axis_01	0	0			LEXIUM32-PROFIBUS-DPV1 F...
Drive Profile Lexium 1_1	0	Processdata-Interface	0...25	0...25	Drive Profile Lexium 1
Empty module_1	0	Optional IO Module			Empty module
Empty module_2	0	Optional IO Module			Empty module
Empty module_3	0	Optional IO Module			Empty module
Empty module_4	0	Optional IO Module			Empty module
Empty module_5	0	Optional IO Module			Empty module
Empty module_6	0	Optional IO Module			Empty module
Empty module_7	0	Optional IO Module			Empty module
Empty module_8	0	Optional IO Module			Empty module

The 'Properties' window for 'Drive Profile Lexium 1_1 [Module]' is shown with the 'Hardware identifier' tab selected. The 'Hardware identifier' field is set to 259.

Note: The Hardware identifier is used in the application to address the IO data of the DriveProfileLexium1. The Hardware identifier must be declared at the input "HW_ID_LXM32" of the function block LXM32_Control.

Example:





3.3.2 Profinet:

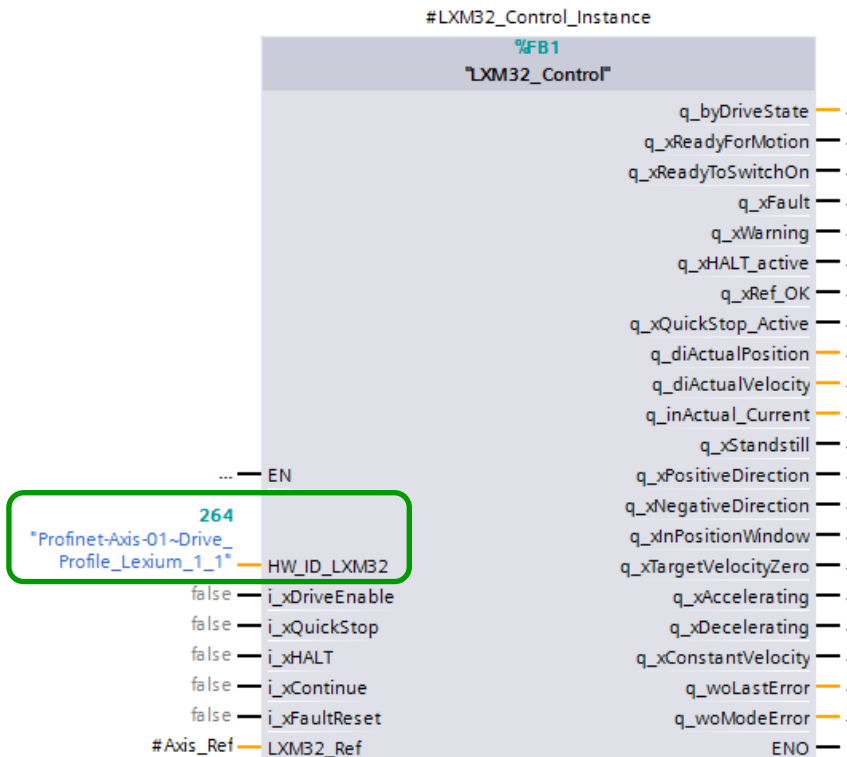
The screenshot shows the 'Device overview' table with the following data:

Module	Rack	Slot	I address	Q address	Type
Profinet-Axis-01	0	0			Schneider Electric Lexium 32
X1	0	0 X1			LXM32M
Drive Profile Lexium 1_1	0	1	256...281	256...281	Drive Profile Lexium
		2			
		3			
		4			
		5			
		6			
		7			

Below the table, the 'Drive Profile Lexium 1_1 [Module]' section is expanded, showing the 'Hardware identifier' field with the value '264'.

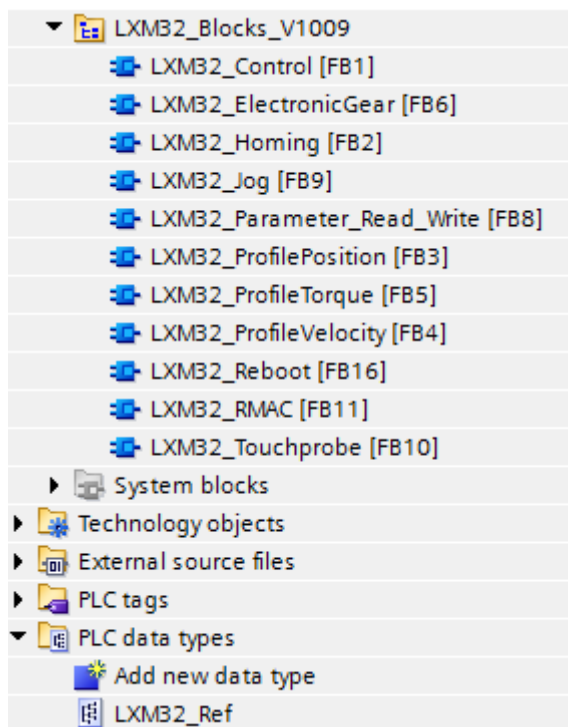
Note: The Hardware identifier is used in the application to address the IO data of the DrifeProfileLexium1.
The Hardware identifier must be declared at the input “HW_ID_LXM32” of the function block LXM32_Control.

