Powerlink PLCopen Library for XENAX®

Version 3.0.6

Edition 7. June 2023

XENAX® Ethernet Servo Controller with POWERLINK® fieldbus module

Functional safety, TÜV certified

Force processes for “Force Control“

This manual describes the integration of the XENAX® Xvi75V8 Servo Controller into a B&R PLC with the Automation Studio.
Therefore the Jenny Science PLCopen library “JsMcLib” based on the CANopen Motion Profile DS402 will be used.

This documantation includes examples with configurations, program integration and testing.

General

Contents

1 Development environment 5
1.1 Controller, Tools, Libraries 5
1.2 XENAX® Servo Controller 6
1.2.1 State LEDs on the Powerlink fieldbus module. 6
1.2.2 Update Firmware & WebMotion 6
1.2.3 WebMotion® 7
1.2.4 Powerlink connection from B&R PLC to XENAX® 8
Industrial Hub for more than 4 XENAX® 8
1.3 LINAX® Linear Motor Axis 8
1.4 ELAX® Linear Motor Slide 8
1.5 ROTAX® Rotary Motor Axes 9

2 PLCopen Library (JsMcLib) 10
2.1 Drive Modes: point to point or interpolated 10
2.2 State Diagram 11
2.2.1 Profile Position Mode 12
2.2.2 Cyclic Synchronized Mode 12
2.3 Function blocks 14
2.3.1 INIT 14
2.3.2 CyclicIn 14
2.3.3 CyclicOut 14
2.3.4 Power 15
2.3.5 Reference 15
2.3.6 MoveAbsolte 15
2.3.7 MoveRelative 16
2.3.8 JogVelocity 16
2.3.9 MoveCyclicPosition 16
2.3.10 Halt 16
2.3.11 Stop 17
2.3.12 Reset 17
2.3.13 ReadStatus 17
2.3.14 ReadPSR 17
2.3.15 ForceCalibration 18
2.3.16 ReadDigitalInput 18
2.3.17 WriteDigitalOutput 18
2.3.18 ReadActualPosition 18
2.3.19 ReadActualCurrent 18
2.3.20 SetPDO 19
2.3.21 WriteParameter 19
2.3.22 ReadParameter 19

2.3.23 ReadAxisError 20
2.4 Minimum and Maximum Values of Function Blocks 21
2.5 Error numbers 21
2.6 Error sources 24
2.7 Error type 24
3 Automation Studio 25
3.1 New Project 25
3.2 Import of the JsMcLib library 28
3.2.1 JsMcLib User Help 29
3.3 Embed XENAX® CANopen object file (.XDD) 29
3.4 Embed XEANAX® Servo Controller in project 30
3.4.1 Configuration XENAX® based on the xdd description 32
3.4.2 Cyclic transmission 32
3.4.3 View I/O Mapping 33
3.5 Embedding XENAX® Error Messages 34
4 Program example "Profile Position Mode" 36
4.1 Get program example and save it 36
4.2 Import program example to the project 37
4.3 Configure Ethernet interface 38
4.4 Move Program example into Task 40
4.5 I/O Mapping, connect .xdd channels to program variables 41
4.6 Start Program Example 42
5 Program example Cyclic Synchronous Position Mode 43
5.1 Integrate second XENAX® Servo Controller in project. 43
5.2 Insert Virtual Axis AS V4.0-V4.6 44
5.2.1 Add Virtual Axis 44
5.2.2 Disable Virtual Axis 46
5.2.3 Configure Virtual Axis 47
5.2.4 Insert the NC-Mapping Table 48
5.3 Insert virtual Axis in AS >V4.7 50
5.3.1 Add Virtual Axis 50
5.3.2 NC Object Settings 53
5.4 Parameter Adjustments 54
5.4.1 Axis Parameter 54
5.4.2 Cycle time 56
5.5 Insert Program Example 56
5.6 Configure Ethernet Interface 57
5.7 Move Program Example to Task with Desired Cycle Time 58
5.8 I/O Mapping, connect the .xdd Channels with the program variables. 59
5.9 Start Program Example 60
6 Program example Forceteq® 61

7 Upgrade from XENAX® 75V8 to 75V8S 63

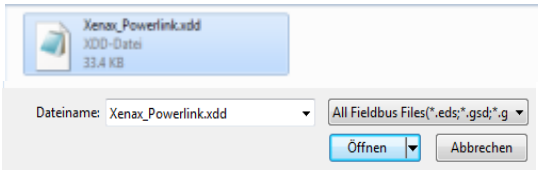
1 Development environment

1.1 Controller, Tools, Libraries

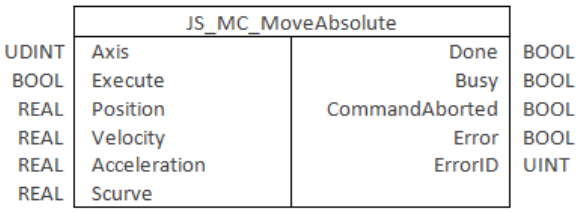
Automation Studio
Software for the configuration of the B&R PLC with project handling.

Software for configuration and programming of the B&R PLC, including project management. This manual is based on Automation Studio V4

Electronic data sheet
XDD-file (XML Device Description) is a standard XML-format according to ISO 15745-4. This file includes communication parameter and objects of the XENAX® Servo Controller. These are necessary for the integration into Automation Studo V4.



JsMcLib
(Jenny Science Motion Control Library)
This PLCopen library is based on the motion profile DS 402 including the corresponding "state-machine".
Using the Jenny Science library will simplify the "motion"-programming.



B&R PLC with Powerlink
For example the X20-System, controls the cyclic data communication and defines the clock pulse for the "cyclic synchronous position mode" which is used for axis interpolation.

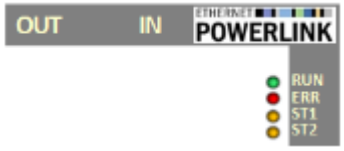
1.2 XENAX® Servo Controller

XENAX® Ethernet Servo Controller
With additional POWERLINK fieldbus module.

To use the JsMcLib v2.10.1 or higher the XENAX® Servo Controller firmware v3.64D or higher and the POWERLINK fieldbus module firmware v2.0 or higher are required.

1.2.1 State LEDs on the Powerlink fieldbus module.

These are the states of the LED on the Powerlink fieldbus module.



LED state	**RUN** (STAT)	**ERR**	**ST1** Status 1	**ST2** Status 2
<OFF>	In init process or no power	Bus module no error		Bus module l ready
<ON>	Operational state	State bus off	No application in the flash	
<Blink>	Pre-Operational state			Protocol download in progress

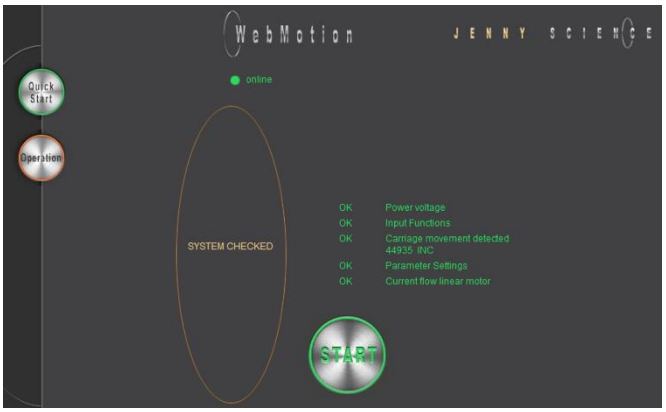
1.2.2 Update Firmware & WebMotion

The XENAX® firmware and WebMotion can be updated with the help of the Ethernet Installer. Ethernet Installer, Firmware and WebMotion can be downloaded from www.jennyscience.ch.

1.2.3 WebMotion®

WebMotion®

This is the graphical user interface from Jenny Science. It is stored in the embedded Web server of the XENAX® Servo Controller.
WebMotion ® is launched with a web browser by entering the TCP/IP address of XENAX® Servo Controller.
LINAX® / ELAX® linear motor axes are automatically recognized. The corresponding controller parameters are saved and loaded automatically.
With the Quick Start button, the linear motor axis can operate immediately. No user manual is needed.



The parametrization of the XENAX® Servo Controller is made over an Ethernet TCP/IP connection.

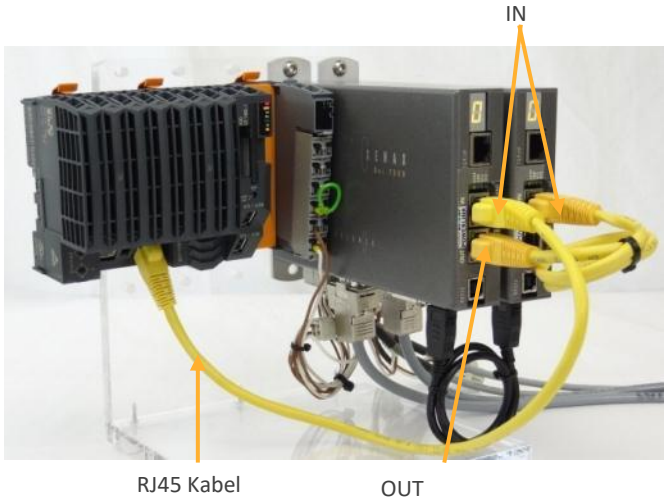
You can find the pre-set IP-address on the backside of your XENAX® Servo Controller.

Web address:
http://192.168.2.1xx/XENAX.html
To change the IP/ Web address you can use the Lantronix DeviceInstaller.

For additional information of the TCP/IP connection you could use the "XENAX_Xvi75V8_Manual_EN.pdf"or the video Tutorial on YouTube http://www.jennyscience.de/video-tutorials.

1.2.4 Powerlink connection from B&R PLC to XENAX®

Typically the Powerlink fieldbus is controlled with a linear structure from device to device. Shielded RJ45 cables go from the IN to the OUT to connect the devices with each other.

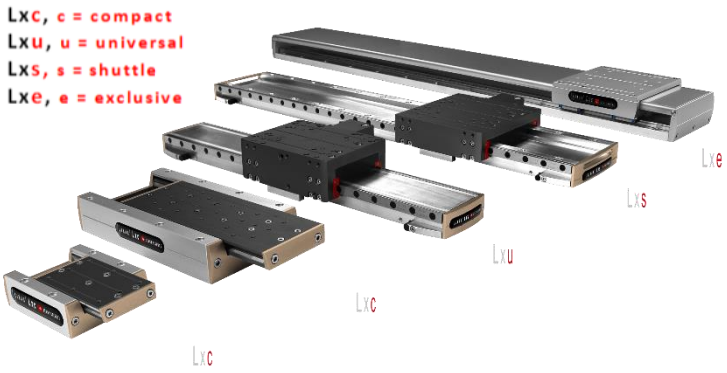


Industrial Hub for more than 4 XENAX®
A maximum of 4 XENAX® Servo Controllers can be connected in a row.
If there are more than 4 Servo Controller used, an industrial hub from B&R will be necessary.
Each contact can be used for 4 Servo Controller.

For example: With the HUB Nr. 0AC808.9-1 from B&R.
A maximum of 7*4 = 28 controllers could be connected.

1.3 LINAX® Linear Motor Axis

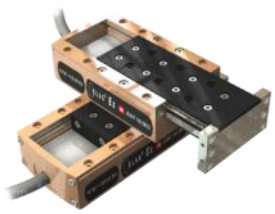
LINAX® Linear Motor-Axis
Are available in different lengths and types. The LINAX® linear motor axes are highly modular and can be combined flexibly amongst each other.



1.4 ELAX® Linear Motor Slide

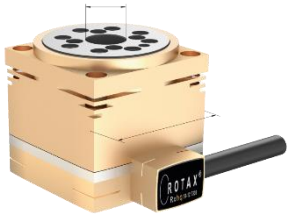
ELAX® Linear Motor Slide

The ELAX® Linear Motor Slide are predestined for fast, precise positioning tasks.
It's a Modular system with strokes of 30-150mm.
The variable one-cable connection can be mounted on the back or sidewise.



1.5 ROTAX® Rotary Motor Axes

ROTAX® Rotary motor axes
Specifically designed for fast and precise assembly and handling tasks. It can be equipped with standard gripping tools which enables a 360° rotation and has a hollow shaft feedthrough for vacuum or compressed air.



Rxhq = high torque

Rxvp = vacuum pressure

2 PLCopen Library (JsMcLib)

Jenny Science provides a PLCopen library for Automation Studio. The PLCopen standard is easy to understand and includes basic movement functions as well as Jenny Science specific features.

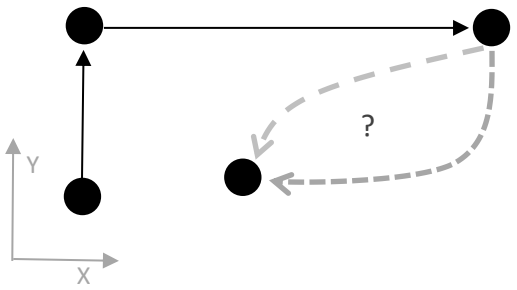
2.1 Drive Modes: point to point or interpolated

The library support two fundamental different drive modes.

1. **Point to point = Profile Position Mode:**
The parameters distance, speed, acceleration and s-curve are fixed before a drive. The trajectory (driving curve) is calculated on the Xenax®. This driving mode is simpler to implement in a B&R PLC, but gives less control over the driving curve to the PLC. It is not possible drive a straight line with a XY-Axis since both Axis can be started at the same time but will reach their target at different times. It is also not possible to drive a round curve because only the target position can be specified and not the path to the target location.

This mode fits a small PLC with low performance. There is no need for a virtual axis.

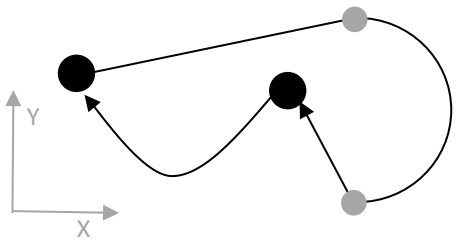
XY-Axis Profile Position



Limited control over driving path between two target positions with different X and Y coordinates. Furthermore, the speed and target position can not be changed during a drive. An Axis has to stop at every target position.

2. **Interpolated = Cyclic Synchronous Position Mode:**
In the cyclic synchrouns position mode, the target position is passed to the XENAX® servo controller at cyclic time intervals (for example every millisecond). The trajectory (driving curve) is calculated on the B&R PLC. For this reason, a virtual Axis for each Axis is needed. This enables full control over the driving curve. Thanks to the virtual Axis, round curves or other complex driving paths are now possible.

XY-Axis Cyclic Synchronous Position



Full control over Axis movement. Two grey circles show a change in direction ab speed without a stop.

2.2 State Diagram

The following diagram shows the state and the behaviour of the axis when multiple motion control function blocks are “simultaneously” active.

Each motion command is a transition that changes the state of the axis and, as a consequence, influences the method of calculation of the current movement.

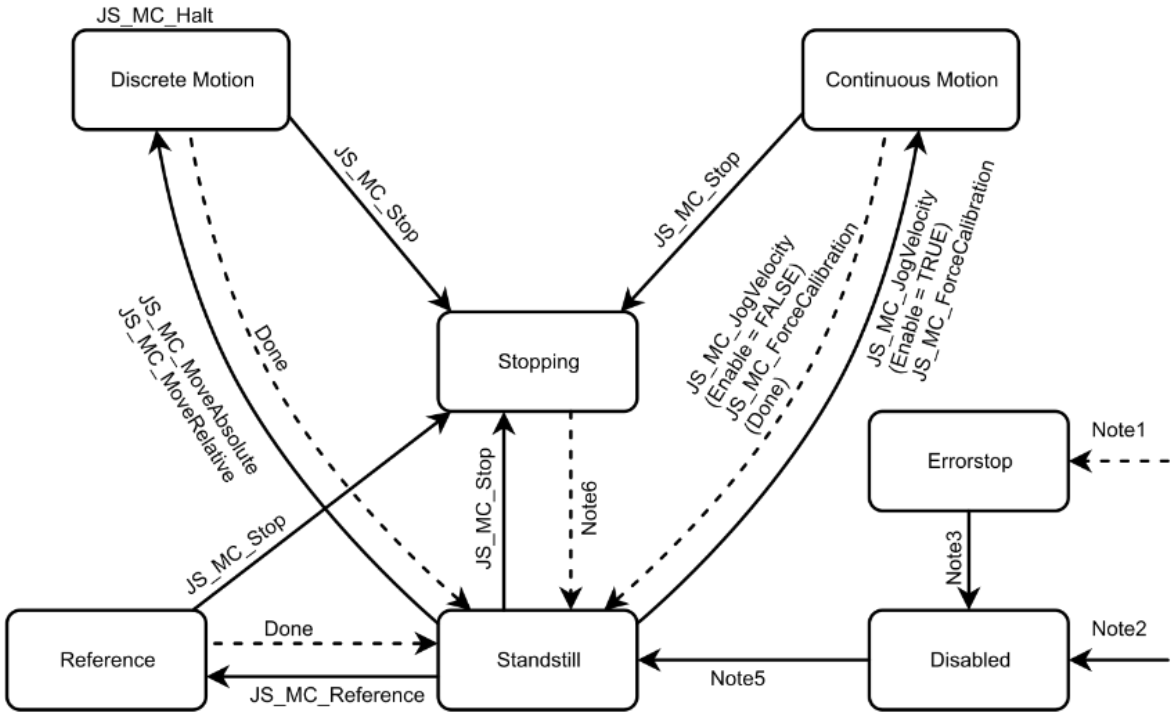
All function blocks which do not appear in the state diagram, do not affect the state of the axis.

The current state of the axis can be determined with the function block **“JS_MC_ReadStatus”**. If a function block is called where it is not allowed, the function block reports an error.

The notes describe the necessary conditions that must be met for a change in an axis state.

