APPLICATION NOTE

CONVERTING AN AC500 V2 PROJECT TO AN AC500 V3 PROJECT

GUIDES, HINTS AND TIPS

Contents

1 Introduction ... 5
1.1 Scope of the document ... 5
1.2 Compatibility ... 6

2 Main changes from AC500 V2 to AC500 V3 ... 7
2.1 Hardware and system overview ... 7
2.1.1 Hardware and Operating System ... 7
2.1.2 Memory Sizes ... 8
2.1.3 Input- / output addresses ... 9
2.1.4 Memory allocation / byte order ... 9
2.1.5 Addressable variables ... 10
2.1.6 Modbus addresses / Byte order ... 11
2.1.7 Interfaces and slots ... 13
2.2 Automation Builder V3 Editor ... 14
2.2.1 Configuration and programming in the same tool ... 14
2.2.2 Devices and POUs tree ... 14
2.3 Library management and versioning ... 15
2.4 Visualization ... 16
2.5 Diagnosis ... 16
2.5.1 Online diagnosis within Automation Builder ... 16
2.5.2 Diagnosis in IEC applications ... 18
2.6 Security ... 19
2.7 User Management ... 20
2.8 Additional AC500 V3 capabilities ... 21
2.8.1 Object oriented programming (OOP) ... 21
2.8.2 Static code analysis (SCA) ... 21
2.8.3 Subversion control (SVN) ... 21
2.9 Documentation of POUs and libraries ... 21
2.9.1 Documentation of the POUs inside the library manager ... 21
2.9.2 System technology about special libraries ... 23
2.9.3 Online Web Library's Documentation ... 23

3 Hardware, protocols and libraries availability ... 25
3.1 Hardware ... 25
3.1.1 CPU overview ... 25
3.1.2 CPUs detailed HW feature lists ... 26
3.1.3 Safety CPUs ... 34
3.1.4 AC31-Replacement ... 34
3.1.5 Hardware Interfaces ... 34
3.1.6 Fieldbus communication modules (CM) ... 34
3.1.7 Communication Interface modules ... 35
3.1.8 Other devices ... 35
3.1.9 Memory Card ... 36
3.2 Protocols ... 36
3.2.1 Serial protocols ... 36
3.2.2 Ethernet based protocols ... 37
3.2.3 SMTP protocol ... 38
3.3 Libraries ... 39
3.3.1 AC500 System libraries ... 39
3.3.2 Communication libraries ... 40
3.3.3 Safety libraries ... 41
3.3.4 Application / Product libraries ... 41
3.3.5 CAA libraries ... 42

3.3.6 System libraries ... 43
3.3.7 CODESYS libraries ... 44
3.3.8 Application libraries ... 44
3.3.9 C-code libraries ... 44

4 Conversion of custom libraries ... 45
4.1 Upgrade to V3.5 ... 45
4.2 Changes in POUs, Libraries, … ... 48
4.3 Install and share the custom V3 library ... 49

5 Conversion of the project ... 51
5.1 Typical adaptions in the Hardware configuration ... 52
5.1.1 IOs ... 52
5.1.2 Alarms, Traces & Recipes ... 52
5.1.3 Library Manager ... 52
5.1.4 SNTP ... 53
5.2 Typical adaptions in the IEC application ... 56
5.2.1 PLCopen conform function blocks ... 56
5.2.2 Modbus addresses %M and %R ... 56
5.2.3 ACS/DCS Drives Library ... 57
5.3 Typical migration for 3rd party devices ... 58

6 Reestablishing communication to field devices ... 59
6.1 Modbus TCP ... 59
6.1.1 AC500 V2 ... 59
6.1.2 AC500 V3 ... 60
6.1.3 Summary of steps for migration from AC500 V2 to AC500 V3 ... 62
6.2 Modbus RTU ... 63
6.2.1 Modbus RTU HW configuration ... 63
6.2.2 Modbus RTU IEC program adaption ... 65
6.3 Profinet ... 67
6.3.1 General ... 67
6.3.2 DriveManager ... 67
6.3.3 Bus cycle ... 68
6.3.4 Cyclic data exchange ... 69
6.3.5 Acyclic data exchange ... 69
6.3.6 Bus control ... 69
6.3.7 Diagnosis ... 70
6.4 Profibus ... 72
6.4.1 General ... 72
6.4.2 DriveManager ... 72
6.4.3 Bus cycle ... 73
6.4.4 Cyclic data exchange ... 74
6.4.5 Acyclic data exchange ... 74
6.4.6 Bus control ... 75
6.4.7 Diagnosis ... 76
6.5 CS31 Bus connections ... 76

7 Reestablishing communication to SCADA devices ... 78
7.1 Symbol File ... 78
7.1.1 AC500 V2 PLC ... 78
7.1.2 AC500 V3 PLC ... 81
7.2 CP600 ... 83
7.2.1 Renaming tags e.g. CoDeSys V2 ETH to CODESYS V3 ETH protocol ... 83