Example:





4 LXM32 Function Blocks



4.1 Create Axis Reference Structure

Each LXM32 needs an own axis reference structure of the PLC data type “[LXM32_Ref](#)”. This structure can be defined in a global data block or in the static variables of a function block.

In this example you can see how to declare the axis reference structure in the stat variables of a function block.

FB_LXM32_Motion_Blocks								
	Name	Data type	Default value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	▼ Input				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	■ LXM32_HW_ID	HW_IO	16#0	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	▼ Output				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4	■ <Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5	▼ InOut				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
6	■ <Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
7	▼ Static				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
8	▶ Axis_Ref	"LXM32_Ref"		Non-ret...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	▶ LXM32_Control	"LXM32_Control"			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

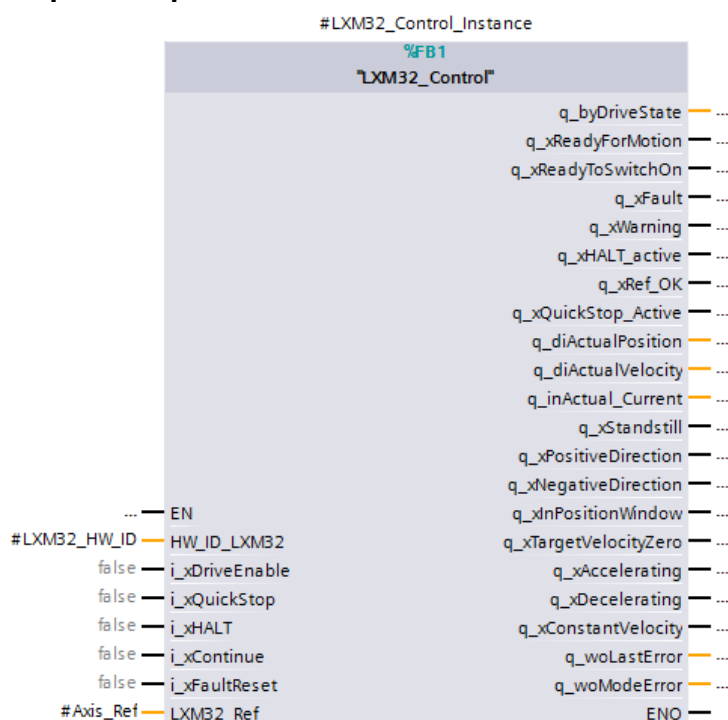


4.2 LXM32_Control

Function description

- Read / write of LXM32 Input / Output data. (FB must be called cyclic)
- Control the LXM32 drive state (Enable power stage, fault reset, QuickStop, HALT)
- Monitor LXM32 status (drive state, actual position, actual velocity, actual motor current, drive error ID, mode error ID, motion status)

Graphical representation



Input parameter description:

Parameter	Data type	Description
HW_ID_LXM32	HW_IO	Hardware ID of LXM32 – Drive Profile Lexium 1 (see HW configuration)
i_xDriveEnable	BOOL	Enable power stage
i_xQuickStop	BOOL	Stop movement with QuickStop
i_xHALT	BOOL	Stop movement with HALT
i_xContinue	BOOL	Continue movement after HALT
i_xFaultReset	BOOL	Error reset

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure



Output parameter description:

Parameter	Data type	Description
q_byDriveState	Byte	Drive status 3 = disabled 4 = ready to switch on 6 = operation enable (ready for motion) 7 = QuickStop active 9 = fault active
q_xReadyToSwitchOn	BOOL	Power stage can be enabled (drive state = 4)
q_xReadyForMotion	BOOL	Power stage is enabled -> motion command can be executed (drive state = 6)
q_xFault	BOOL	Fault is active
q_xWarning	BOOL	Warning is active (for detailed information see LXM32 Object _LastWarning 7186)
q_xHALT_Active	BOOL	HALT is active
q_xRef_OK	BOOL	Reference position is valid
q_xQuickStop_Active	BOOL	Quick stop is active
q_diActualPosition	DINT	Actual motor position in USR units
q_diActualVelocity	DINT	Actual velocity in USR units
q_inActual_Current	INT	Actual motor current (100 = 1,00 Arms)
q_xStandstill	BOOL	Drive is in state standstill
q_xPositiveDirection	BOOL	Drive is moving in positive direction
q_xNegativeDirection	BOOL	Drive is moving in negative direction
q_xInPositionWindow	BOOL	Motor is in position window
q_xTargetVelocityZero	BOOL	Target velocity is zero
q_xAccelerating	BOOL	Profile generator accelerates
q_xDecelerating	BOOL	Profile generator decelerates
q_xConstantVelocity	BOOL	Profile generator moves at constant velocity
q_woLastError	BOOL	Error causing a stop (error classes 1 to 4) Number of the current error. Any consecutive errors do not overwrite this error number. Example: If a limit switch error reaction caused an overvoltage error, this parameter would contain the number of the limit switch error.
q_xwoModeError	BOOL	Error code for synchronous errors Usually, this is an error that was caused by the activation of an operating mode. The ModeError bit relates to MT-dependent parameters.