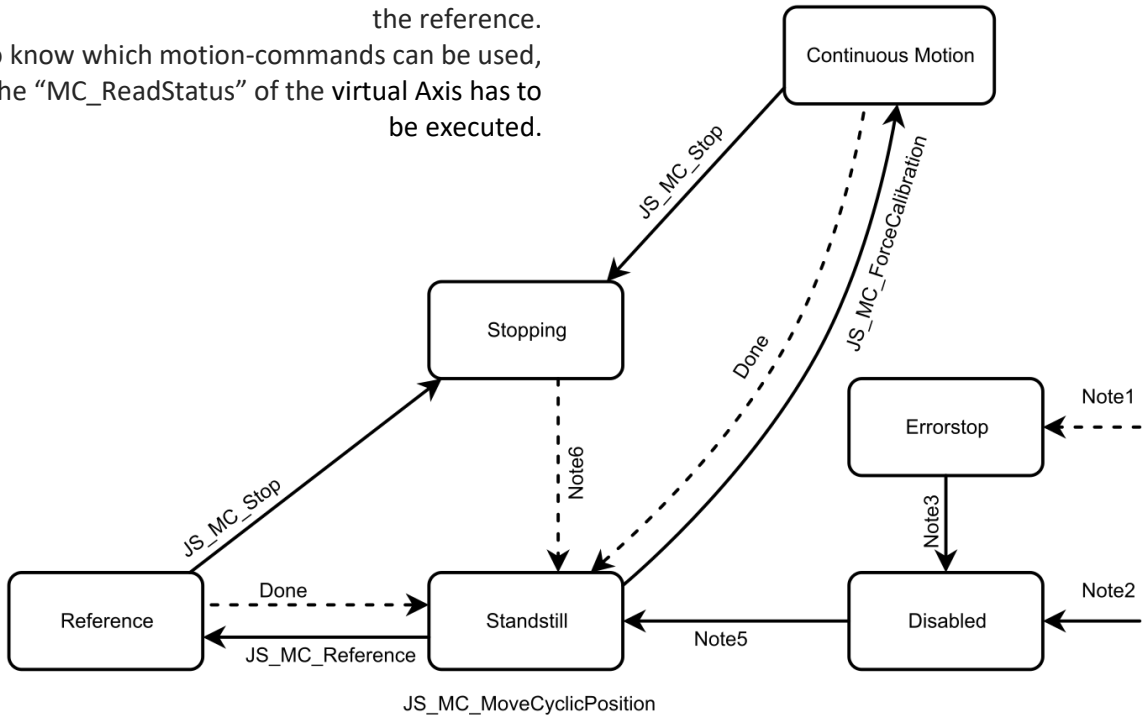
Important:
In the states “**Stopping**”, “**ErrorStop**”, ”**Disabled**” and “**Reference**” no motion blocks can be called. In standstill condition, an axis must always be referenced before starting a movement.

2.2.1 Profile Position Mode



2.2.2 Cyclic Synchronized Mode

In the Cyclic Synchronized mode the JS_MC_ReadStatus will be in „Standstill" after the reference.
To know which motion-commands can be used, the "MC_ReadStatus" of the virtual Axis has to be executed.



Note 1:
From any state. An error in the axis occurred.
Note 2:
From any state. JS_MC_Power.Enable = FALSE
and there is no error in the axis.
Note 3:
JS_MC_Reset AND JS_MC_Power.Status = FALSE.
Note 5:
JS_MC_Power.Enable = TRUE AND
JS_MC_Power.Status = TRUE
Note 6:
JS_MC_Stop.Done = TRUE AND
JS_MC_Stop.Execute = FALSE

2.3 Function blocks

The functionality of the JsMcLib is implemented in various small function blocks. In this subchapter, all those function blocks are described. Deprograms in the subsequent chapters will show the function blocks in action.

IF B&R has a similar function block, the JsMcLib function block is implemented based on the original block.

2.3.1 INIT

This function block must be called once at start up. It provides a reference to the Axis which is needed in all other JsMcLib function blocks. The function block must be called before any other JsMCLib block is called.

JS_MC_INIT	
Input	Output
AdrIoMap pDevice Node OperationMode	Axis

2.3.2 CyclicIn

Has to be called at the start of the periodically called program. This block reads data from the Powerlink bus.

JS_MC_CyclicIn	
Input	Output
Enable Axis	Valid Error ErrorID

2.3.3 CyclicOut

Has to be called as the last JsMcLib block in the periodically called program. This block writes values to the Powerlink bus.
Important: All other JsMcLib blocks must be called between CyclicIn and CyclicOut.

JS_MC_CyclicOut	
Input	Output
Axis	

2.3.4 Power

The enable input of the power blocks switches the power stage on and off. The power stage is turned on when the Status and Valid output is set.

JS_MC_Power	
Input	Output
Enable	Status
Axis	Valid
	Error
	ErrorID

2.3.5 Reference

With linear motors, a reference drive must be executed before any other movement can be performed. During a reference drive, the motor moves in one direction. The direction can be specified with the RefrenceMode input.

Rotary motors can be referenced, but they do not need to be referenced. However, some functions of the XENAX® servo controller require a referenced motor. Motors with ABZ encoders can be referenced with a Z-Mark in the Motor. ZMarkSpeedRot defines the speed during such a reference drive and the ReferenceMode defines the direction. All rotary motors can be optionally referenced with a limit switch. The speed during a limit switch reference drive is defined by the input ReferenceSpeedRot.

JS_MC_Reference	
Input	Output
Execute	Done
ReferenceMode	Busy
ZMarkSpeedRot	CommandAborted
ReferenceSpeedRot	Error
Axis	ErrorID

2.3.6 MoveAbsolte

Drives to an absolute position. The drive is start with a positive edge at the execute input and is finished when done equals one.
(Only for profile position mode. In cyclic synchronous position mode use MC_MoveAbsolute from B&R)

JS_MC_MoveAbsolute	
Input	Output
Axis	Done
Execute	Busy
Position	CommandAborted
Velocity	Error
Acceleration	ErrorID
Scurve	

2.3.7 MoveRelative

Drives a defined relative distance. The drive is started with a positive edge at the execute input and is finished when done equals one. (Only for profile position mode. In cyclic synchronous position mode use MC_MoveRelative from B&R)

JS_MC_MoveRelative	
Input	Output
Execute	Done
Distance	Busy
Velocity	CommandAborted
Acceleration	Error
Scurve	ErrorID
Axis	

2.3.8 JogVelocity

Drives with a constant speed in positive or negative direction. (Only for profile position mode. In cyclic synchronous position mode use MC_JogVelocity from B&R)

JS_MC_JogVelocity	
Input	Output
Enable	Active
Velocity	Busy
Acceleration	CommandAborted
Deceleration	Error
JogPositive	ErrorID
JogNegative	Jogging
Axis	

2.3.9 MoveCyclicPosition

Connects the XENAX® servo controller to a B&R virtual Axis. Drive commands are executed on the virtual Axis and not in the JsMcLib. (only for cyclic synchronous position mode)

JS_MC_MoveCyclicPosition	
Input	Output
Position	Valid
Enable	CommandAborted
Axis	Error
	ErrorID

2.3.10 Halt

Stops the current motion and switches to Stanstill state.

JS_MC_Halt	
Input	Output
Execute	Done
Deceleration	Busy
Axis	CommandAborted
	Error
	ErrorID

2.3.11 Stop

Stops the current motion an switches to a Stopping state. No movement command can be executed in this state. The JsMCLib remains in a Stopping state until the execute input is reset. While Halt only aborts the current move command, the Stop block also aborts all future move commands until the execute input is reset.

JS_MC_Stop	
Input	Output
Execute Deceleration Axis	Done Busy CommandAborted Error ErrorID

2.3.12 Reset

Resets the XENAX® servo controller. This is needed when the servo controller is in the Error state.

JS_MC_Reset	
Input	Output
Execute Axis	Done Busy Error ErrorID

2.3.13 ReadStatus

Reads the current state of the XENAX® servo controller.

JS_MC_ReadStatus	
Input	Output
Axis Enable	Valid Error ErrorID Errorstop Disabled Stopping Standstill DiscreteMotion Reference ContinuousMotion

2.3.14 ReadPSR

Reads the Process Status Register (PSR). This registers contains various information about the XENAX® servo controller. The PSR shoes for example if the servo controller is referenced.

JS_MC_ReadPSR	
Input	Output
Axis Enable	Valid Error ErrorID ProcessStatusRegister

2.3.15 ForceCalibration

Starts a Force calibration. The Axis moves from start- to endposition and measures cogging force and friction. All those Forces are then compensated in future drives.
If the motor oscillates during the Force Calibration, set IterativeFcDisable. This will clear old calibration data before a new calibration is started.

JS_MC_ForceCalibration	
Input	Output
Execute	Done
StartPosition	Busy
EndPosition	CommandAborted
IterativeFcDisable	Error
Axis	ErrorID

2.3.16 ReadDigitalInput

Reads digital inputs which are located in the XENAX® socket.

JS_MC_ReadDigitalInput	
Input	Output
Enable	Valid
Axis	Error
	ErrorID
	DigitalInput

2.3.17 WriteDigitalOutput

Writes digital ouputs which are located in the XENAX® socket.

JS_MC_WriteDigitalOutput	
Input	Output
DigitalOuput	Done
Execute	Error
Axis	ErrorID

2.3.18 ReadActualPosition

Reads the position of the XENAX® servo controller in increments.

JS_MC_ReadActualPosition	
Input	Output
Enable	Position
Axis	Valid
	Error
	ErrorID

2.3.19 ReadActualCurrent

Reads the motor current in mA.

JS_MC_ReadActualPosition	
Input	Output
Enable	Position
Axis	Error
	ErrorID
	Valid

2.3.20 SetPDO

This block is used to limit the motor current (LimitIForce). The index input must be set to 5 and the Value must be set to $x * 10mA$. A value of 7 would set the maximal current to 70mA.

JS_MC_SetPDO	
Input	Output
Axis	Done
Execute	Busy
Index	Error
Value	ErrorID

2.3.21 WriteParameter

Use this block to write CANopen parameters. The available CANopen parameters are described in the Document Xvi_CANopen which can be downloaded from our Website under XENAX® servo controller -> Manual Bus Module -> CANOPEN.zip. Most of the parameters from WebMotion® can be changed in a PLC program this way.

DataObject selects the parameter group and is a number between 1000h and 8000h. A parameter in a parameter group can be selected with SubID. Note that all direct commands have a SubID of 0. Data Length is the number of bytes of the value. The new Value of the parameter must be set too.

JS_MC_WriteParameter	
Input	Output
Execute	Done
DataObject	Busy
SubID	Error
DataLength	ErrorID
Value	
Axis	

2.3.22 ReadParameter

The CANopen parameters can also be read. Addressing a parameter work the same way as with WriteParameter. A value can be read as soon as the Done output is set.

JS_MC_ReadParameter	
Input	Output
Axis	Value
Enable	Done
DataObject	Busy
SubID	Error
DataLength	ErrorID

2.3.23 ReadAxisError

JS_MC_ReadAxisError	
Input	Output
Enable	Valid
Acknowledge	Busy
DataAddress	Error
DataLength	ErrorID
DataObjectName	ErrorRecordAvailable
Axis	ErrorRecord
	FunctionBlockErrorCount
	AxisErrorCount
	AxisWarningCount

All errors in an Axis or in a JsMcLib function are collected by the ReadAxisError block. Each collected error must be acknowledged before further drive commands are possible.

In case of an Error, the ErrorRecordAvailable gets set. An Error Code and the source block of the error can be found in the ErrorRecord. A list of error codes can be found at the end of this chapter.
In addition to that, it is possible to get an error message as a string. A string with a minimal length of 50 must be instantiated. The address and the actual length of the string must be written to the DataAddress and DataLength input. DataObjectName sould be set to 'JsMcEtxEn'.

ReadAxisError collects 3 different types of messages: axis errors, axis warnings and JsMcLib errors. The number of each type is counted by AxisErrorCount, AxisWarningCount and FunctionBlockErrorCount. Every message must be acknowledged with a positive edge at the Acknowledge input. Note that axis errors must be cleared first with the RESET block. In case of an JsMcLib error, the enable or execute input of the block in which the error occurred must be reset before it can be acknowledged. The error type and the error source can be found in the ErrorRecord output.

2.4 Minimum and Maximum Values of Function Blocks

Following minimum and maximum values of the function blocks should be adhered to.

name	datatype	min	max
Velocity linear	UDINT	10 inc/s	9000000 inc/s
Velocity rotative	UDINT	10 inc/s	100000000 inc/s
Deceleration	UDINT	2000 inc/s^2	1000000000 inc/s^2
Acceleration	UDINT	2000 inc/s^2	1000000000 inc/s^2
S-curve	UDINT	1 %	100 %

2.5 Error numbers

The following ErrorIDs can be generated by the JsMcLib function blocks. Lower numbers than 5000 are Axis Error generated by the XENAX® servo controller. Please look up those errors in the XENAX® Manual.

Value	Name	Description	Correction
0	ERR_OK	FUB executed correctly with no errors	None.
50000	jsmcERR_NIL_POINTER	No axis passed to FB	Ensure function block call only with correct axis passed.
50001	jsmcERR_DRIVE_NOT_READY	controller is not ready to switch on	Check controller for errors
50002	jsmcERR_DRIVE_SWITCHED_OFF	controller is switched off	Don't call function block when controller is switched off
50004	jsmcERR_REFERENCE_WRONG_METHOD	Reference method is not correct for the motor	Check documentation for allowed reference methods for the motor
50006	jsmcERR_ACCE_TO_SMALL	Acceleration is to small	Use larger acceleration (>=2000 inc/s^2)
50008	jsmcERR_SCURVE_NOT_IN_RANGE	Scurve is not in allowed range	Use Scurve in allowed range (1...100%)
50010	jsmcERR_SDO_COMM_FAILURE	Failure during SDO communication	Check power link connection to the Servo Controller
50011	jsmcERR_POWER_UP_FAILURE	Failure during power up sequence	Check Servo Controller for correct power supply
50012	jsmcERR_POWER_LOST	Power was turned off outside of JS_MC_Power control	Check and quit errors from other function blocks or axis, which caused the power off
50013	jsmcERR_WRONG_STATE_FOR_FB	The FB cannot be used in the current state	Check program to call FB's only in allowed states

50014	jsmcERR_WRONG_OP_MODE_FOR_FB	The FB cannot be used in the current mode of operation	Only use allowed FB's for the desired mode of operation (profile position or cyclic synchronized)
50015	jsmcERR_EXECUTION_ERROR	The FB failed during execution by an external error	Check and quit errors from other function blocks or axes, which caused the fault
50016	jsmcERR_BUFFER_TO_SMALL	The buffer for the error text string is to small	Put a pointer to a buffer for the error text string which size is at least 50 characters
50017	jsmcERR_TEXT_OBJ_NOT_FOUND	Error text object or function block text object not found	Enter correct name of the error text object and ensure, that the error text object (JsMcEtxDe/JsMcEtxEn) and the function block text object (JsMcFBtxEn) are present in the project
50018	jsmcERR_TEXT_READOUT_FAILURE	Error text or function block text could not be read successfully	Ensure that the error text object (JsMcEtxDe/JsMcEtxEn) and the function block text object (JsMcFBtxEn) are present in the project
50019	jsmcERR_WRONG_GENERAL_OP_MODE	general mode of operation not supported	Set a supported general mode of operation in JS_MC_Init (OperationMode = jsmcMODE_PROFILE_POSITION or jsmcMODE_CYCLIC_SYNC)
50020	jsmcERR_REF_SPEED_NOT_IN_RANGE	Reference speed for rotative motors is out of range	Use reference speed in allowed range (0...250000 inc/s)
50021	jsmcERR_ZMARK_SPEED_NOT_IN_RANGE	Z-Mark speed for rotative motors is out of range	Use Z-Mark speed in allowed range (0...100000 inc/s)
50022	jsmcERR_VELOCITY_NOT_IN_RANGE	Velocity is out of range	Use velocity in allowed range (10...9000000 inc/s for linear motor, 10...100000000 inc/s for rotative motor)
50023	jsmcERR_ACCE_TO_LARGE	Acceleration is to large	Use smaller acceleration (smaller than 1000000000 inc/s^2)
50024	jsmcERR_CYCLE_TIME_FAILURE	Cycle time setting failure	Use correct cycle time setting (powerlink bus cycle time >= 400us and software task cycle time >= powerlink bus cycle time)
50025	jsmcERR_DECE_TO_SMALL	Deceleration is to small	Use larger deceleration (>=2000 inc/s)

50026	jsmcERR_DECE_TO_LARGE	Deceleration is to large	Use smaller deceleration (smaller than 1000000000 inc/s^2)
50027	jsmcERR_FW_VERS_FAILURE	Firmware version failure	Firmware not supported, use at least XENAX firmware V3.64D and powerlink bus module firmware V2.0 or newer
50028	jsmcERR_PDO_MAPPING_CHK_FAILURE	Failure during PDO mapping check	Error in AsIOPVInfo() function block of AsIO library
50029	jsmcERR_PDO_MAPPING_MISSING	Necessary PDO mapping missing	Check, if all necessary PDOs are mapped in I/O Mapping
50030	jsmcERR_NO_DATA_ADDRESS_ASSIGNED	No data address for error text string assigned	Assign valid data address for error text string
50031	jsmcERR_SDO_ACCESS_FAILURE	Invalid SDO access	Check input values DataObject, SubID and DataLength and set correct values
50032	jsmcERR_CYCLIC_COMM_INTERRUPTED	Cyclic communication interrupted	Don't enable power until JS_MC_CyclicIn is valid and ~~so~~ cyclic communication is running
50033	jsmcERR_SPAD_FAILURE	Wrong set point acknowledge setting	Check the Parameter “SPAD” for correct setting
50034	jsmcERR_INDEX_NOTVALID	Index not valid	
50035	jsmcERR_VALUE_OUTOFRANGE	Value not in range	
50036	jsmcERR_FC_INPUTS_NOTVALID	Force calibration inputs not valid	
50037	jsmcERR_FC_NO_LINEAR	Force calibration only with linear motors	
50038	jsmcERR_FC_REF_ERROR	Force calibration: Error during reference	
50039	jsmcERR_FC_MOTION_ERROR	Force calibration: Error during motion	
50040	jsmcERR_UNKNOWN_MOTORTYPE	Unknown motor type	

2.6 Error sources

The error source block can be found in the ErrorRecord output of the ReadAxisError block. The table below associates sources number with the corresponding function block.

ErrorSource	Error srouce
1	Axis error or warning
2	CyclicIn
3	Power
4	Reference
5	MoveAbsolute
6	MoveRelative
7	MoveCyclicPosition
8	Stop
9	Halt
10	AxisErrorCollector
11	ReadAxisError
12	ReadParameter
13	WriteParameter
14	JogVelocity
15	ReadActualCurrent
16	ReadDigitalInput
17	ReadDigitalOutput
18	WriteDigitalOuput
19	SetPDO
20	ForceCalibration

2.7 Error type

The error type is important for error handling. Because of that, the error type is provided in the error record in an additional field.

ErrorTyp	Funktionsblock im Fehler
1	Achsenfehler
2	Achsenwarnung
3	Fehler in einem Funktionsblock

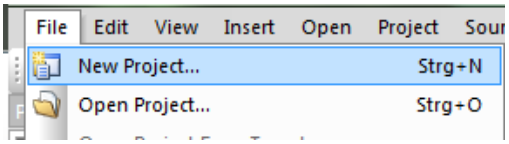
3 Automation Studio

3.1 New Project

In the following steps it will be shown how a new project can be opened.

Start the Automation Studio from B&R.