7.2.2 Replacing tags e.g. ABB Modbus TCP V2 to ABB Modbus TCP V3 ... 88
7.3 OPC DA ... 89
7.3.1 OPC DA for AC500 V2 ... 90
7.3.2 OPC DA for AC500 V3 ... 94
7.3.3 Update the project from AC500 V2 to AC500 V3 ... 98
7.4 UDP ... 99
7.5 MQTT & JSON ... 99

1 Introduction

1.1 Scope of the document

This document guides a user how to upgrade an existing AC500 V2 project to the AC500 V3 PLC generation. This document is split into the following chapters:

- **Main changes from AC500 V2 to AC500 V3**

  Overview of main changes between the AC500 V2 and AC500 V3 platforms. Most of these changes are concerning the hardware, e.g. different interfaces, different memory layout, different handling of addresses, etc.

- **Hardware, protocols and libraries availability**

  The scope of communication modules, fieldbus protocols, libraries, etc. that are supported by AC500 V3 is constantly increasing. This chapter shall provide an overview on what is currently available, so it is easy to decide, if a certain application can be fully converted to AC500 V3.

  In case something is missing in the AC500 V3 PLC, please get in touch with us, probably it is already on the roadmap.

- **Conversion of custom libraries**

  Many AC500 V2 projects are using custom libraries, which could be either own developed libraries or libraries from other sources. The recommended first step before migrating the project is to either migrate the required AC500 V2 libraries (source code has to be available) or to check for AC500 V3 compatible versions of the libraries.

- **Conversion of the project**

  Automated conversion of a project from AC500 V2 to AC500 V3 typically leads to some errors. In this chapter typical errors and recommended fixes are listed.

- **Reestablishing communication to field devices**

  In most cases some dedicated manual adaptions have to be done for re-establishing communication to field devices. In this chapter typical errors and recommended adaptations to fix the errors are listed.

- **Reestablishing communication to SCADA devices**

  In some cases dedicated manual adaptions have to be done for re-establishing communication to SCADA devices. In this chapter typical errors and recommended adaptations to fix the errors are listed.

  When migrating from AC500 V2 to AC500 V3, security aspects should also be considered, e.g. by introduction of user management or by changing OPC DA communication to OPC UA communication.

1.2 Compatibility

The descriptions and screenshots in this document are based on an Automation Builder 2.5 installation.

In earlier versions some features described in this document are not available.

In newer versions the described steps might be partly different.

Automation Builder V2.6

Automation Builder versions can be installed side-by-side starting from version 2.6.0. The different versions are installed in separate installation folders.

For technical reasons there are some shared components across the different Automation Builder versions. Even despite the side-by-side installation of Automation Builder versions, these components will continue to be shared. Examples of shared components are AC500 V2 libraries, CODESYS V2 engineering and gateway, Panel Builder or Drive composer pro.

Limitations of Automation Builder 64bit:

- Library migration from AC500 V2 to AC500 V3 can only be triggered with an Automation Builder 32bit but not in the 64bit version
- Drive Manager for AC500 V2 is not available with Automation Builder 64bit

2 Main changes from AC500 V2 to AC500 V3

2.1 Hardware and system overview

This chapter contains a general overview of hardware capabilities of AC50 V3 compared to AC500 V2.

2.1.1 Hardware and Operating System

	AC500 V2	AC500 V3
Processor type	Power PC	TI ARM
Processor speed	Up to 400 MHz	Up to 1 GHz
Memory	Up to • 4MB Code • 4MB Data • 32k %I • 32k %Q • 2MB Config	Up to 160 MB (Code, data, %I area, %Q area, configuration, IEC drivers)
M(odbus) memory	Up to 8 Segments with 64k each	128kB
Persistent memory	Up to 8 Segments %R with 64k each	128kB for persistent retain variables
Operating System	SMX	Linux
IO Bus	Up to 10 S500 devices	Up to 10 S500 devices
Ethernet	0-2 onboard Ethernet ports	1 or 2 onboard Ethernet ports

2.1.2 Memory Sizes

2.1.2.1 AC500 V2

	PM5x4	PM5x6	PM573	PM583	PM585	PM590	PM591	PM592	PM595
User Prog. Memory	128kB	512kB	512kB	1 MB	1 MB	2 MB	4 MB	4 MB	16 MB
User Data	10kB	64kB	224kB	736kB	1536kB	1536kB	4096kB	4096kB	16 MB
%M	2kB	64kB	128kB	128kB	512B	512kB	512kB	512kB	1024kB
VAR-Retain	1kB	1kB	32kB	32kB	512B	512kB	512kB	512kB	1024kB
%R	1kB	1kB	128kB	128kB	512B	512kB	512kB	512kB	1024kB
Total Variables	14kB	130kB	512kB	1024kB	3072kB	3072kB	5632kB	5632kB	19456kB
SRAM Disk	-	-	32 kB	64 kB	256 kB	256 kB	256 kB	256 kB	960 kB
Flash	1024kB	1512kB	3072kB	4096kB	12288kB	12288kB	16384kB	16384kB	40960kB
User RAM Disk	912kB	1424kB	1424kB	4096kB	4096kB	8 MB	8 MB	8 MB	32 MB
Flash Disk								4 GB	4 GB