4.3 LXM32_Homing

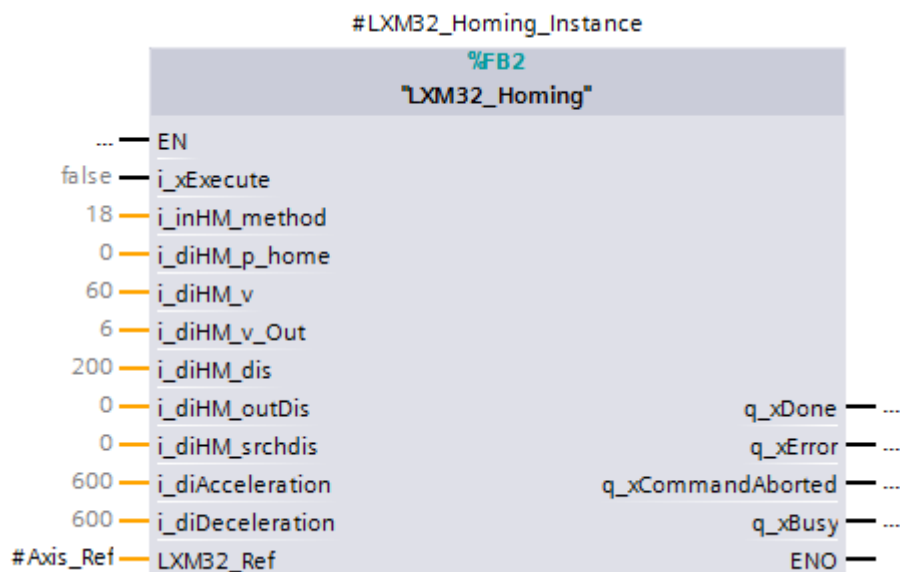
Function description

The operating mode Homing is used to define a reference point. The reference point establishes an absolute position reference between the motor position and a defined axis position. The reference point can be defined by means of a reference movement or by means of position setting.

- Reference movement: Movement to a limit switch, a reference switch or the index pulse of the motor encoder. When the position is reached, a position reference is automatically created. This position becomes the absolute user-defined position.
- Position setting: (homing method 35). The current motor position is set to a desired position value. The zero point is defined by the position value. Position setting is only possible when the motor is at a standstill.

The operating mode Homing must be completed without an error for the new reference point to be valid.

Graphical representation





Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the homing
i_inHM_method	INT	<p>LIMN with index pulse 1 : LIMP with index pulse 2:</p> <p>7 = REF+ with index pulse, beyond REF, in direction of LIMN 8 = REF+ with index pulse, within REF, in direction of LIMN 9 = REF+ with index pulse, within REF, in direction of LIMP 10 = REF+ with index pulse, beyond REF, in direction of LIMP 11 = REF- with index pulse, beyond REF, in direction of LIMN 12 = REF- with index pulse, within REF, in direction of LIMN 13 = REF- with index pulse, within REF, in direction of LIMP 14 = REF- with index pulse, beyond REF, in direction of LIMP 17 = LIMN 18 = LIMP 23 = REF+, beyond REF, in direction of LIMN 24 = REF+, within REF, in direction of LIMN 25 = REF+, within REF, in direction of LIMP 26 = REF+, beyond REF, in direction of LIMP 27 = REF-, beyond REF, in direction of LIMN 28 = REF-, within REF, in direction of LIMN 29 = REF-, within REF, in direction of LIMP 30 = REF-, beyond REF, in direction of LIMP 33 = on index pulse, in direction of LIMN 34 = on index pulse, in direction of LIMP 35 = Dimension setting without movement</p>
i_diHM_p_home	DINT	Position is set as current motor position after successful reference movement [usr]. Value range: depends on scaling factor, initial value: 0.
I_diHM_v	DINT	Speed for searching the limit or reference switch [user units]. Drive stops when switching edge has been detected. Value range: 1...2147483647; Initial value: 60.
i_diHM_v_Out	DINT	Speed for clearance movement back to the switching edge [user units]. The max. travel distance when searching for the switching edge can be restricted with the parameter <i>POutHome</i> . Value range: 1...2147483647; Initial value: 6.
i_diHM_dis	DINT	Distance between the switching edge and the reference point [usr]. At end of movement, the drive moves back towards switching edge until the distance has been reached. Value range: 1..2147483647, initial value: 200.
i_diHM_outDis	DINT	0: Clearing monitor switched off. >0: Run-off [usr], i.e. max. travel distance when searching for the switching edge. If the switching edge is not found in this distance, the reference movement is interrupted with an error. Value range: 0..2147483647, initial value: 0.
i_diHM_srchDis	DINT	Maximum search distance after overtravel of switch 0: Search distance monitoring disabled >0: Search distance The switch must be activated again within this search distance, otherwise the reference movement is canceled. Changed settings become active the next time the motor moves.
i_diAcceleration	DINT	Value for the acceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.
i_diDeceleration	DINT	Value for the deceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.



Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xDone	BOOL	Homing is finished without error
q_xError	BOOL	Homing is finished with error
q_xCommandAborted	BOOL	Homing aborted
q_xBusy	BOOL	Homing is active



4.4 LXM32_Jog

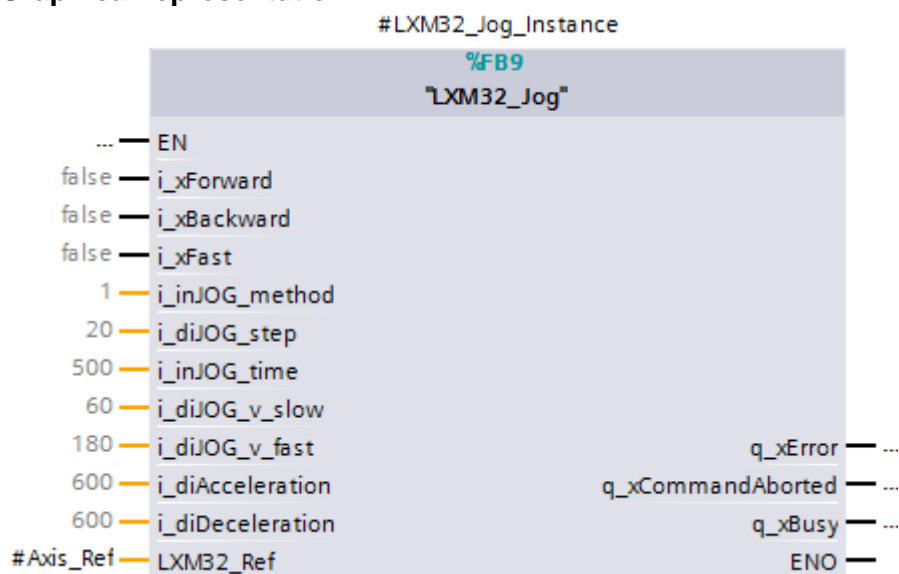
Function description

In the operating mode Jog, a movement is made from the actual motor position in the desired direction. The velocity can be set. As long as the signal for the direction is available, a continuous movement is made in the desired direction.

If movements in positive and negative directions are requested at the same time, there is no motor movement.

The function block starts the operating mode Jog. TRUE at the input Forward or the input Backward starts the jog movement.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xForward	BOOL	JOG movement in positive direction
i_xBackward	BOOL	JOG movement in negative direction
i_xFast	BOOL	The velocity can be changed during the movement. FALSE: Movement at the velocity set in i_diJOG_v_slow. TRUE: Movement at the velocity set in i_diJOG_v_fast.
i_inJOG_method	INT	Selection of jog method 0 / Jog with continuous movement 1 / Jog with step movement
i_diJOG_Step	DINT	Distance for step movement
I_diJOG_time	DINT	Wait time for step movement
i_diJOG_v_slow	DINT	Velocity for slow movement
i_diJOG_v_fast	DINT	Velocity for fast movement
i_diAcceleration	DINT	Value for the acceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.
i_diDeceleration	DINT	Value for the deceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.



Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xError	BOOL	JOG movement is finished with error
q_xCommandAborted	BOOL	JOG movement aborted
q_xBusy	BOOL	JOG movement is active



4.5 LXM32_ProfilePosition

Function description

The following settings can be made in the operating mode Profile Position:

- Target position
- Type of movement (relative movement or absolute movement)
- Target velocity
- Acceleration and deceleration ramps

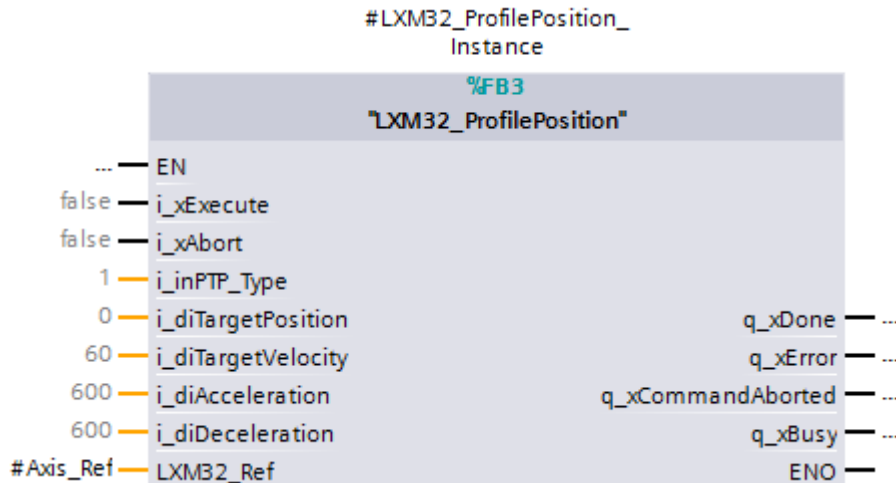
The movement to the target position is made on the basis of a motion profile. The motion profile is calculated by the profile generator in the drive. The calculation is performed on the basis of the actual position and the target position, the actual velocity and the target velocity and the acceleration and deceleration ramps.

In the operating mode Profile Position, absolute movements, relative movements and additive movements are possible.

- Absolute movement with reference to the zero point
- Relative movement with reference to the actual position
- Additive movement with reference to the previous target position

A zero point must be defined with the operating mode Homing prior to the first absolute movement.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the movement
i_xAbort	BOOL	Stops (aborts) an active movement
i_inPTP_Type	INT	Type of movement 1 = Absolute movement 2 = Additive movement 3 = Relative movement
i_diTargetPosition	DINT	Target Position for absolute movement or target distance for additive/relative movement In USR units



i_diTargetVelocity	DINT	Target velocity in USR units
i_diAcceleration	DINT	Value for the acceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.
i_diDeceleration	DINT	Value for the deceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xDone	BOOL	Movement is finished without error
q_xError	BOOL	Movement is finished with error
q_xCommandAborted	BOOL	Movement aborted
q_xBusy	BOOL	Movement is active

Note:

New target position can be started with a new rising edge on “i_xExecute” during an active movement.

Changing target velocity value during an active movement takes effect immediately. (Only in mode absolute movement)

4.6 LXM32_ProfileTorque

WARNING

EXCESSIVELY HIGH VELOCITY DUE TO INCORRECT LIMIT VALUE

Without a proper limit value, the motor can reach a very high velocity in this operating mode.

- Check the parameterized velocity limitation.

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

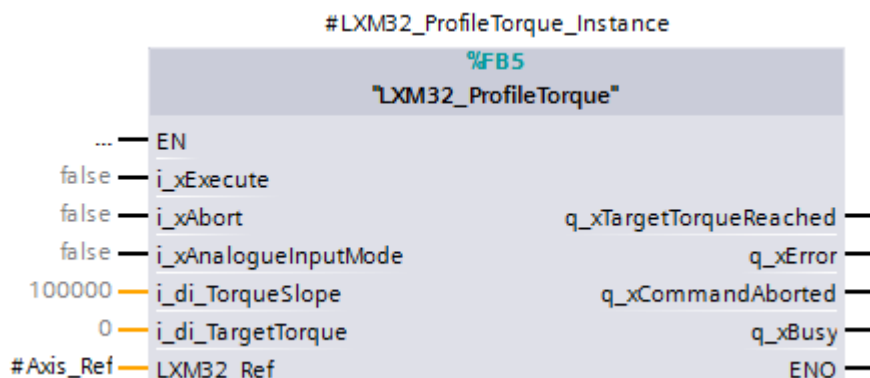
You can set a target torque in the operating mode Profile Torque. The movement is made with this target torque in the operating mode Profile Torque.

Function description

The function block starts the operating mode Profile Torque. In the operating mode Profile Torque, a movement is made with a desired target torque. The reference value for the target torque is supplied via the input i_diTargetTorque. When the target torque is reached, the output q_xTargetTorqueReached is set. The input i_diTorqueSlope lets you set the slope of the motion profile for the torque.



Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the movement
i_xAbort	BOOL	Stops (aborts) an active movement
i_xAnalogueInputMode	BOOL	FALSE: Target Torque is set by input value "i_diTargetTorque" TRUE: Target Value is set by analogue input of the extended IO Module
i_di_TorqueSlope	DINT	Value range: Initial value: 100000 The value corresponds to 0.1% per second of the nominal torque of the motor. Example: If i_di_TorqueSlope = 1000, then 100% of the nominal torque of the motor is reached in one second. Use the parameter _M_M_O to get the nominal torque of the motor.
i_diTargetTorque	DINT	Value range: -30000 ... 30000 Initial value: 0 Target torque The value corresponds to 0.1% of the nominal torque of the motor. Example: Torque = 300 corresponds to 30% of the nominal torque of the motor. Use the parameter _M_M_O to get the nominal torque of the motor.