File -> New Projekt ->

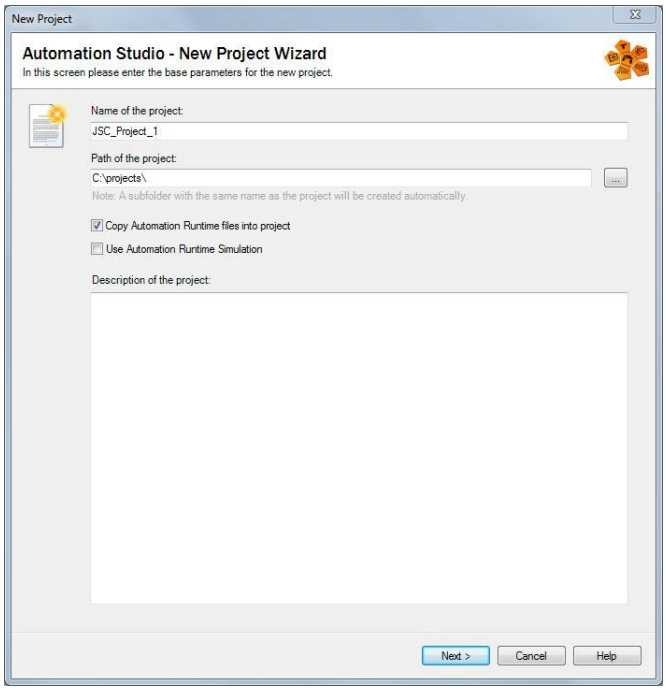


Enter your project name e.g. „JSC_Project_1“
The storage path will be shown and can be changed.

Select
☑Copy Automation Runtime support…..

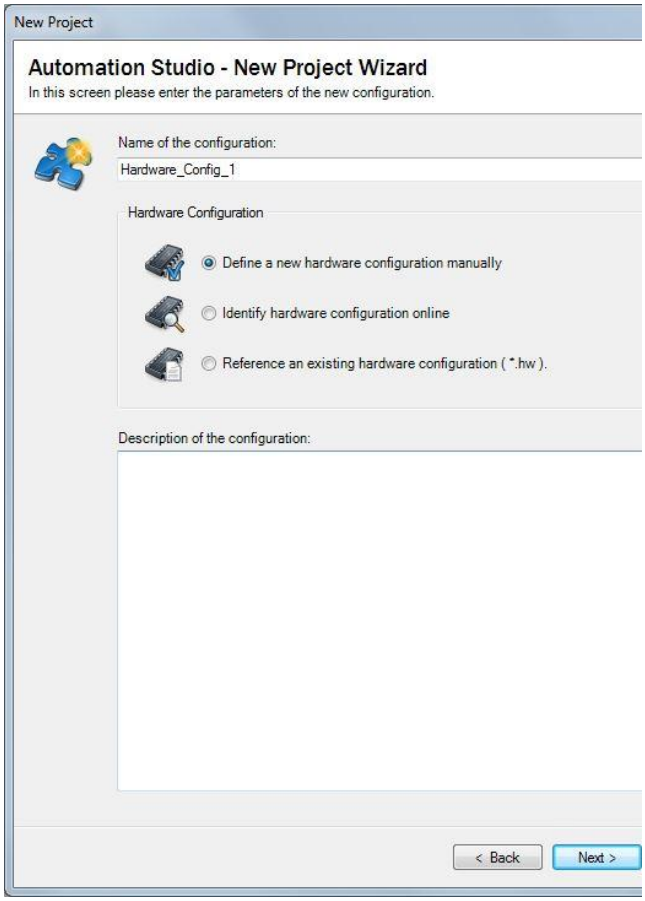
☐Use Automation Runtime Simulation.
It is not necessary to run a simulation if the PLC is used as hardware.

-> Next



A Configuration name suggestion will be shown.

Our Configuration name is:
„Hardware_Config_1“



☑ Define a new hardware configuration manually

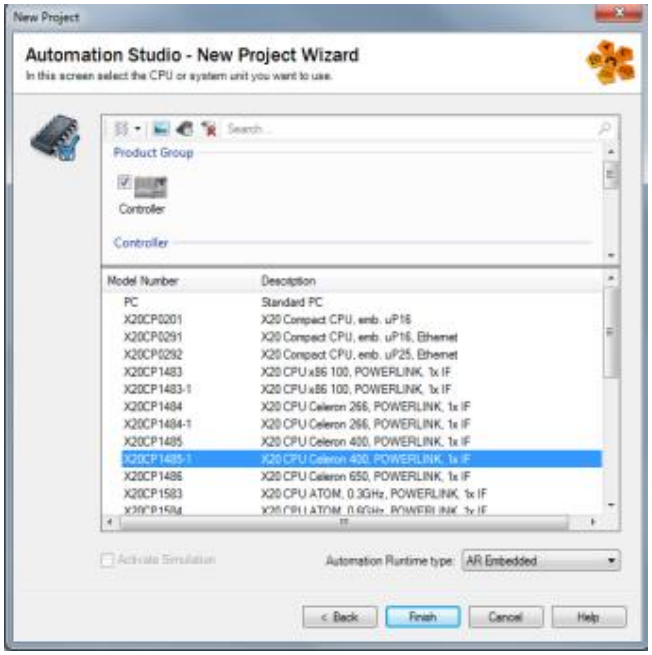
The basic hardware for the configuration can be assembled from the following pages of the wizard. This option has to be selected if Automation Studio is used for the first time and no B&R control is connected to the PC.

☐ Identify hardware configuration online
The hardware configuration will be loaded automatically from the connected B&R control The hardware configuration is automatically visible in the configuration.

☐ Reference an existing hardware configuration
The existing hardware configuration from another Automation Studio project or from another configuration in the same project can be used as a reference.

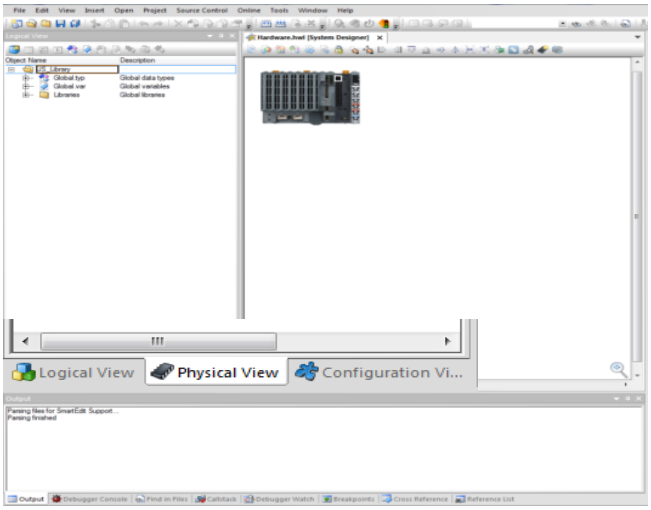
-> Next

We choose the B&R PLC X20CP1485-1

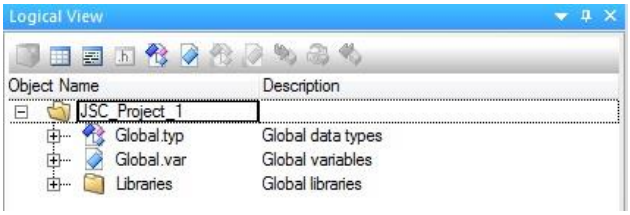


-> Finish

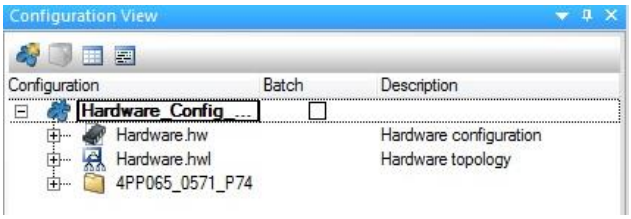
You can choose between three different views: ***Logical View, Physical View*** and ***Configuration View***. After including the B&R control, each view has now basic settings.



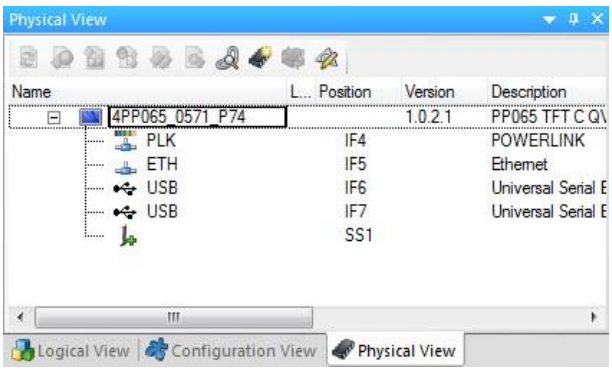
The **Logical View** represents the hardware independent view of the application.
Use this view to list and manage data type declarations, variable declarations, packages, programs, libraries, data objects and documentation files.



The **Configuration View** represents the hardware dependent application perspective. Use this view to display and manage the hardware configuration, software configuration files, declaration of permanent variables, I/O routings, Automation Runtime configuration, configuration NC, NC routings, VC keyboard mapping, configuration-specific data objects and documentation files.

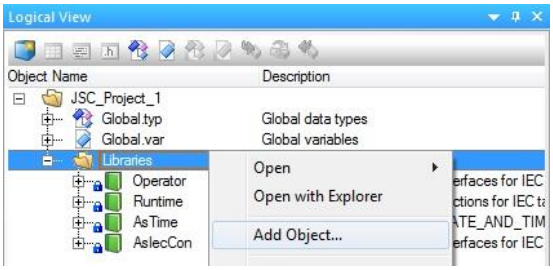


The **Physical View** provides the hardware point view of the active configuration. It displays the active configuration hardware modules in a directory tree.



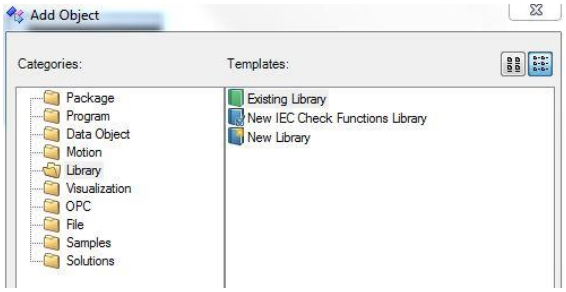
3.2 Import of the JsMcLib library

In the *Logical View* right click on the folder Libraries,
-> Add Object…



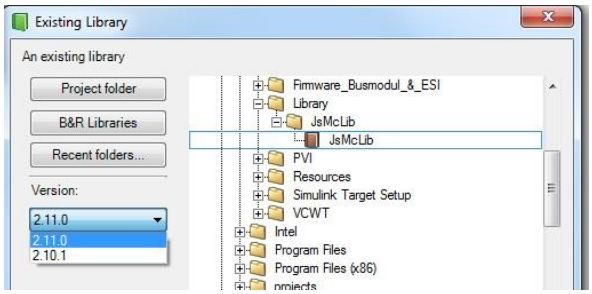
Categories -> Library -> Existing Library

-> Next



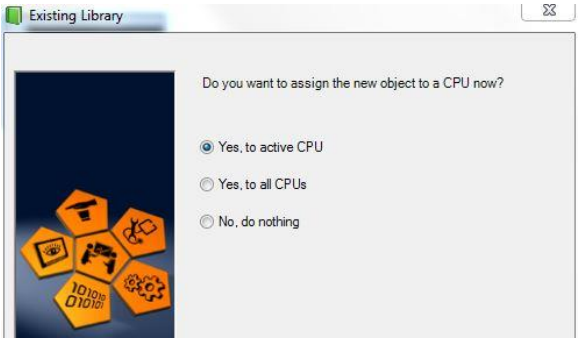
In the following step you have to select the "JsMcLib" library. If there are several versions, you can select the latest version in the "Version" drop-down list on the left side.

-> Next

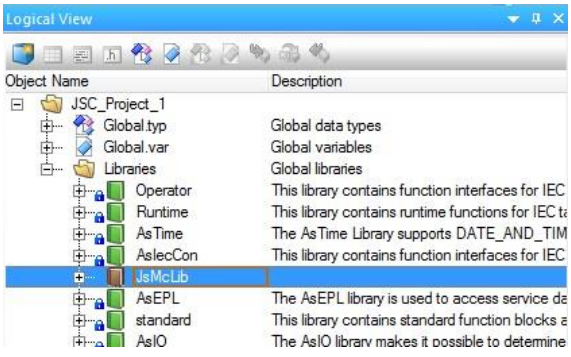


With **"Yes, to active CPU"** the "library" is added to the active "CPU".
By choosing **"Yes, to all CPUs**" the library is added to all CPU's, in the case of more than a configured CPU.
By choosing **"No, do nothing**", the "library" will not be assigned to any CPU. The assignment has to be done manually.

-> Finish

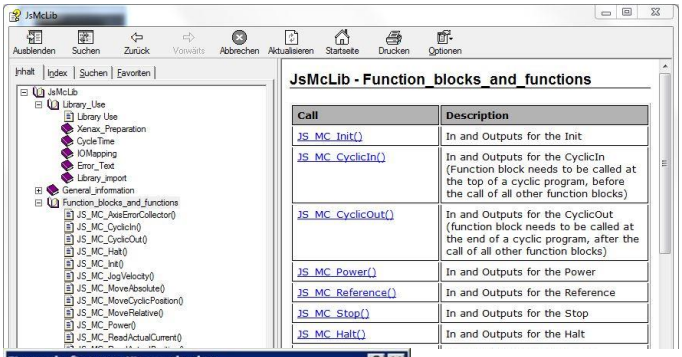


The inserted library is now visible in the *Logical View -> Libraries*..



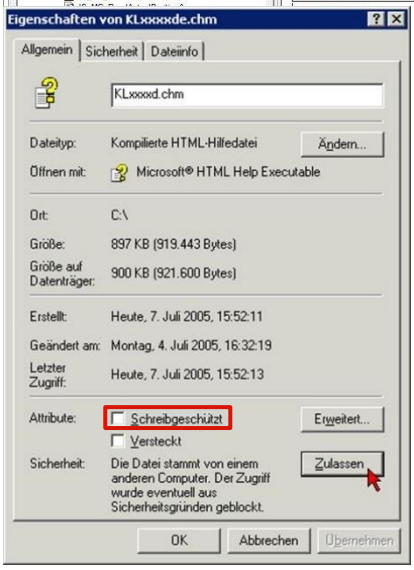
3.2.1 JsMcLib User Help

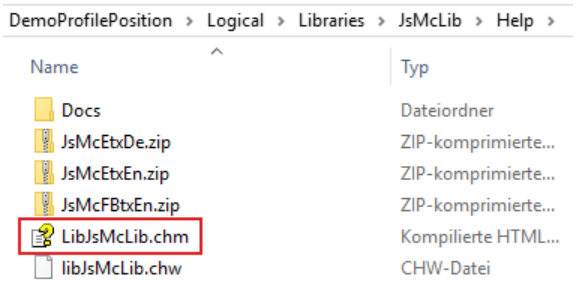
The help for the function blocks of the JsMcLib library, can be opened by selecting the JsMcLib library and then pressing the “F1” button.



It is possible that the help window is completely white. This happens when the PLC project is saved on a network drive.

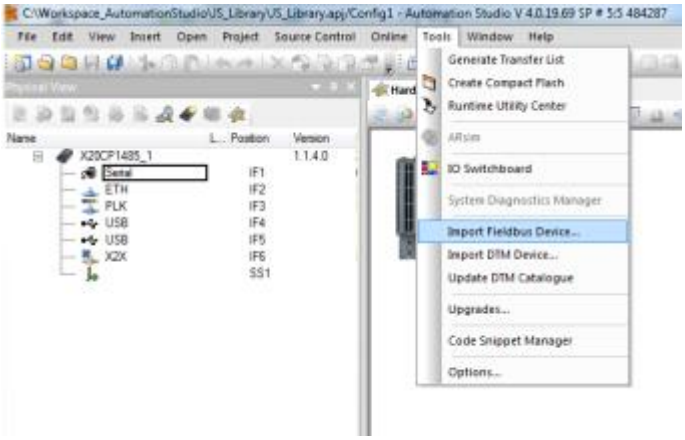
Furthermore, sometimes the CHM file must be activated. This file can be found under: \Logical\Libraries\JsMcLib\Help\LibJsMcLib.chm right click on the file->Properties->General tab->click Allow





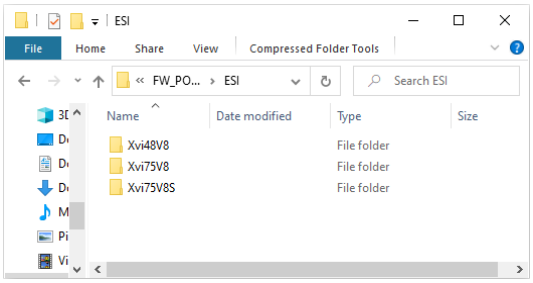
3.3 Embed XENAX® CANopen object file (.XDD)

The XENAX® Servo Controller is operating according to the standardized device profile for electric drives CANopen DS402.
This device profile then goes on the Powerlink Ethernet fieldbus (CANopen over Powerlink).

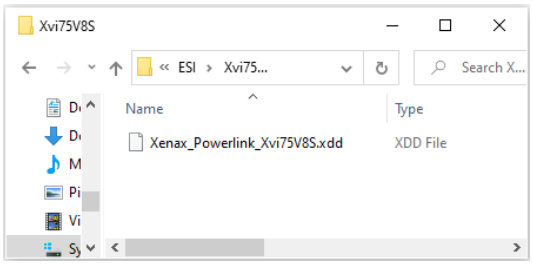


The CANopen object list is stored in an Ethernet Slave Information file or short ESI. This ESI file can be found on the Jenny Science website: *Products->Downloads->XENAX® Servocontroller -> Firmware Bus Module and EDS->FW_POWERLINK*
It is important that EDS file is only used with the compatible Busmodule firmware which is provided in the same zip file as the ESI file.

There are 3 different versions of the XENAX®. Use the folder according to your device.



Choose the ESI file with the .xdd file ending

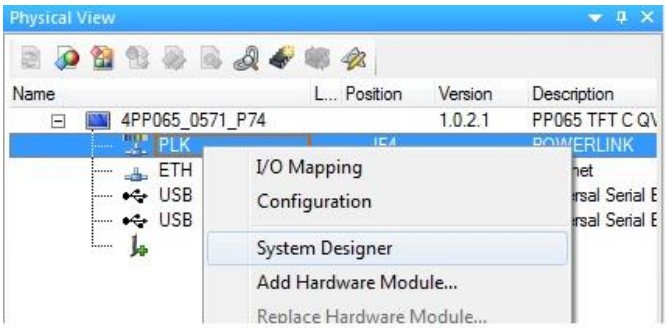


„... imported successfully“ will be shown on the „Output“ window

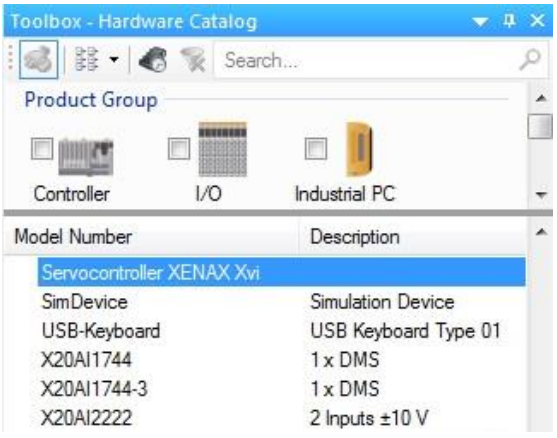
3.4 Embed XEANAX® Servo Controller in project

Include „Servo Controller XENAX Xvi“ from Toolbox into the project

Physical View-> select the Power Link Interface -> right click -> System Designer.



Toolbox-> Hardware Catalog select “Servo Controller XENAX Xvi” and drag into the System Designer.

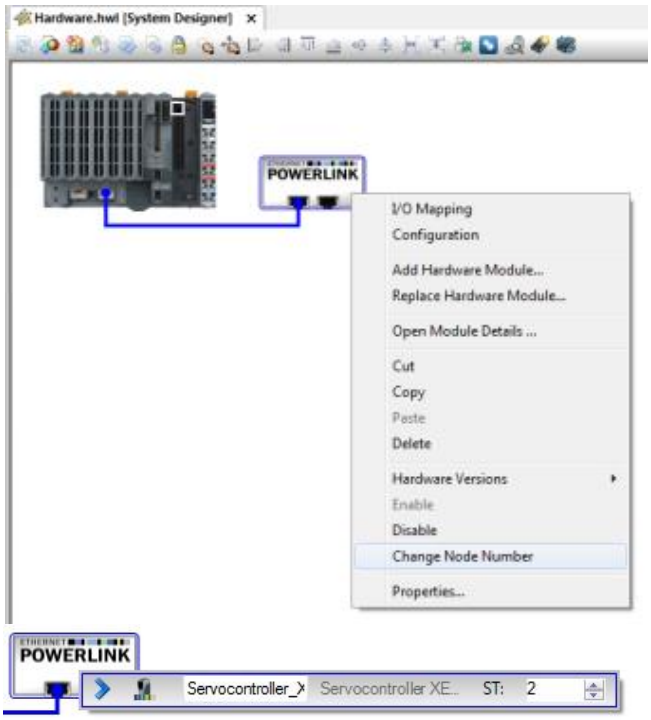


Connect the XENAX with the Powerlink bus.

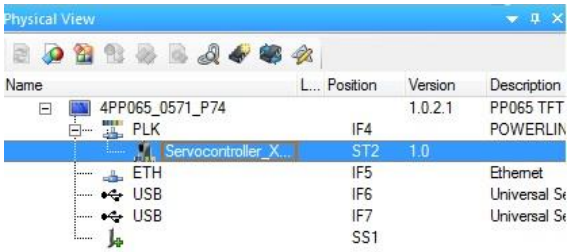
For the connection, the „Node Number" hast to be the same as the "Card Identifier" (CI) on the XENAX® Servo Controller.

The CI number can easily be changed on the WebMotion® under "*by command line" with the ASCII code e.g. "CI2"*
Read with "CI?"

A power cycle is required to apply the new card identifier.

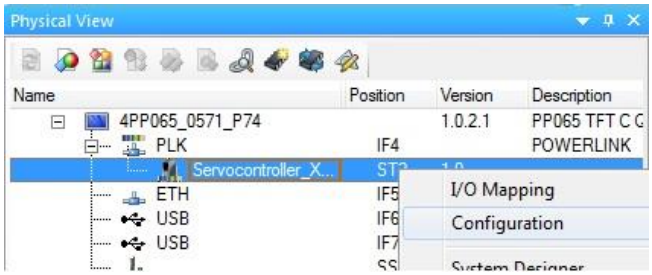


The Servo Controller XENAX® Xvi75V8 can now be seen in the Physical View under PLK (Powerlink).



3.4.1 Configuration XENAX® based on the xdd description

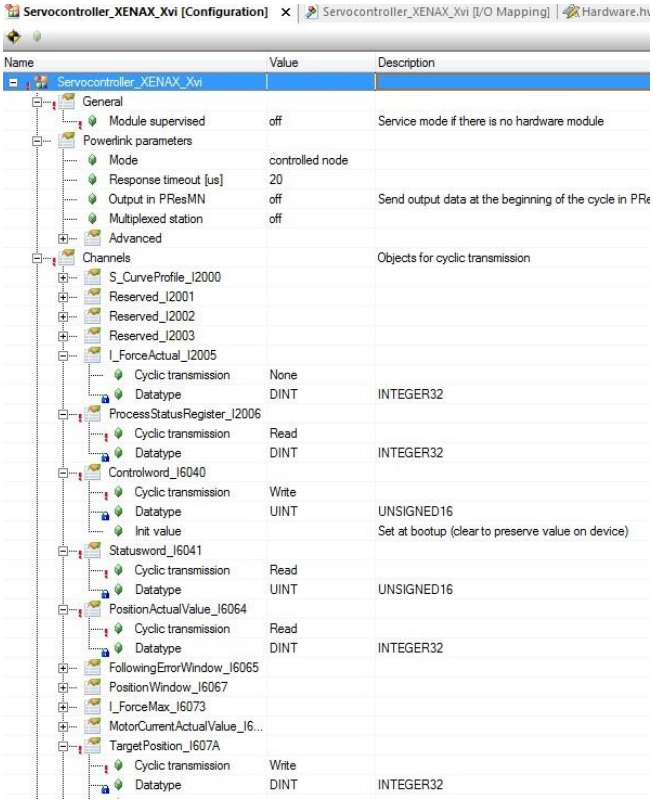
Physical View/PLK/Servocontroller_XENAX_Xvi -> right click -> Configuration.



3.4.2 Cyclic transmission

In the submenu “Channels” all usable objects will be shown.

Important:
The Objects 1001h (Error bit register) and 1F8Ch (NMT_CurrNMTState -> Network Managment status) are not mapable as PDO.
This information should not unnecessarily burden the real-time cycle.



Objects which are transferred cyclic, can be activated.

Fort the JsMcLib library the following objects have to be transferred cyclic in order that the “**Cyclic Synchronized Mode”** can be used.

Name of the Object	**Mode of transmission**
ProcessStatusRegister_I2006	Read
Controlword_I6040Out	Write
Statusword_I6041	Read
PositionActualValue_I6064	Read
TargetPosition_I607AOut	Write

If the **„Profile Position Mode“** is used, the following object have to be transferred additionally.

Name of the Object	Mode of transmission
S_CurveProfile_I2000Out	Write
ProfileVelocity_I6081Out	Write
ProfileAcceleration_I6083Out	Write

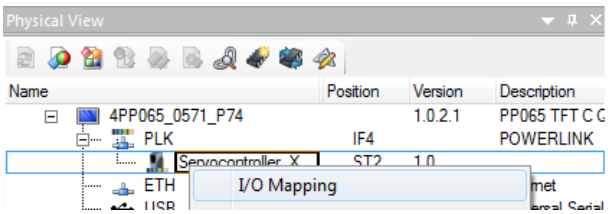
A detailed description of the Objects can be found in the document *Xvi75V8_CANopen_Ethernet* at page 17.
The document can be found on our website:
www.jennyscience.ch/downloads

Note:
If there are several XENAX® Servo Controller used, the object settings can easily be copied:

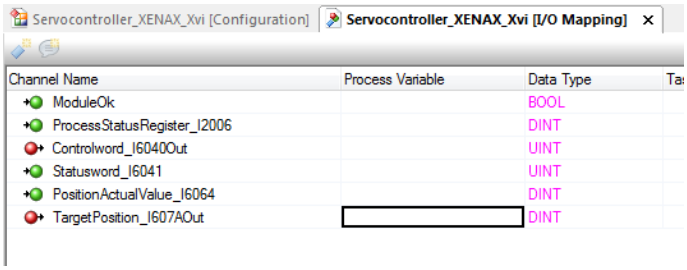
1. Select all object settings with „Ctrl“+ „A“.
2. Right click-> *Copy*
3. *Open I/O Configuration* of the second Servo Controller select all objects with „Ctrl“+ „A“.
4. Right click -> paste

3.4.3 View I/O Mapping

The objects will be visible under the *I/O Mapping (Physical View)*.
In case the objects can't be seen, please check the settings of the *I/O Configuration.*



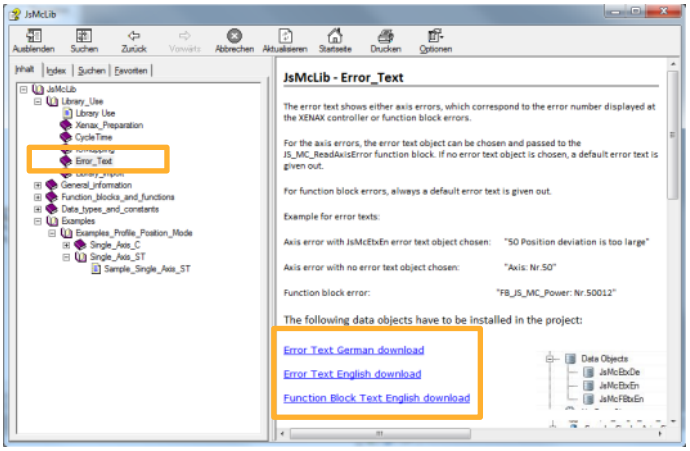
The I/O mapping should look like displayed on the right
.



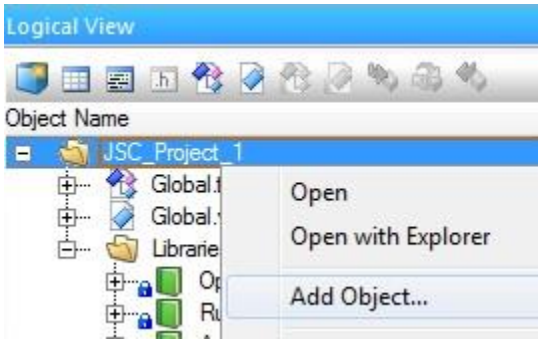
3.5 Embedding XENAX® Error Messages

Error on the XENAX® Servo Controller can be read with the „JS_MC_ReadAxosError" function block.
To provide this function block with an error text the "JsMcEtxEn" file has to be integrated into the Automation Studio.

Under JsMcLib ⊞ Library_Use -> Error_Text, the corresponding file in in German or English can be selected. Specify a desired location and unzip the file by clicking right and press -> Extract All.

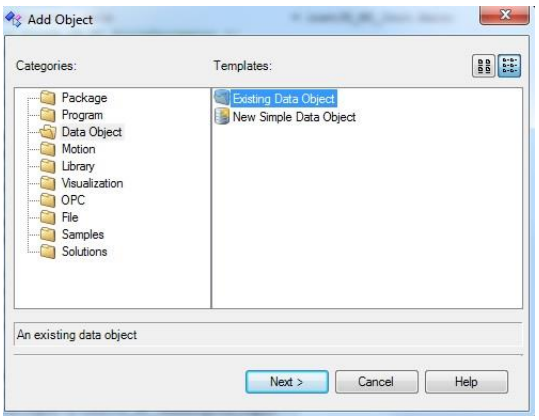


In *Logical View select* the project name, „JSC_Project_1",
-> Right click -> *Add Object*

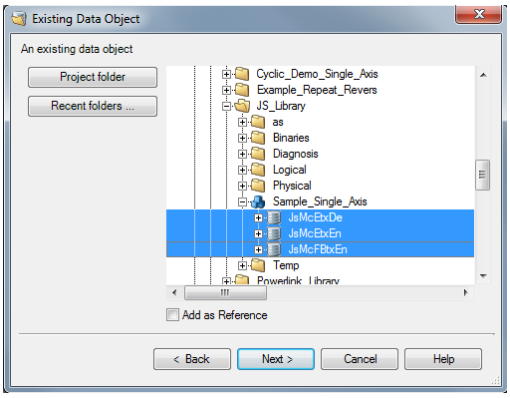


Categories: Data Object -> Templates: Existing Data Object

->Next

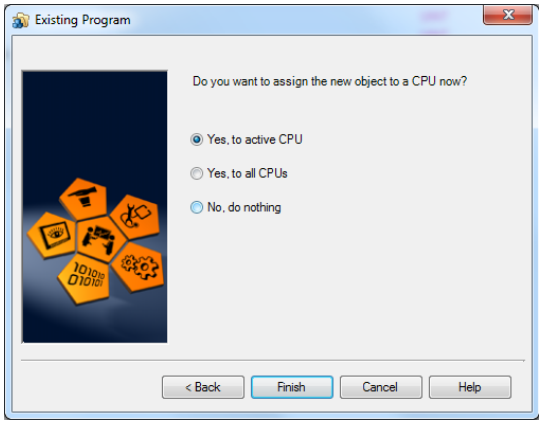


Select all 3 files („JsMcEtxDe", „JsMcEtxEn" and „JsMcFBtxEn").



-> Next

Yes, to active CPU adds the "Error text" to the active CPU (here X20CP1485-1.).

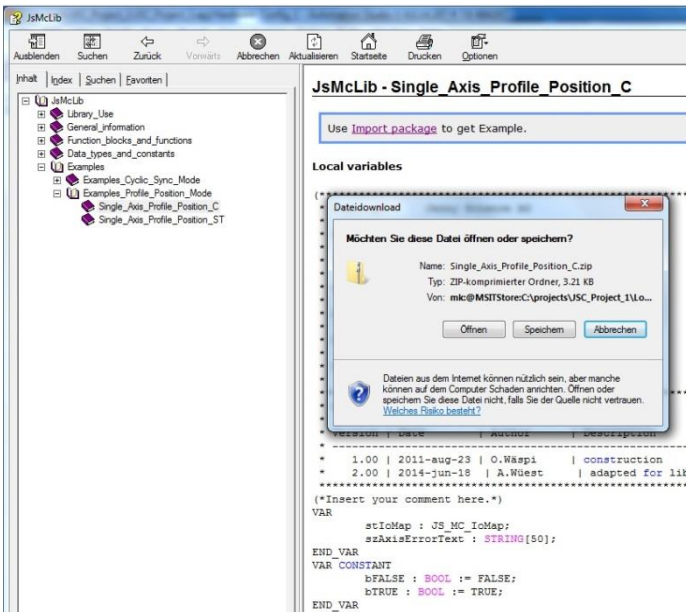


-> Finish

4 Program example “Profile Position Mode“

4.1 Get program example and save it

The program example is stored as c-coder or structured text in the “user help” menu of the JsMcLib.
Firstly the example needs to be saved to the PC and then it can be imported to the project.



Logical View -> select JsMcLib Library -> press F1 to open the user’s help.

Under Examples the program examples can be found.

In our Example we choose “Single_Axis_Profile_Position_C” and click on “Import Package”.

Then we save this file on the PC and extract the .zip-file. Right click -> “extract all”

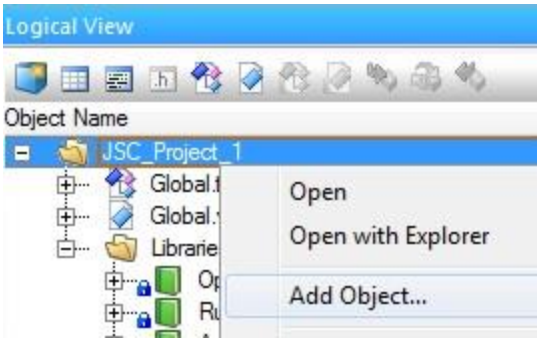
Note:

Program
A program has direct access to all variables, functions and function blocks.

Package
Bigger Applications including a big amount of data objects, variables, functions and so on can be split into single units.
The summary of these units is called a „Package“ and can be named and imported.

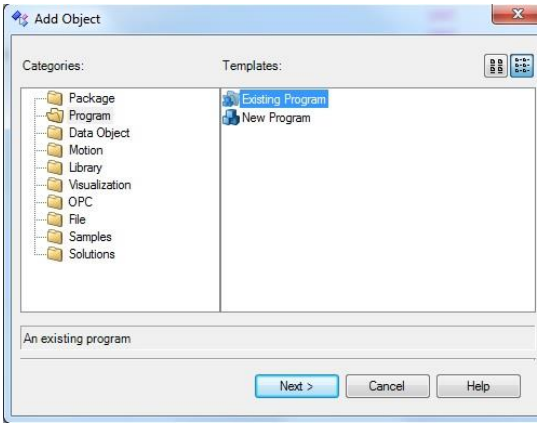
4.2 Import program example to the project

Under *Logical View,* select the, „JSC_Project_1“.
-> Right click -> *Add Object*



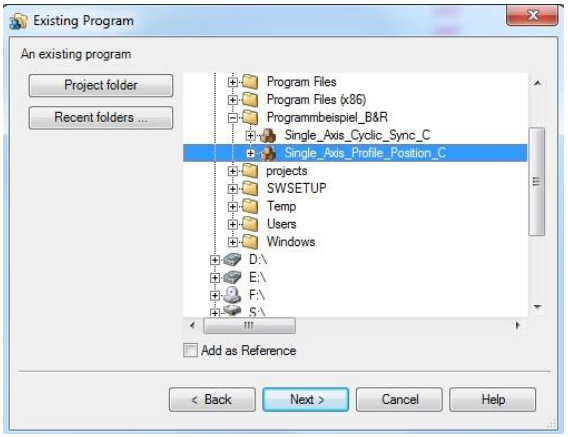
Categories -> Program -> Templates ->Existing Program

-> Next



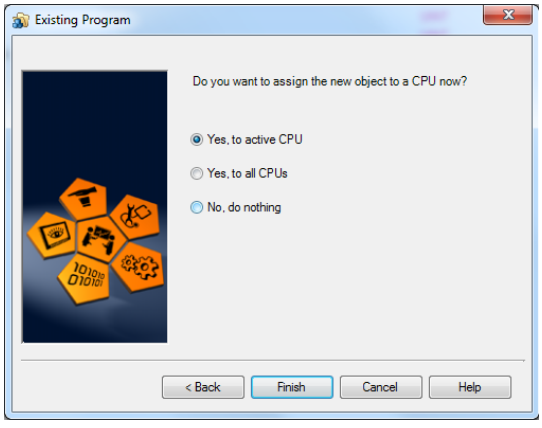
Select the program example, „Single_Axis_Profile_Position_C“.

-> Next



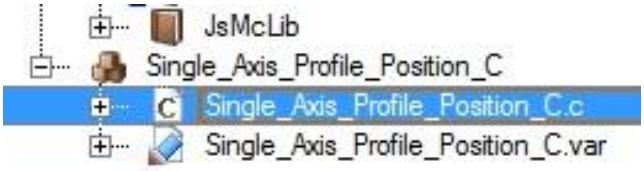
Yes, to active CPU adds the program example to the active CPU (here X20CP1485-1.).