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xTargetTorqueReached	BOOL	FALSE: Target torque is not yet reached. TRUE: Target torque reached.
q_xError	BOOL	Movement is finished with error
q_xCommandAborted	BOOL	Movement aborted
q_xBusy	BOOL	Movement is active

Note:

Changing target torque value during an active movement takes effect immediately.



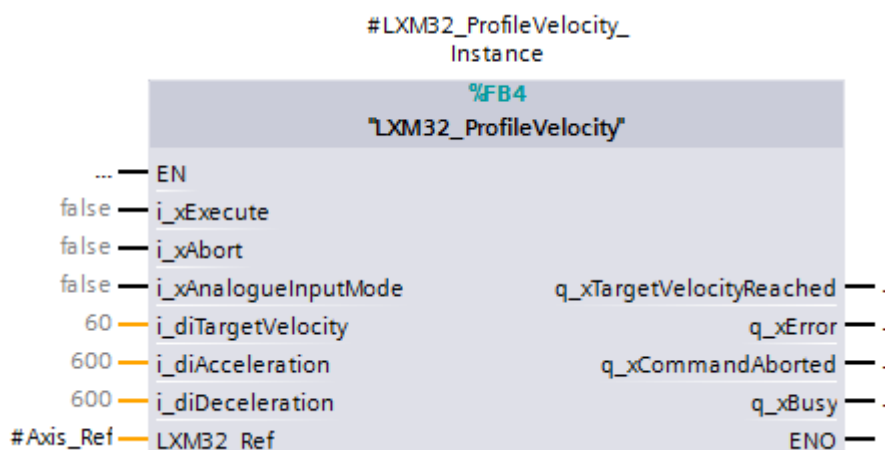
4.7 LXM32_ProfileVelocity

Function description

You can set a target velocity in the operating mode Profile Velocity. The movement is performed with this target velocity in the operating mode Profile Velocity. The movement continues until a new target velocity is set or until the operating mode is aborted.

Transitions between two target velocities are performed on the basis of a motion profile. The motion profile is determined by the profile generator in the drive on the basis of the actual velocity, the target velocity and the acceleration and deceleration ramps.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the movement
i_xAbort	BOOL	Stops (aborts) an active movement
i_xAnalogueInputMode	BOOL	FALSE: Target Velocity is set by input value "i_diTargetVelocity" TRUE: Target Value is set by analogue input of the extended IO Module
i_diTargetVelocity	DINT	Target velocity in USR units
i_diAcceleration	DINT	Value for the acceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.
i_diDeceleration	DINT	Value for the deceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure



Output parameter description:

Parameter	Data type	Description
q_xTargetVelocityReached	BOOL	Target velocity is reached
q_xError	BOOL	Movement is finished with error
q_xCommandAborted	BOOL	Movement aborted
q_xBusy	BOOL	Movement is active

Note:

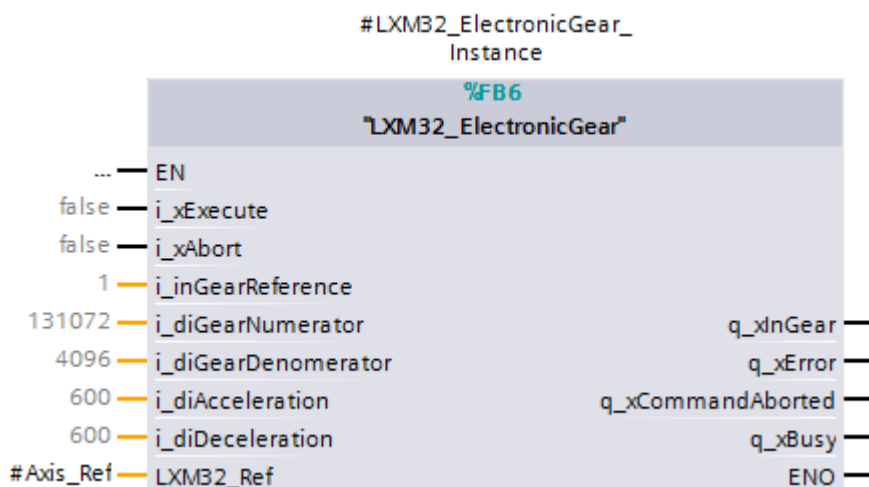
Changing target velocity value during an active movement takes effect immediately.