-> Finish



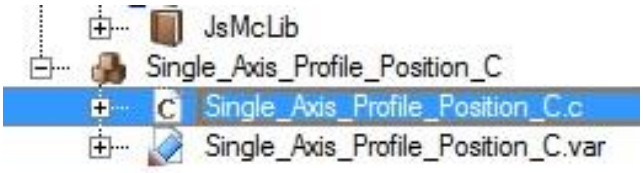
Check

Under *Logical View* the „*Single_Axis_Profile_Position_C*“ now appears.



4.3 Configure Ethernet interface

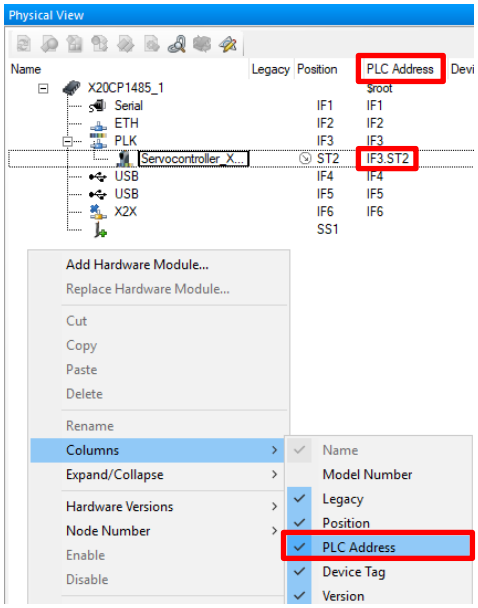
With double click on the “Single_Axis_Profile_Position_C.c” the program code is opened. The program code includes the Ethernet Powerlink interface address (IFx), Node number and the used error object.



Adjust Ethernet Powerlink address EPL and Node nr. Setting in the example code:

The required values can be found in the physical view in the column PLC Address. This column must be added manually with a right click.

IF3 refers to the connection interface on the PLC where the XENAX® is connected. ST2 stands for the XENAX® with the node nr. 2. This number must be equal to the card identifier of the XENAX®.



The values in the example program must be adjusted like shown on the right. In a project with multiple PLCs, the PLC number must be defined too.

Beispiel mit einer SPS

PLC Address: **IF3.ST2**

```
instJS_MC_Init.pDevice  = (UDINT)"IF3";
instJS_MC_Init.Node     = 2;
```

Beispiel mit mehreren SPSen

PLC Address: **SL2.IF1.ST6**

```
instJS_MC_Init.pDevice  = (UDINT)"SL2.IF1";
instJS_MC_Init.Node     = 6;
```

The following line of the program example has to match the interface used.

```
instJS_MC_Init.pDevice = (UDINT)”IF3”;
```

```
/* pass address of interface address */
instJS_MC_Init.pDevice = (UDINT)"IF3";
```

Node nr. Setting:

The Node number is set to 2 in the Line `instJS_MC_Init.Node= 2;`.

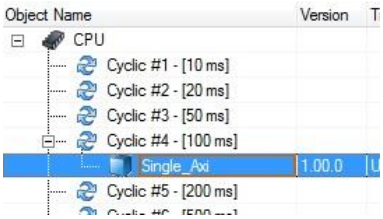
Set the Node number of the connected XENAX® Servo Controller (Node / Card Identifier CI, set with WebMotion®, see chapter “Card Identifier”).

```
/* initialize node number */
instJS_MC_Init.Node = 2;
```

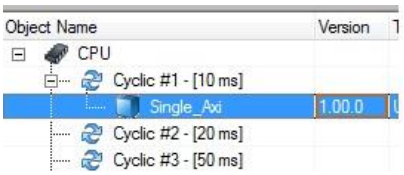
4.4 Move Program example into Task

To see the *Software Configuration* go to *Physical View* and double click on B&R PLC (here *X20CP1485-1*).

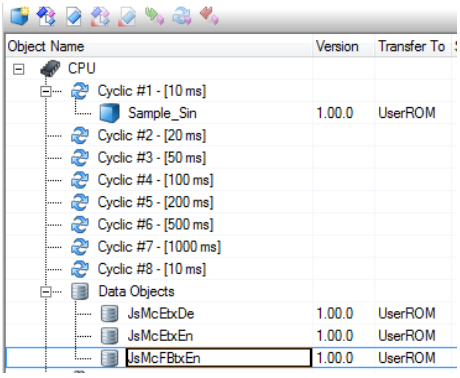
The default cycle of the program is “Cyclic #4” (100ms).



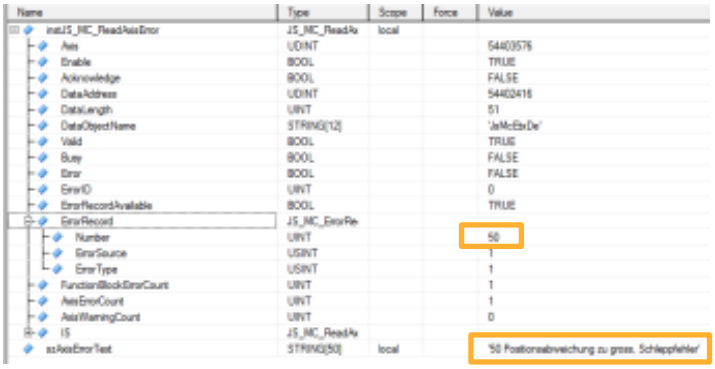
We move the program by drag & drop in “Cyclic #1” which has a shorter cycle time (10ms).



To verify the correct integration, the error objects can be found in „Data Objects“.



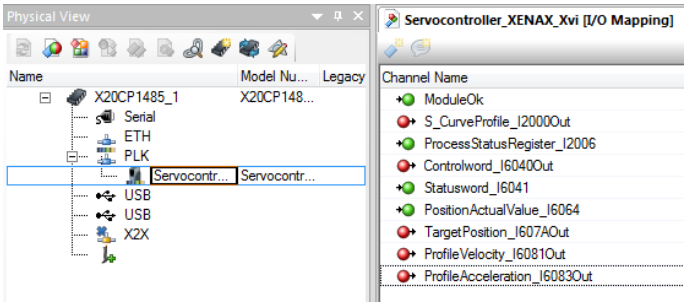
Example:
If the XENAX® Servo Controller shows the Error 50 on the display, the function block “JS_MC_ReadAxisError” will show, “50 position deviation is too large”**.**



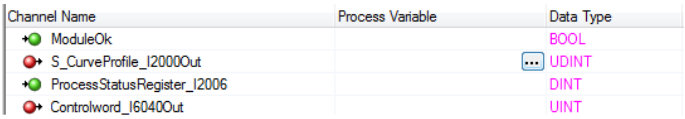
4.5 I/O Mapping, connect .xdd channels to program variables

To use the .xdd channels (cyclic data from the XENAX®) in the library, they have to be connected to the “Process Variable”.

Physical View, double click XENAX® Servo Controller -> *I/O Mapping*.



Select the Channel on the row “Process Variable” (e.g. S_CurveProfil_I2000Out). Click on the ... symbol.

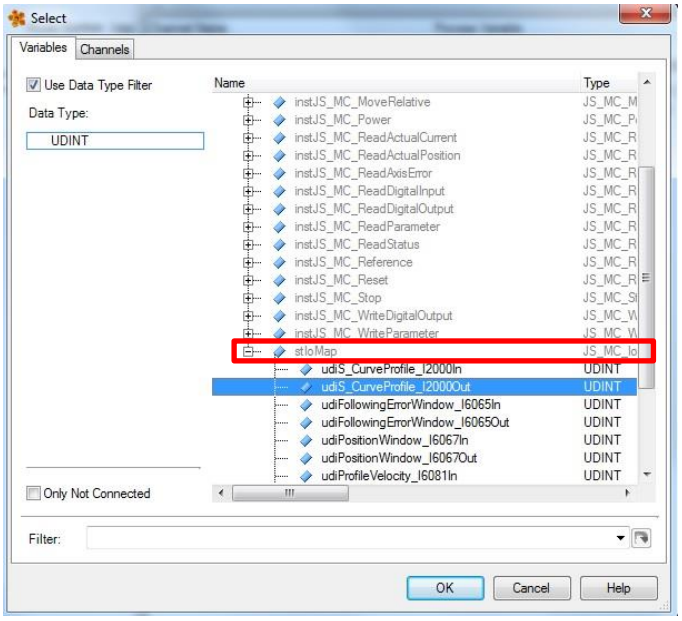


The “Select” window will open.

You can open the according program which you like to use, with the symbol ⊞

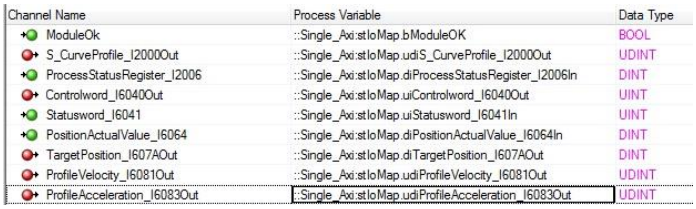
In the tree layout of ⊟ „stIoMap“ are all “Process Variables” which have a data type which matches the channel.

-> OK



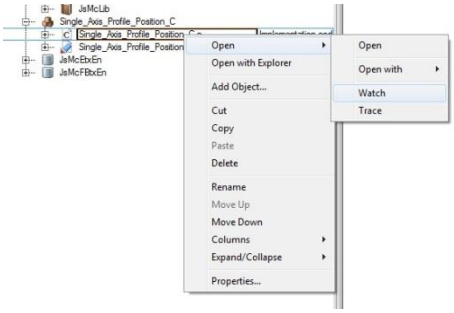
After selecting the variable, the path of the variable can be seen in the I/O Mapping.

The assignment of a “Process Variable” has to be completed for every channel.

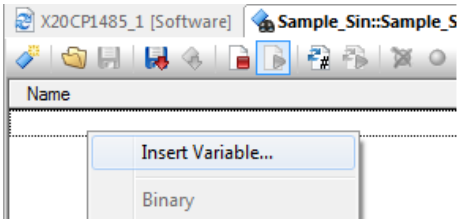


4.6 Start Program Example

Logial View -> double click on the B&R PLC -> right click on the program example *„Single_Axis_Profile_Position_C“ -> Open -> Watch*



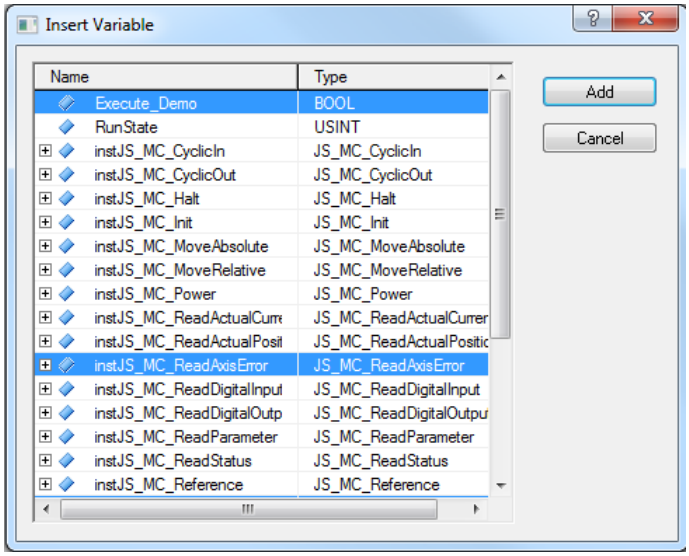
An empty window will be opened, right click -> *Insert Variable*



The window „Insert Variable“ will be opened. Every function block and variable which is used in the program example can be seen.

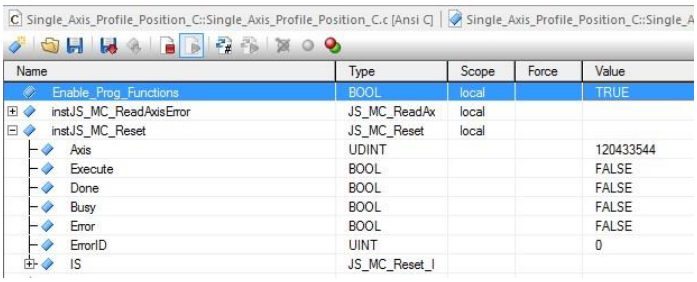
Select the required variables and function blocks. You will need at least the following:
Execute_Demo
instJS_MC_ReadAxisError
instJS_MC_Reset
szAxisErrorText

->Add



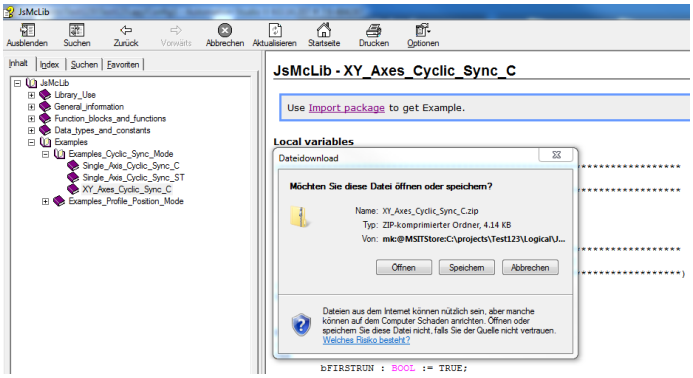
To Start the program example set the „Enable_Prog_Functions“ to TRUE. With FALSE the program is idling in a loop.

If an error occurs it can be quit by putting the variable “Execute” on TRUE in the function block “JS_MC_Reset”.



5 Program example Cyclic Synchronous Position Mode

The program example is stored as c code or structured text in the user's help of the JsMcLib. Firstly the example needs to be saved to the PC and then it can be imported to the project.



Logical View, select JsMcLib Library, press F1 to open the user's help.

Under "Examples" the examples program can be found.

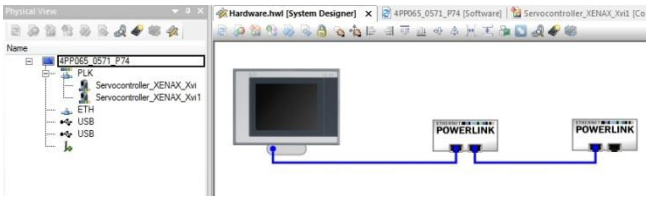
In our Example we choose "XY_Axes_Cyclic_Sync_C" and click on "Import Package".

Then we save this file on the PC and extract the .zip-file. Right click -> "extract all"

This program example works with an X-Y cross table and interpolates a circle. Therefore two XENAX® Servo Controllers are needed.

5.1 Integrate second XENAX® Servo Controller in project.

Integrate an additional XENAX® Servo Controller in project like seen above under „ Embed XEANAX® Servo Controller in project"



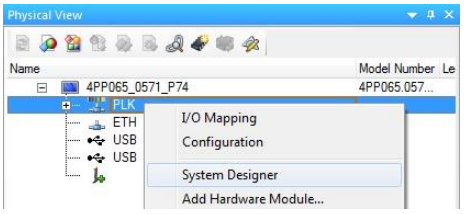
Please note that the node numbers (1 and 2) have to be different in the program code.

5.2 Insert Virtual Axis AS V4.0-V4.6

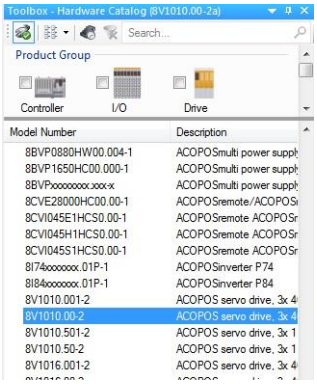
For the **Cyclic Synchronous Position Mode**, a virtual axis is needed in the Automation Studio. The Virtual Axis calculates and transmits a new position for the XENAX® Servo Controller every [ms]

5.2.1 Add Virtual Axis

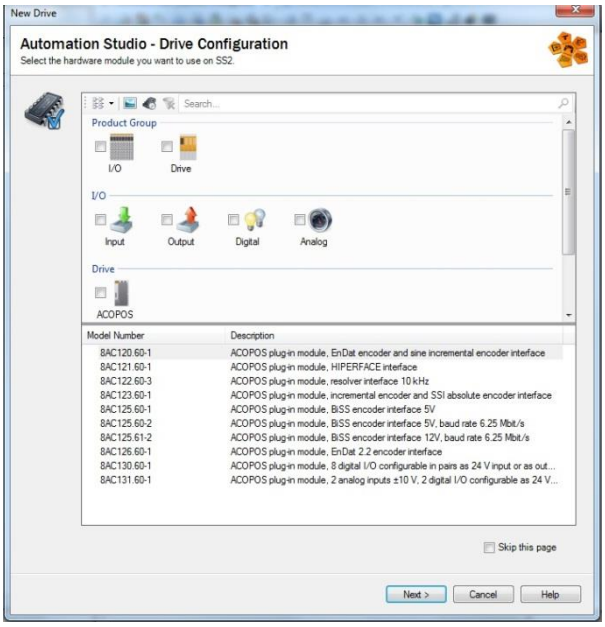
In *Physical View* select right click the power link interface (PLK) -> *System Designer*.



From the “Toolbox Hardware Catalog” add the „AOPOS servo drive 8V1010.00-2“ by double click.

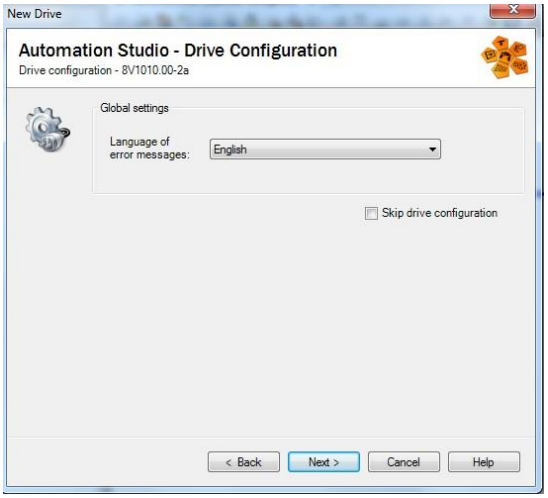


The window „New Drive“ can be confirmed three times with ->*Next*.



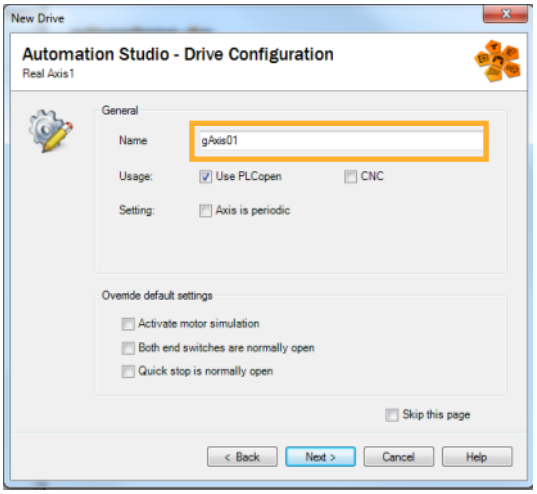
Choose the language for the error messages of the virtual axis.

->Next



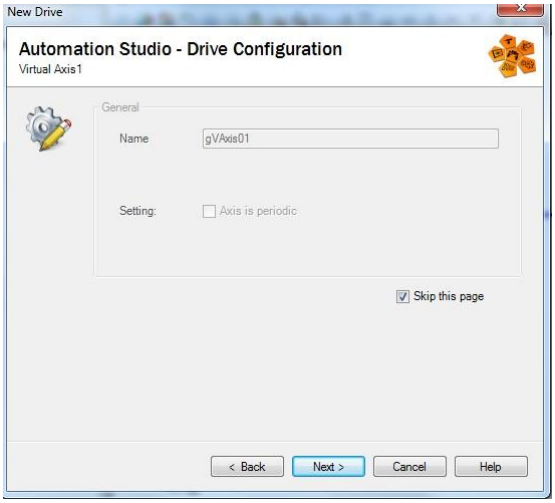
Write a name for the virtual axis.

->Next



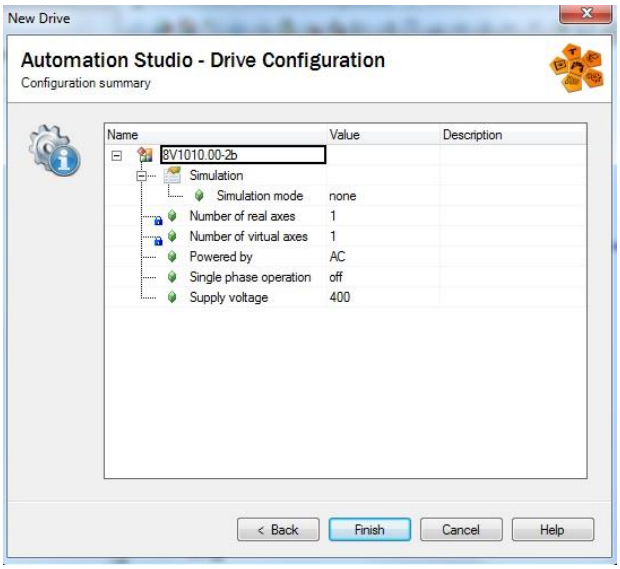
Confirm this window with

->Next



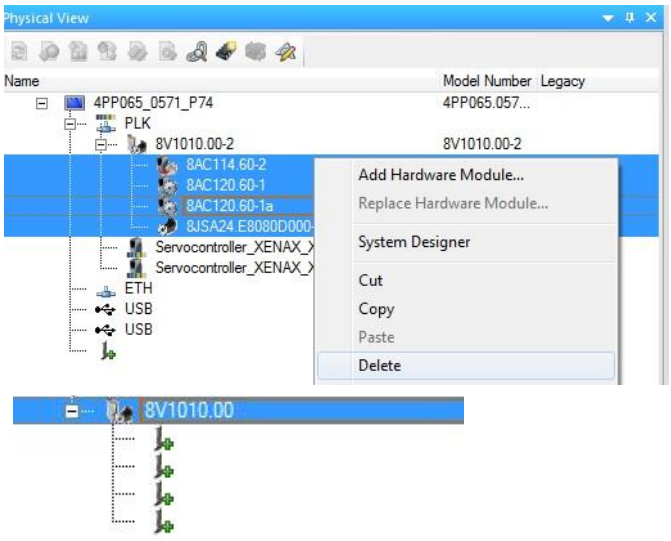
Confirm the „Configuration summary“ with

->Finish

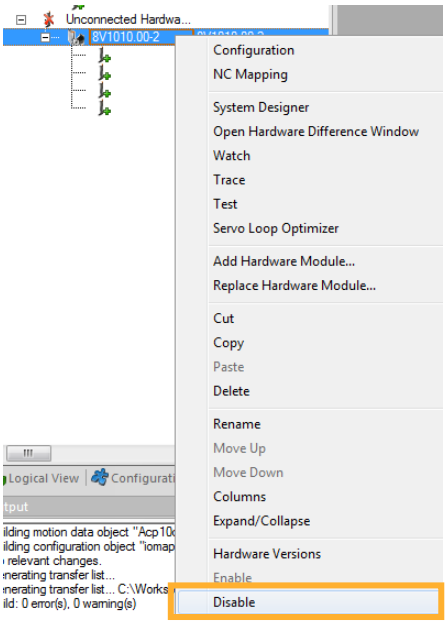


5.2.2 Disable Virtual Axis

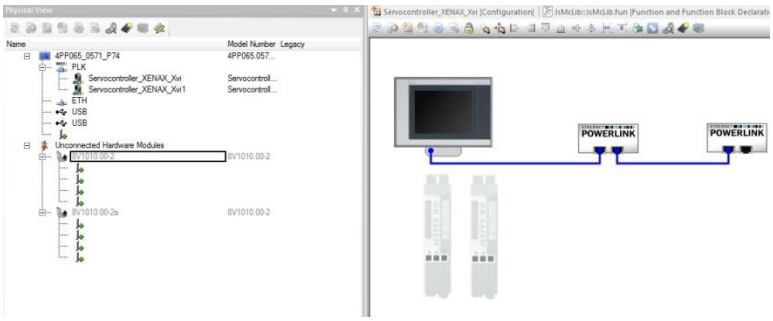
In the *Physical View* under the B&R PLC (here *X20CP1485-1*) the new virtual Axis with all its functionalities can be seen.
Because we only use this Axis virtual, we need to delete all the function objects.
Select all -> right click-> Delete



Because there is no real connection between the B&R PLC and the virtual Axis, we need to disable the Axis.
Right click -> Disable

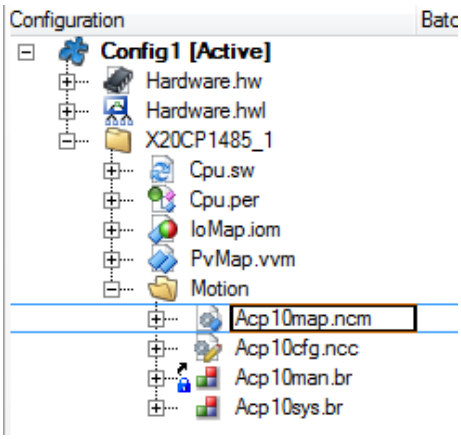


Note:
This example works with 2 Axis.
A second virtual Axis has to be included.

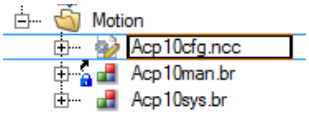


5.2.3 Configure Virtual Axis

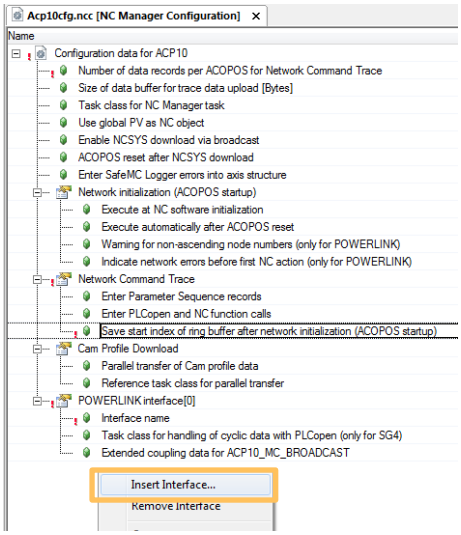
In the *Configurations View* under *X20CP1485_1 -> Motion.* Mark “Acp10map.ncm“ and delete.



To open the NC Manager configuration double click on “Acp10cfg.ncc”.



Right click on an empty spot in the NC manager Configuration -> Insert Interface…



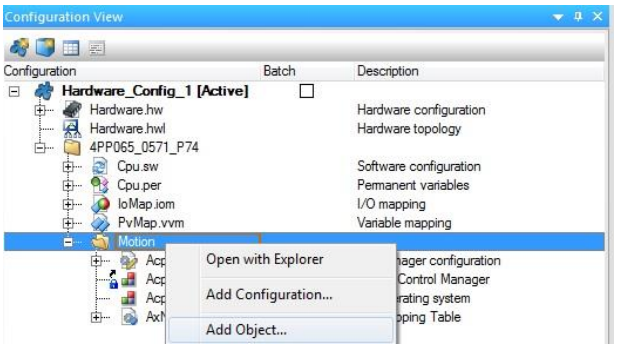
Select Smart Device Controller (SDC) and confirm with *OK*.

Note: This step has to be made twice, once for each axis.

5.2.4 Insert the NC-Mapping Table

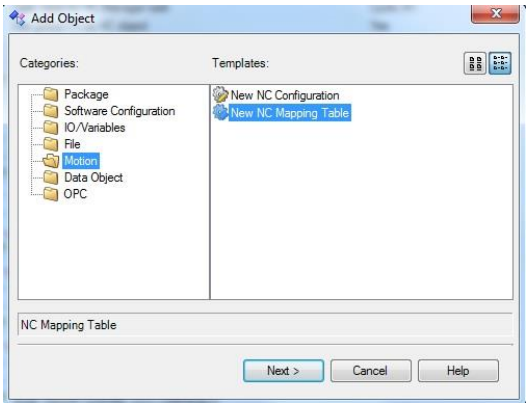
The NC Mapping table will be used for every virtual axis and needs to be inserted only once.

Under *Configuration View -> X20CP1485_1 -> right click on Motion -> Add Object...*



Categories -> Motion -> Templates -> New NC Mapping Table

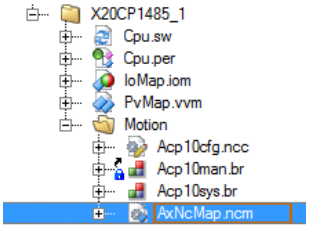
-> Next



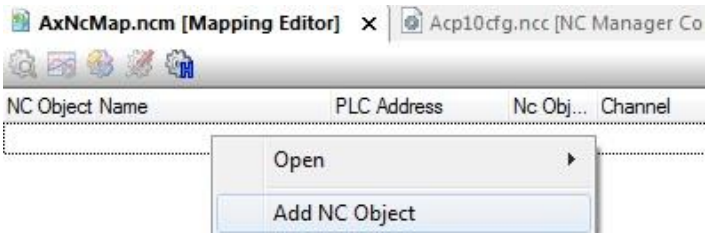
Insert name *„AxNcMap.ncm“*

-> Finish

To open The NC Mapping Table double click *AxNcMap.ncm* in the "*Configuration View*".

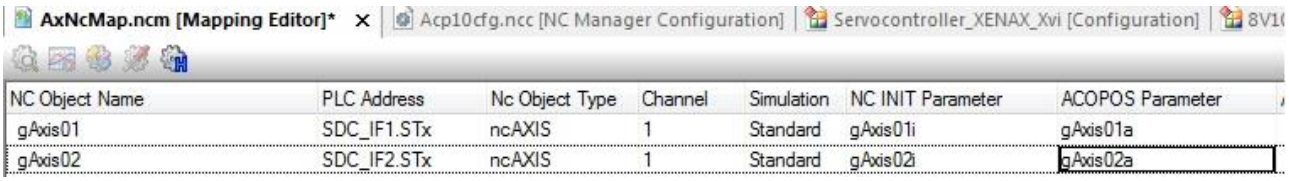


Right click "*Add NC Object*"



The following parameters need to be set for the NC object.

„NC Object Name" is similar to the name of the inserted virtual Axis.
This step is also taken twice

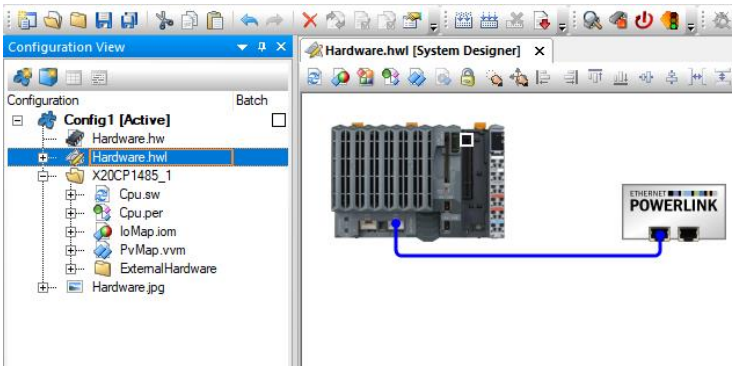


NC Object Name	PLC Address	Nc Object Type	Channel	Simulation	NC INIT Parameter	ACOPOS Parameter
gAxis01	SDC_IF1.STx	ncAXIS	1	Standard	gAxis01i	gAxis01a
gAxis02	SDC_IF2.STx	ncAXIS	1	Standard	gAxis02i	gAxis02a

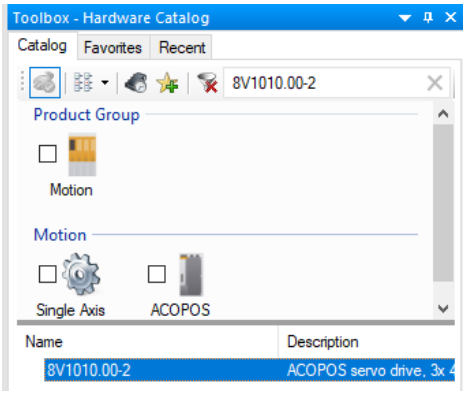
5.3 Insert virtual Axis in AS >V4.7

5.3.1 Add Virtual Axis

Go to System Designer by double-clicking on Hardware.hwl.

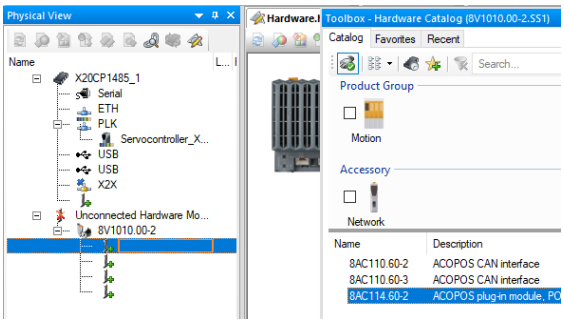


Drag and drop the 8V1010.00-2 from the Toolbox Catalog into the System Designer.

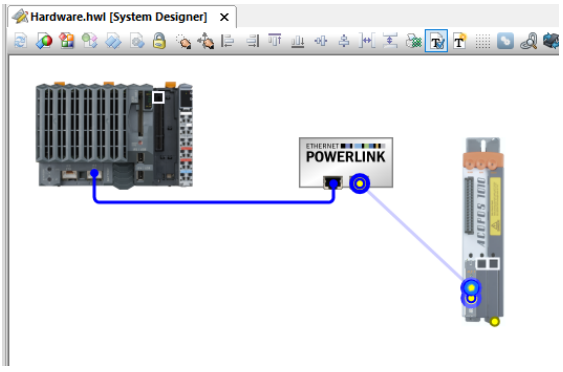


Select the first sub element of the newly created virtual axis.

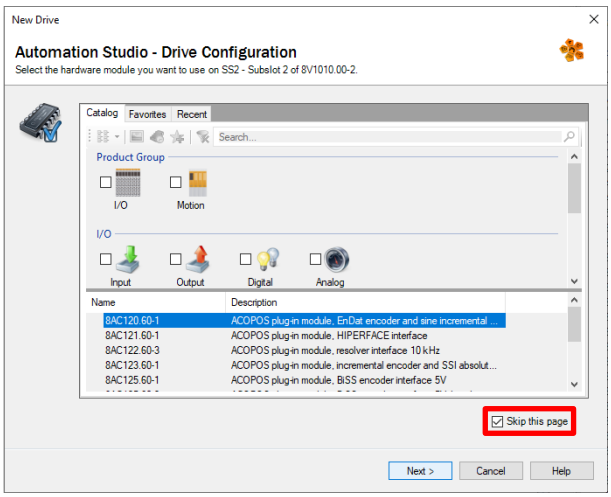
Add the 8AC114.60-2 Powerlink interface to the virtual axis.



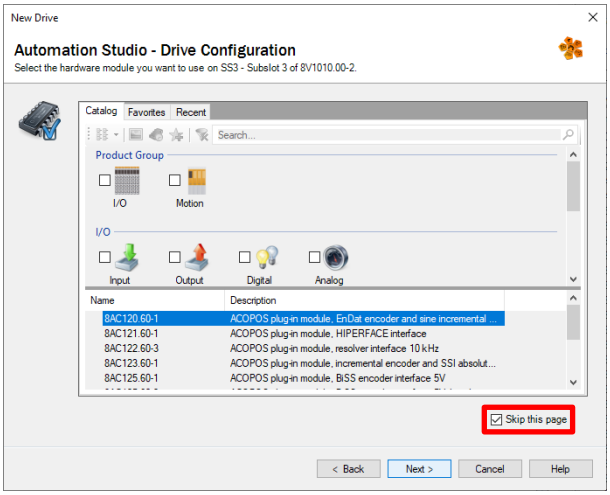
Connect the virtual axis with the Powerlink bus.



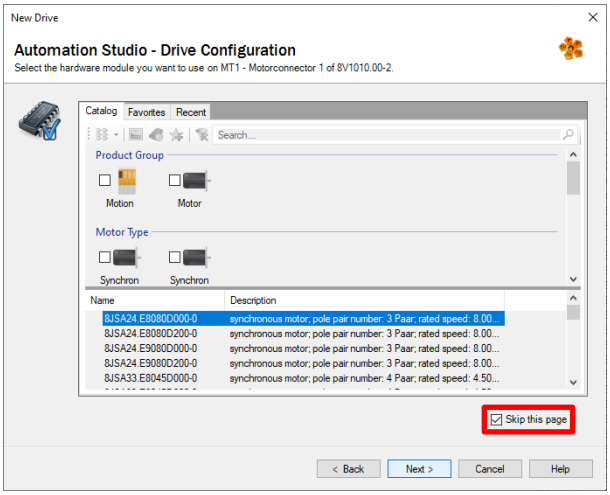
Select “Skip this page“ and press next.



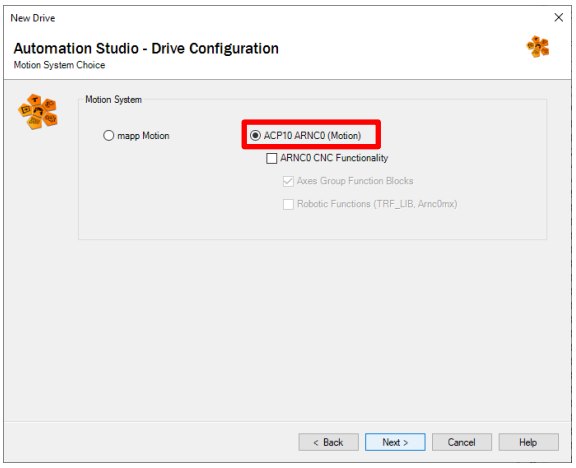
Select “Skip this page“ and press next.