4.8 LXM32_ElectronicGear

Function description

In the operating mode Electronic Gear, movements are carried out according to externally supplied reference value signals. A new position reference value is calculated on the basis of these reference value signals plus an adjustable gear ratio.

Graphical representation





Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the operating mode
i_xAbort	BOOL	Stops (aborts) an active movement / operating mode
i_inGearReference	INT	1: Position synchronization immediate 2: Position synchronization compensated 3: Velocity synchronization
i_diGearNumerator	DINT	Gear ratio numerator
i_diGearDenominator	DINT	Gear ratio denominator
i_diAcceleration	DINT	Only used in VELOCITY SYNCHRONIZATION Value for the acceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.
i_diDeceleration	DINT	Only used in VELOCITY SYNCHRONIZATION Value for the deceleration ramp gradient [user units] Value range: 1...2147483647; Initial value: 600.

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xInGear	BOOL	Gear ratio is reached
q_xError	BOOL	Movement is finished with error
q_xCommandAborted	BOOL	Movement aborted
q_xBusy	BOOL	Movement is active

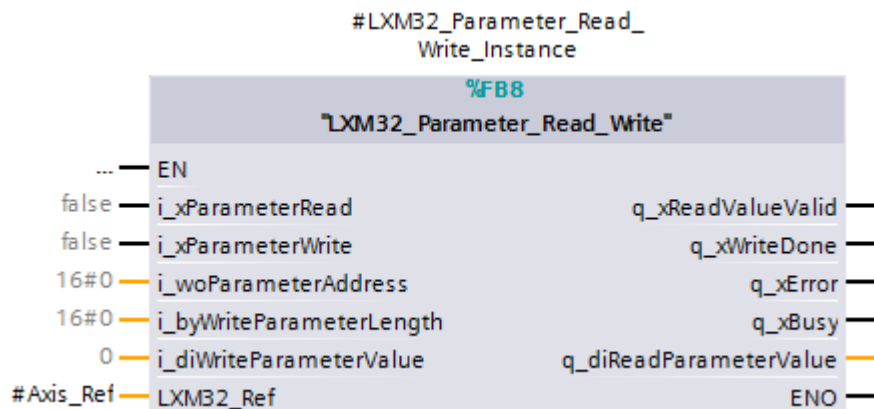


4.9 LXM32_Parameter_Read_Write

Function description

The following functions block allows to read or write drive parameters. See the product manual for a description of the parameters.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xParameterRead	BOOL	Reading a parameter
i_xParameterWrite	BOOL	Writing a parameter
i_woParameterAddress	WORD	Parameter address (see LXM32 manual)
i_byWriteParameterLength	BYTE	Size in bytes of the parameter which will be written.
i_diWriteParameterValue	DINT	Write parameter value

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xReadValueValid	BOOL	When reading a parameter a new value is available on the output "q_diReadParameterValue"
q_xWriteDone	BOOL	Parameter value is written successfully
q_xError	BOOL	FB is finished with error
q_xBusy	BOOL	FB is active
q_diReadParameterValue	DINT	Parameter read value



4.10 LXM32_Touchprobe

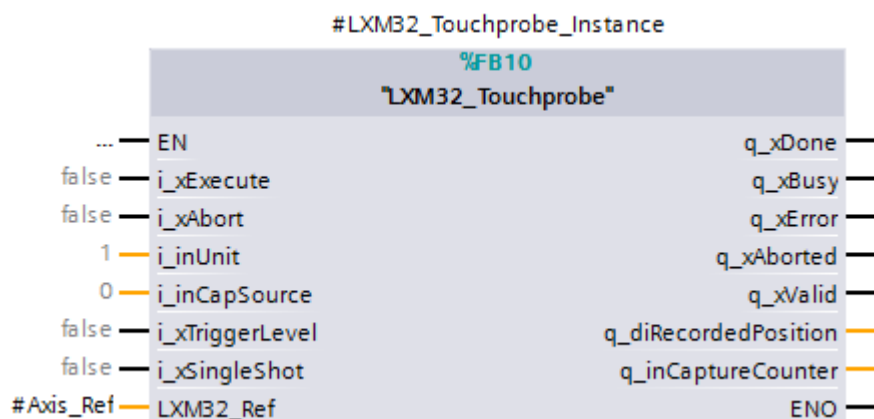
Function description

Position capture via a signal input captures the current position at the point in time at which an edge is detected at one of the digital Capture inputs.

Settings:

- Position capture can be triggered by a rising edge or a falling edge at the signal input.
- It is possible to use one-time or continuous position capture.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the function block
i_xAbort	BOOL	Aborts the active function block
i_inUnit	INT	Select capture input CAP1, CAP2 or CAP3
i_inCapSource	BYTE	0=Encoder1 / 1=Encoder2
i_xTriggerLevel	BOOL	FALSE: Start position capture at falling edge. TRUE: Start position capture at rising edge.
i_xSingleShot	BOOL	FALSE: Set continuous position capture. Continuous capture means that the motor position is captured at every edge. The previously captured value is lost. TRUE: Sets one-time position capture. One-time capture means that the position is captured at the first edge. The capture value is not overwritten by a new edge.