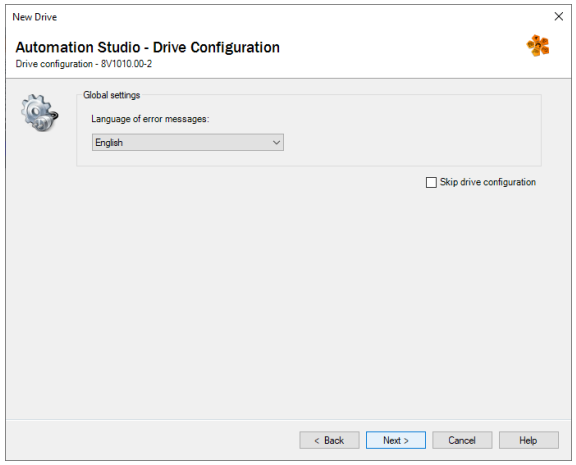
Select “Skip this page“ and press next.



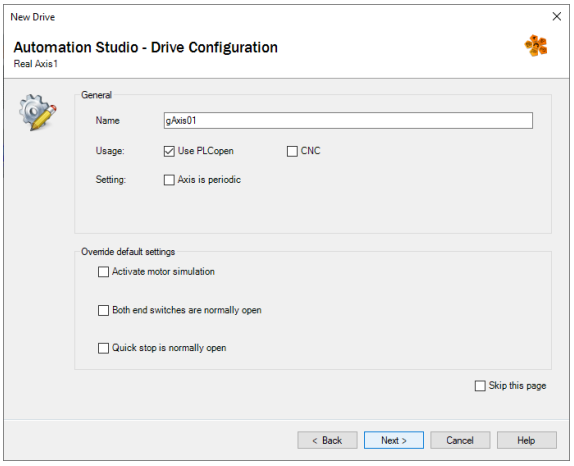
Select ACP10 and press next.



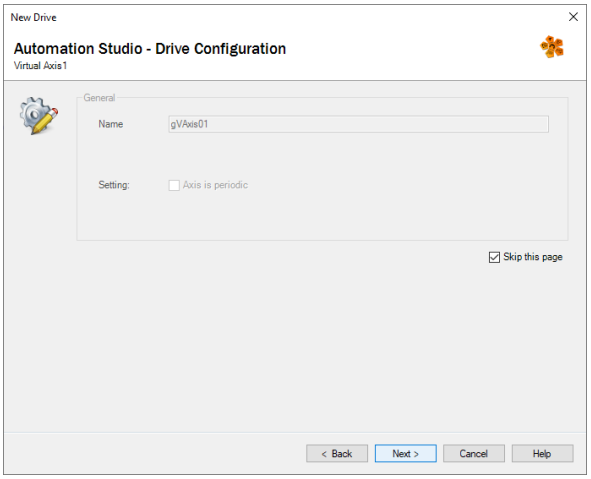
Select your preferred language and press next.



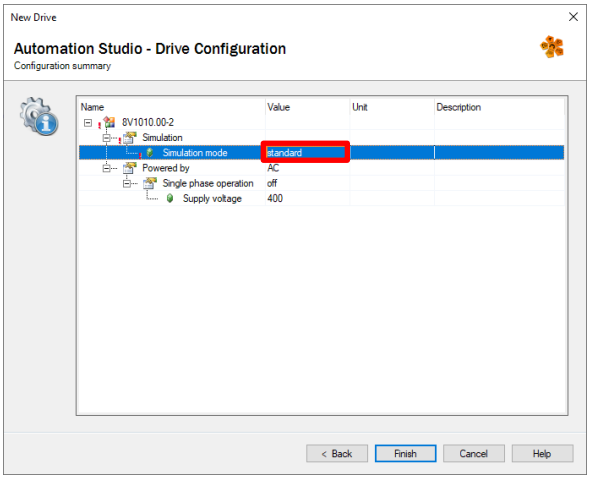
Leave at default and press next.



Leave at default and press next.

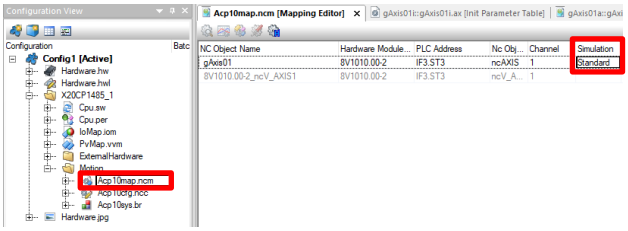


Change simulation mode to standard and press finish.



5.3.2 NC Object Settings

Change also the NC Object to Standard Simulation.



5.4 Parameter Adjustments

5.4.1 Axis Parameter

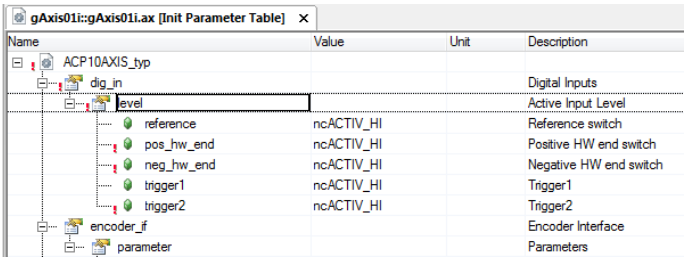
Open the *Logical View* select the inserted virtual Axis:
gAxisXXobj -> gaxisXXi double click.

The following parameter need to be adjusted:

ACP10AXIS_typ -> dig_in -> level

Adjust the Input functions as follows:
pos_hw_end to „ncACTIV_HI”
neg_hw_end to „ncACTIV_HI”
trigger2 to „ncACTIV_HI“

Input function

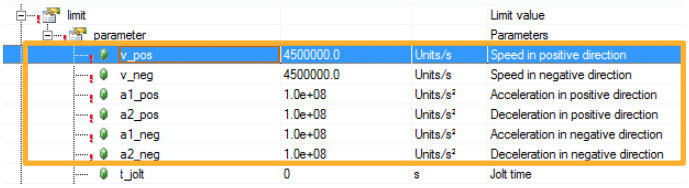


Velocity, Acceleration and brake ramp

Set the maximum velocity (Speed), maximum acceleration (Acceleration) and maximum brake ramp (Deceleration), for the Project.

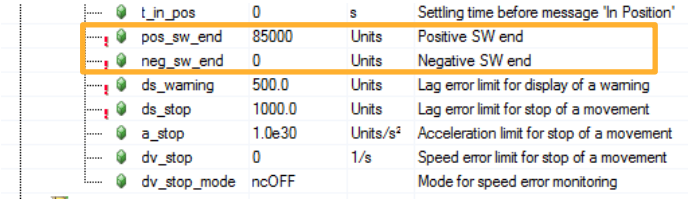
ACP10AXIS_typ -> limit -> parameter

Speed (**v_pos** und **v_neg**),4’500’000
Acceleration (**a1_pos** und **a1_neg**),1.0e + 8
Deceleration (**a2_pos** und **a2_neg**),1.0e + 8
Set to the maximum of the linear motor axis.



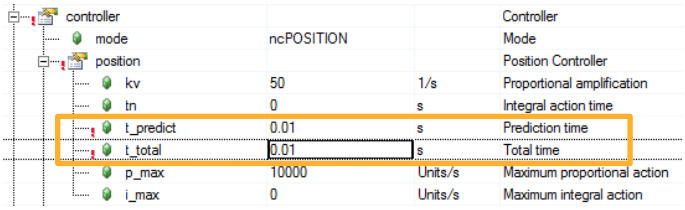
Stroke

The Stroke has to match the mechanical limits of the linear motor.
In our example a ELAX® Ex30F20 is used. The maximum stroke of it is 30000 µm:
neg_sw_end = 0
pos_sw_end = 30000



Time

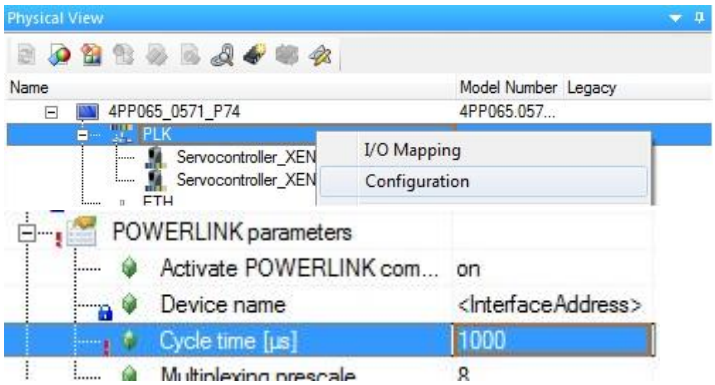
Time adjustment
ACP10AXIS_typ -> controller-> position
t_predict
t_total



Note: This parameter adjustment has to be done for every virtual Axis.

5.4.2 Cycle time

The time used should be as short as possible.
Physical View -> PLK -> right click -> Configuration

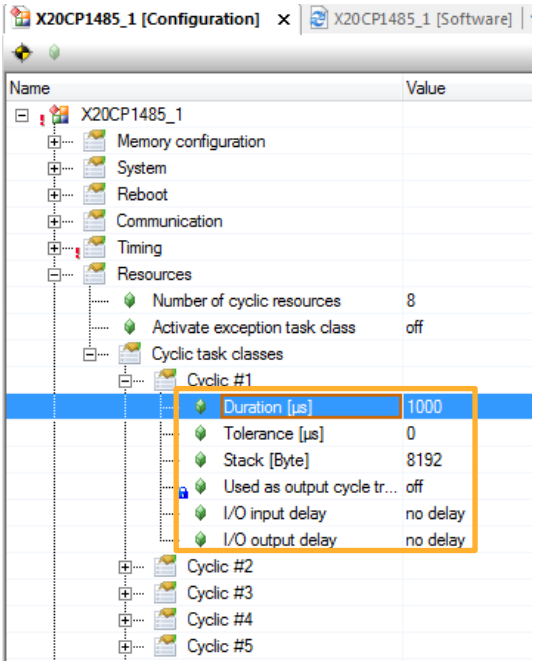


The cycle time is set to 1000µs. Automation Studio may ask you to choose a multiple of 400 µs. If so, choose 1200µs.

Physical View -> double click on the B&R PLC(here *X20CP1485-1*) -> right click *Cyclic #1 -> Properties*.

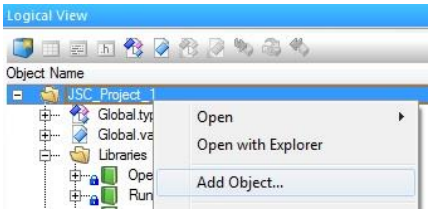
Set the duration for the virtual Axis in the Cyclic#.
The Duration must be the same as the Powerlink Cycle Time which was set before.
The "Tolerance" must be 0ms and "*I/O outputs delay*" has to be set to „*no delay*

Note: If these settings aren't made like explained, the initialisation will be stopped by an error and a commissioning of the virtual axis won't be possible.

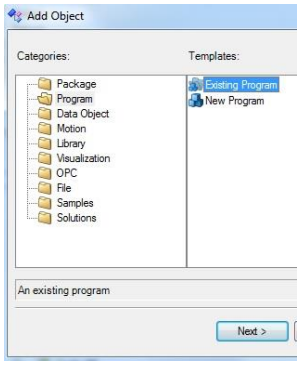


5.5 Insert Program Example

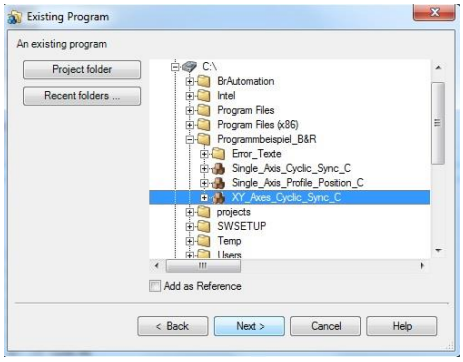
Logical View -> JSC_Project_1 -> right click-> Add Object…



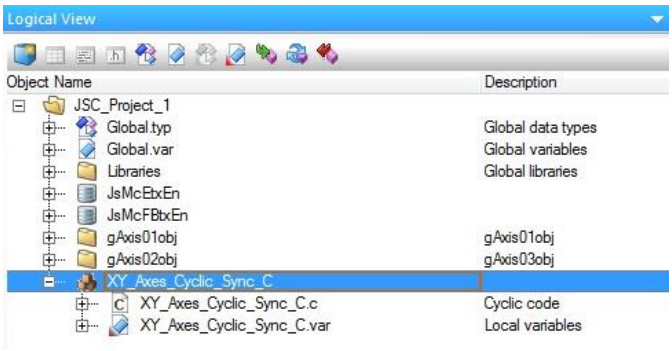
Categories: program ->Existing program -> Next



Select the path of the *"XY_Axes_Cyclic_Sync_C"*. *-> Next*



The program can be seen now in the Logic*al View*.



5.6 Configure Ethernet Interface

With double click on „XY_Axes_Cyclic_Sync_C.c“, the program code will be opened.
Change the Ethernet Powerlink Interface Address (IF3) and node number according to your setup.

Ethernet Powerlink Address EPL:
Adapt the line `instJS_MC_Init.pDevice = (UDINT)"IF4";`. The Powerlink address is for both XENAX® Servo Controller the same.

```
instJS_MC_Init_1.pDevice = (UDINT)"IF4";
instJS_MC_Init_2.pDevice = (UDINT)"IF4";
```

Node Nr.
The node number is defined in the line `instJS_MC_Init.Node= 1` (here 1 & 2). The node number has to be the same as the CI (Card Identifier) of the XENAX® Servo Controller. The CI is explained above.

```
/* initialize node number */
instJS_MC_Init_1.Node = 1;
instJS_MC_Init_2.Node = 2;
```

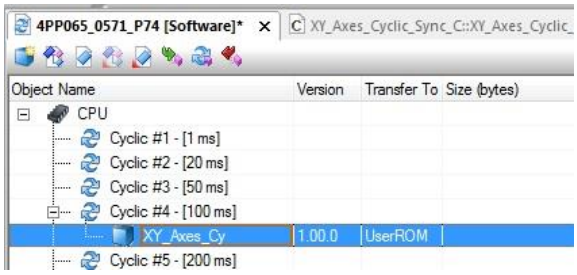
Coupling the virtual Axis
In the row: `Virtual_Axis_Ref_1 = (int) & (gAxis01);` und `instJS_MC_MoveCyclic_1.Axis = Axis_Ref_1;` the axis names (gAxis01 & gAxis02) are visible. The name in the program code hase to correspond the axis name to succesfully couple the virtual axis.

```
  /* Reference to the virtual ADC axis */
  Virtual_Axis_Ref_1 = (int)&(gAxis01);
  Virtual_Axis_Ref_2 = (int)&(gAxis02);
/* transfer position from virtual axis to the real XENAX® axis */
instJS_MC_MoveCyclic_1.Axis             = Axis_Ref_1;
instJS_MC_MoveCyclic_2.Axis             = Axis_Ref_2;
if(bFIRSTRUN == bTRUE)
{
    instJS_MC_MoveCyclic_1.Position     = gAxis01.monitor.s;
    instJS_MC_MoveCyclic_2.Position     = gAxis02.monitor.s;
}
```

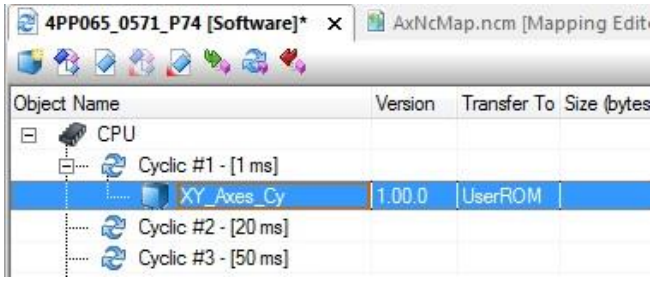
5.7 Move Program Example to Task with Desired Cycle Time

Under Physical *View -> B&R PLC -> double click ->* Software Configuration

The program example is always in Cyclic #4(100ms) as default.



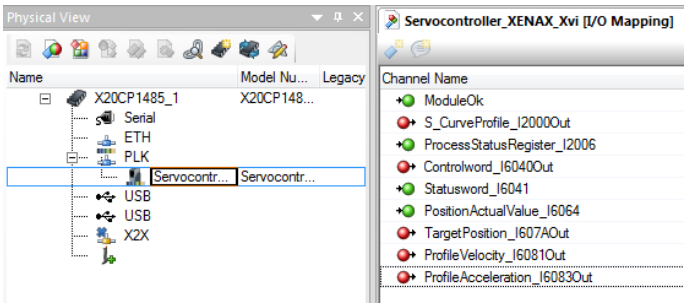
We move the program example to the „Cyclic #1“, which has a shorter cycle time (1ms).



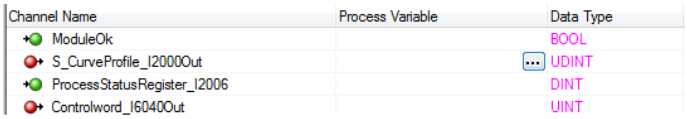
5.8 I/O Mapping, connect the .xdd Channels with the program variables.

To use the .xdd-Channels (cyclic data of the XENAX®) in the library they have to be connected to a “Process Variable”.

Physical View, double click on XENAX® Servo Controller -> *I/O Mapping*.



Select in the row „Process Variable“ the cell of the desired Channel (e.g.: S_CurveProfile_I2000Out). Click on the ... Symbol.

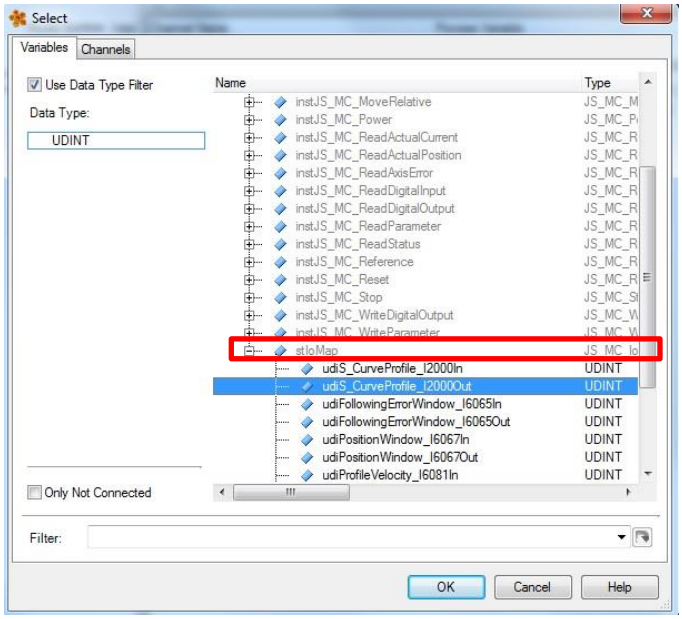


The Window “Select” will be opened

Please open the according program with the ⊞ symbol.