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure



Output parameter description:

Parameter	Data type	Description
q_xDone	BOOL	Function block is finished and "SingleShot" Trigger position is available
q_xBusy	BOOL	FB is active
q_xError	BOOL	FB is finished with error
q_xAborted	BOOL	FB is aborted
q_xValid	BOOL	Continuous Trigger Mode: A new trigger position is available. This output is only set for one PLC cycle.
q_diRecordedPosition	DINT	Recorded position value
q_inCaptureCounter	DINT	Continuous Trigger Mode: Number of detected trigger signals

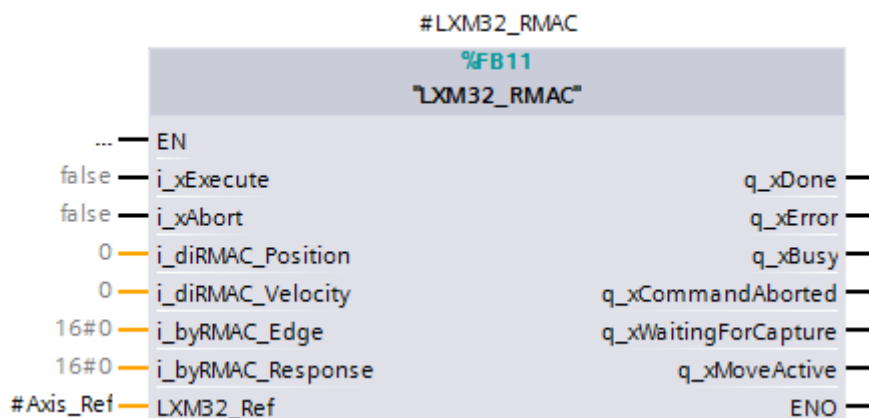
4.11 LXM32_RMAC

RMAC (Relative movement after capture – See LXM32 manual)

Function description

Relative Movement After Capture (RMAC) starts a relative movement via a signal input while another movement is running. The target position and the velocity can be parameterized. The signal input function "Start Signal Of RMAC" is required to start the relative movement.

Graphical representation





Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the function block
i_xAbort	BOOL	Aborts the active function block 1st case: Deactivates RMAC function 2nd case: Stops an active RMAC movement if capture already occurred
i_diRMAC_Position	DINT	Target distance of relative movement after capture
i_diRMAC_Velocity	DINT	Velocity of relative movement after capture Value 0: Use of current motor velocity Value >0: Value is the target velocity
i_byRMAC_Edge	BYTE	Edge of capture signal for relative movement after capture 0 / Falling edge: Falling edge 1 / Rising edge: Rising edge
i_byRMAC_Response	BYTE	Depending on the set velocity, target position and deceleration ramp, the target position may be over travelled. Response if target position is over travelled 0 / Error Class 1: Error class 1 1 / No Movement To Target Position: No movement to target position 2 / Movement To Target Position: Movement to target position

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xDone	BOOL	Function block is finished and Relative movement after capture terminated
q_xError	BOOL	FB is finished with error
q_xBusy	BOOL	FB is active
q_xCommandAborted	BOOL	FB is aborted
q_xWaitingForCapture	BOOL	Function is active and waiting for the trigger signal
q_xMoveActive	BOOL	Trigger signal occurred and relative movement is active

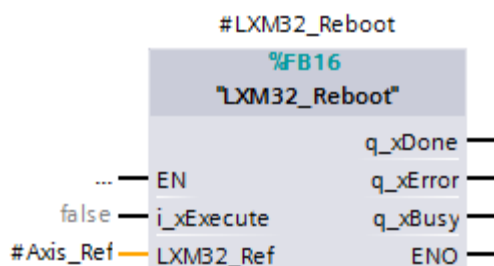


4.12 LXM32_Reboot

Function description

This function can be used to restart the LXM32 device.
This is necessary after changing BOOT UP Parameters.
E.g. Digital Input Configuration.

Graphical representation



Input parameter description:

Parameter	Data type	Description
i_xExecute	BOOL	Rising edge starts the function block

Input/Output parameter description:

Parameter	Data type	Description
LXM32_Ref	LXM32_Ref	Axis reference structure

Output parameter description:

Parameter	Data type	Description
q_xDone	BOOL	LXM32 Restart is finished and fieldbus communication is established again.
q_xError	BOOL	FB is finished with error
q_xBusy	BOOL	FB is active

Note:

The communication will be lost until the LXM32 has booted up again.