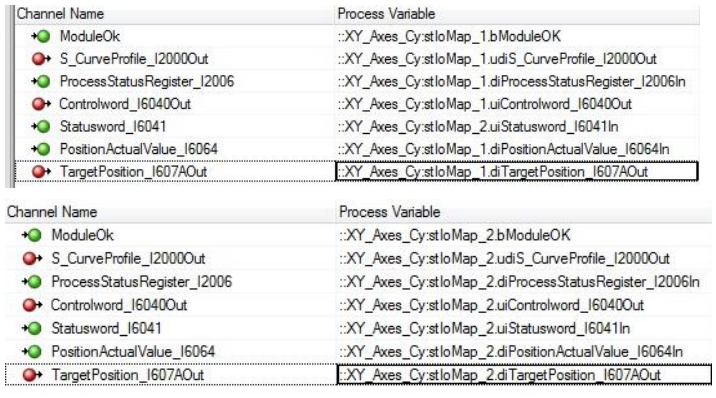
In the tree layout of ⊟ „stIoMap“ there are all “Process Variables” which have a data type that matches the channel.
Select the corresponding variables for the channel.

-> OK



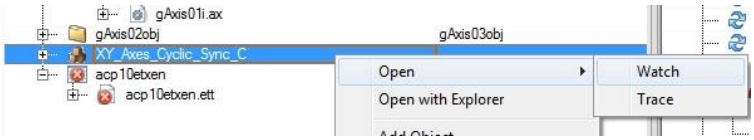
After the process variables are selected, the path can be seen in the “I/O Mapping”.

Every channel needs to be connected to a process variable.
This has to be done for every XENAX®

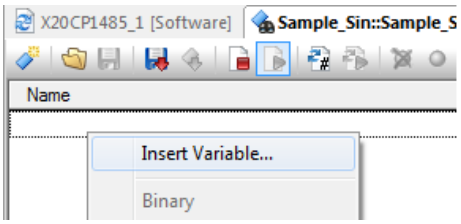


5.9 Start Program Example

Logial View -> double click on the B&R PLC (here X20CP1485-1) -> right click on the program example *„XY_Axes_Cyclic_Sync_C“-> Open -> Watch*



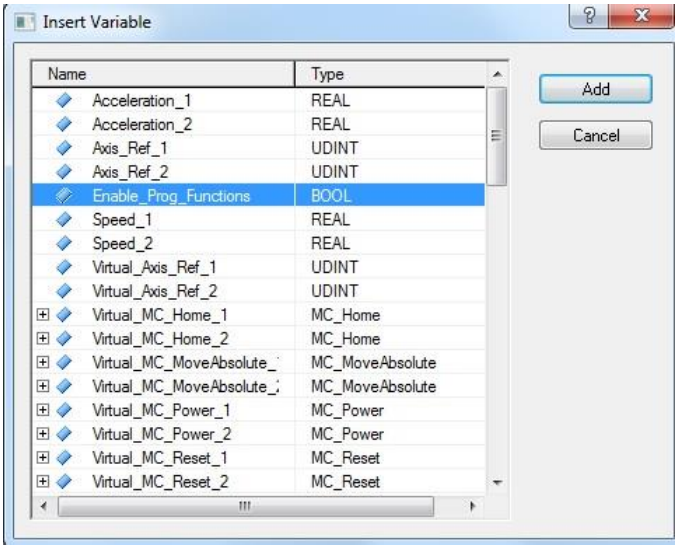
An empty window will be opened, right click -> *Insert Variable…*



The *„Insert Variable“* window is opened with all the function blocks and variables of the program example.

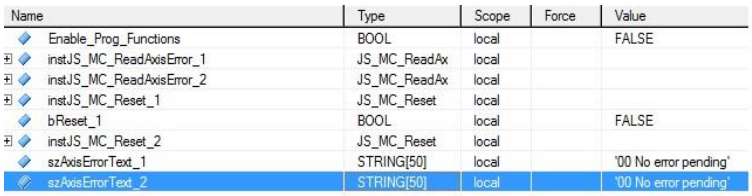
Select the required variables and function blocks. You will need at least the following:
Enable_Prog_Functions
instJS_MC_ReadAxisError_1
instJS_MC_ReadAxisError_2
instJS_MC_Reset_1
instJS_MC_Reset_2
szAxisErrorText_1
szAxisErrorText_2
bReset_1

->Add



Set the “Enable_Prog_Functions” to TRUE to start the program. With FALSE the program is idling in a loop.

A Error can be quit by setting the bReset_1 to TRUE.



Name	Type	Scope	Force	Value
Enable_Prog_Functions	BOOL	local		FALSE
instJS_MC_ReadAxisError_1	JS_MC_ReadAx	local		
instJS_MC_ReadAxisError_2	JS_MC_ReadAx	local		
instJS_MC_Reset_1	JS_MC_Reset	local		
bReset_1	BOOL	local		FALSE
instJS_MC_Reset_2	JS_MC_Reset	local		
szAxisErrorText_1	STRING[50]	local		'00 No error pending'
szAxisErrorText_2	STRING[50]	local		'00 No error pending'

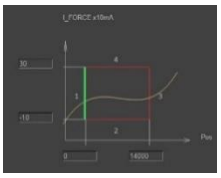
Note:
In the help of the library there is a program example with 1 Axis in Cyclic Synchronized mode.

6 Program example Forceteq®

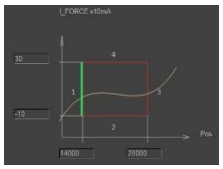
This demo project includes full Forceteq® functionality such as Force Calibration, Force Limitation and Force Monitoring.

The Forceteq® example alternatingly calls a Move Absolute (Position 0 or 44000). During the move from position 0 to 44000 the Force Monitoring with 3 sectors is activated and I_ForceLimit is set to 200mA.

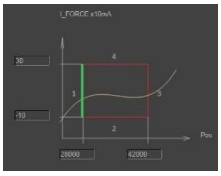
Definition of the 3 sectors:



****** Sector I_Force 1 **********
Sector IForce Start = 0
Sector IForce End = 14000
IForce Low x10mA = -10
IForce High x10mA = 30
Sector Transit Config = 4096



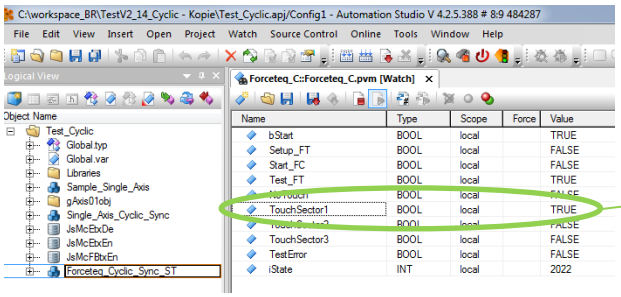
****** Sector I_Force 2 **********
Sector IForce Start = 14000
Sector IForce End = 28000
IForce Low x10mA = -10
IForce High x10mA = 30
Sector Transit Config = 4096

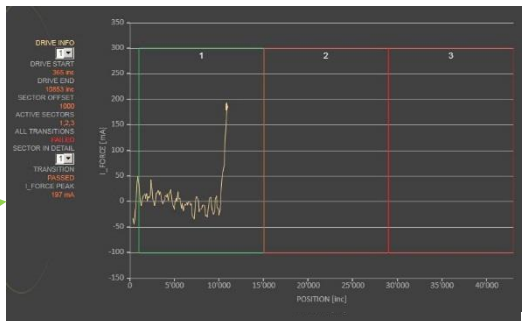


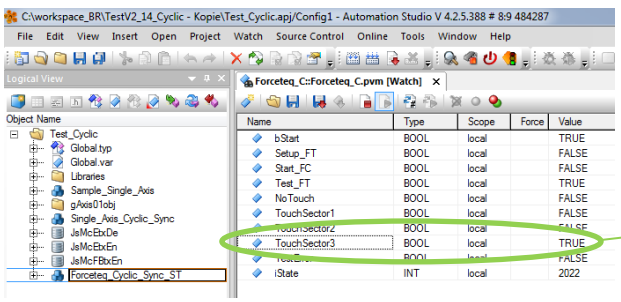
****** Sector I_Force 3 **********
Sector IForce Start = 28000
Sector IForce End = 42000
IForce Low x10mA = -10
IForce High x10mA = 30
Sector Transit Config = 4096

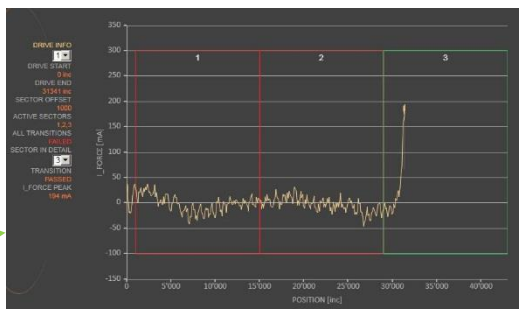
If there is a touching in one of these sectors the move is stopped (see the examples in Sector 1 and Sector 3 below) and a fast backward move to position 0 is started:

Fixed sector offset is set to the 1000



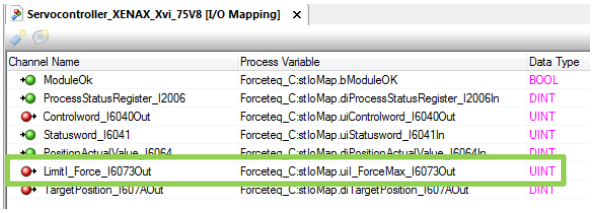






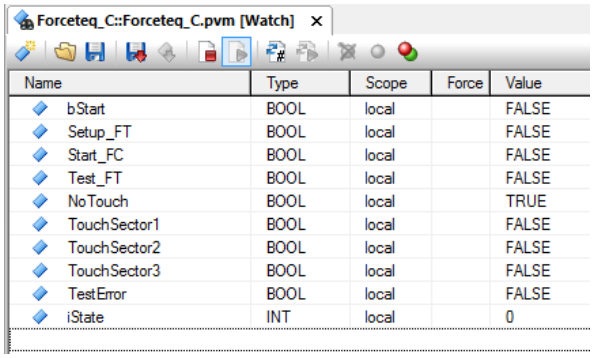
The Forceteq® example controls a linear motor in Cyclic Synchronous Position Mode (Forceteq_Cyclic_Sync_ST) or in Profile Position Mode (Forceteq_Profile_Position_ST) and alternatingly calls a Move Absolute (Position 0 or 40000). Both examples a written in structered text (ST) and you have to use all setups from chapter 5 (Profile Position Mode) or chapter 6 (cyclic Synchronous Position Mode).

Then also include the parameter LimitI_Force_I6073Out to your PDO configuration and link it to the variable stIoMap.uiI_ForceMax_I6073Out.



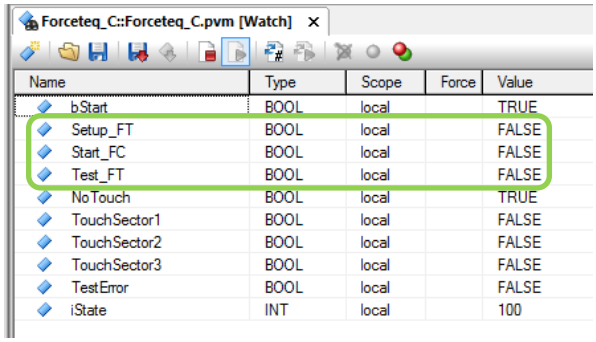
To start and control the example program, add the following variables to the Watch window:

- bStart
- Setup_FT
- Start_FC
- Test_FT
- NoTouch
- TouchSector1
- TouchSector2
- TouchSector3
- TestError
- iState



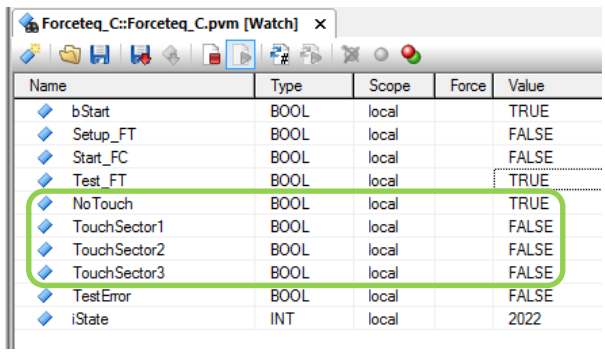
Start the program to set variable bStart to TRUE in “Value” and then save it to the PLC with press the “Enter” key.

Before testing set the sector values by activating the bit Setup_FT once and start the Force Calibration by activating the bit Start_FC.



To start the test activate the bit Test_FT. As long as the bit Test_FT is activated, the test will continue.

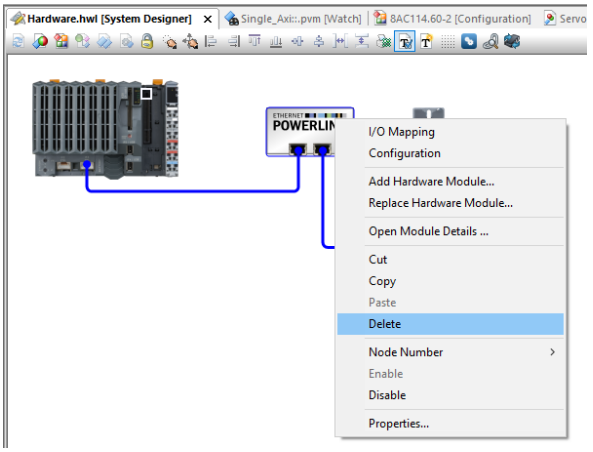
The result of each test is shown in the outputs TouchSector1, TouchSector2, TestSector3 and NoTouch.



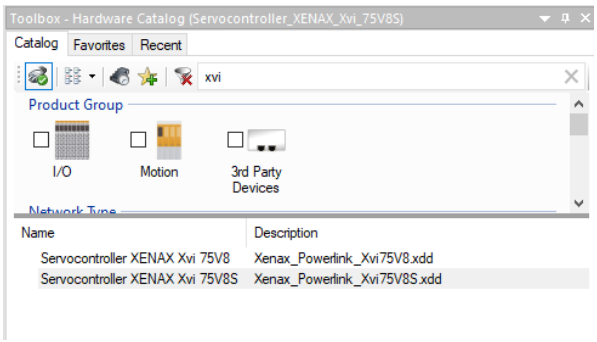
7 Upgrade from XENAX® 75V8 to 75V8S

The XENAX® **75V8S** is a replacement for the older **75V8**. It is meant as a one to one replacement. However, they have a different product ID. This means that the project must be reconfigured for the new product ID. Follow the steps this chapter to replace a XENAX.

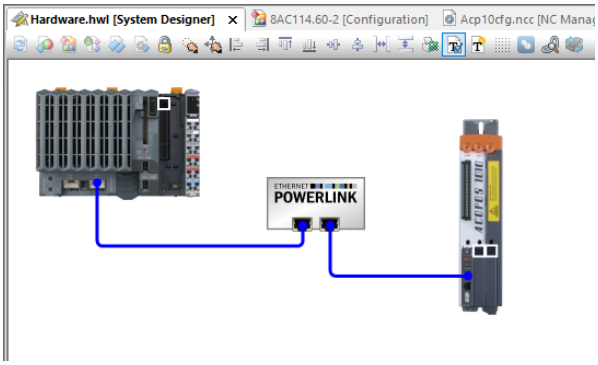
Delete the old XENAX axis in the System Designer..
Note: This will also reset the card identifier and the IO Configuration. Make a backup before deleting.



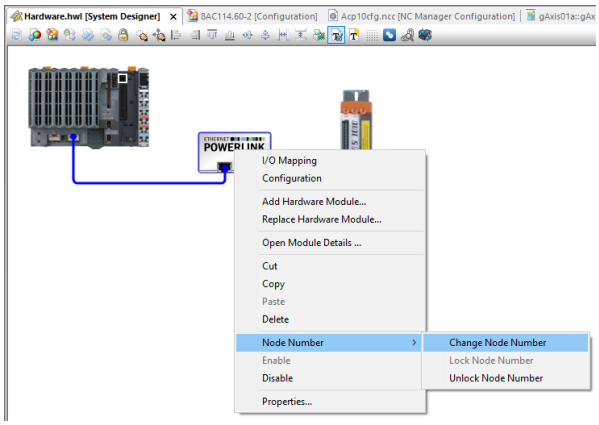
Insert the new XENAX Xvi 75V8S axis into the System Designer by Drag and Drop.



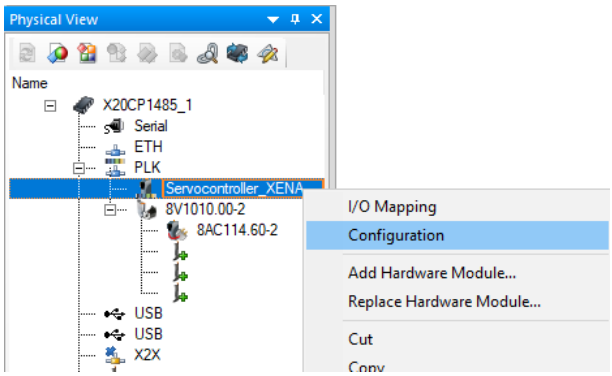
Reconnect the axis with the Powerlink Bus.



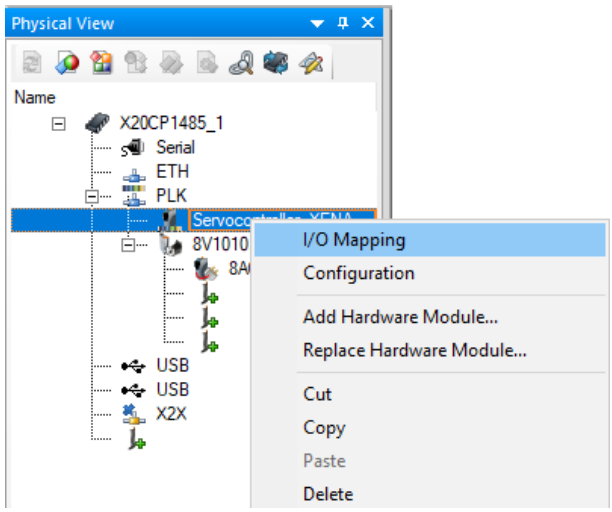
Change the node number (card identifier) as it was before.



Reconfigure the Channels of the XENAX axis.



Remap the channels as they were before.



Notes

Jenny Science AG
Sandblatte 7a
CH-6026 Rain, Schweiz

Tel +41 (0) 41 455 44 55
Fax +41 (0) 41 455 44 50

www.jennyscience.ch
info@jennyscience.ch