2.1.2.2 AC500 V3

	PM5012	PM5032	PM5052	PM5072	PM5630	PM5650	PM5670	PM5675
User Prog. Memory User Data	256 kB	512 kB	768 kB	1 MB	2 MB	8 MB	32 MB	32 MB
Max. Size Download (Code, Data, Conf., WebVisu, Symbols)	1 MB	5 MB	7 MB	9 MB	9 MB	84 MB	176 MB	176 MB
%M	4 kB	16 kB	16 kB	64 kB	128 kB	128 kB	512 kB	512 kB
VAR-Retain	4 kB	16 kB	16 kB	36 kB	128 kB	128 kB	1024 kB	1024 kB
User Flash	30 MB	30 MB	30 MB	30 MB	30 MB	285 MB	643 MB	643 MB
Total Flash	128 MB	128 MB	128 MB	128 MB	128 MB	512 MB	1024 MB	1024 MB
Flash Disk								8 GB

2.1.3 Input- / output addresses

2.1.3.1 AC500 V2

IEC-Variable	Belegung
%IB0 - %IB999	I/O-Bus
%IB1000 - %IB1999	COM1 (CS31-Bus)
%IB2000 - %IB2999	COM2
%IB3000 - %IB3999	FBP-Interface
%IB4000 - %IB4095	Onboard-I/O (AC500-eCo)
%IB0.0 - %IB0.4095	Slot 0 (Interner CM module)
%IB1.0 - %IB1.4095	Slot 1 (Externer CM module1)
%IB2.0 - %IB2.4095	Slot 2 (Externer CM module2)
%IB3.0 - %IB3.4095	Slot 3 (Externer CM module 3)
%IB4.0 - %IB4.4095	Slot 4 (Externer CM module4)

All Inputs and Outputs are flat and Byte oriented

CM Modules: Inputs and Outputs with Slot Offset as Prefix

2.1.3.2 AC500 V3

One flat Address Range for

- IO-Bus
- Onboard CAN
- COM1
- CM Module

Will be filled without Gaps

2.1.4 Memory allocation / byte order

AC500 V2	AC500 V3
Big Endian (Motorola byte order)	Little Endian (Intel byte order)

2.1.4.1 AC500 V2

ADR	adr	adr+1	adr+2	adr+3
BOOL	%IX0.x	%IX1.x	%IX2.x	%IX3.x
BYTE	%IB0	%IB1	%IB2	%IB3
WORD	%IW0		%IW1	
DWORD	%ID0			

%IX0.0	:=TRUE	
%IB0	:=16#01	:=1
%IW0	:=16#0100	:=256 (Bit 8)
%ID0	:=16#01000000	:=16777216
%IX3.0	:=TRUE	
%IB3	:=16#01	:=1
%IW1	:=16#0001	:=1 (Bit 0)
%ID0	:=16#00000001	:=1

2.1.4.2 AC500 V3

ADR	adr	adr+1	adr+2	adr+3
BOOL	%IX3.x	%IX2.x	%IX1.x	%IX0.x
BYTE	%IB3	%IB2	%IB1	%IB0
WORD	%IW1		%IW0	
DWORD	%ID0			

%IX0.0	:=TRUE	
%IB0	:=16#01	:=1
%IW0	:=16#0100	:=1 (Bit 0)
%ID0	:=16#00000001	:=1 (Bit 0)
%IX3.0	:=TRUE	
%IB3	:=16#01	:=1
%IW1	:=16#0001	:=1 (Bit 8)
%ID0	:=16#01000000	:= 16777216

2.1.5 Addressable variables

2.1.5.1 AC500 V2

- %M- and %R- Area (4k - 512k depending on CPU type)

- Max 8 Segments with 64kB

Linie	IEC - Variable
0	%MB0.0 .. %MB0.65535 %RB0.0 .. %RB0.65535
1	%MB1.0 .. %MB1.65535 %RB0.0 .. %RB0.65535
..	.. ..
6	%MB6.0 .. %MB6.65535 %RB0.0 .. %RB0.65535
7	%MB7.0 .. %MB7.65535 %RB0.0 .. %RB0.65535

- Bits are Byte oriented

ADR	adr	adr+1	adr+2	adr+3
BOOL	%MX0.0. 0 .. 7	%MX0.1. 0 .. 7	%MX0.2. 0 .. 7	%MX0.3. 0 .. 7
BYTE	%MB0.0	%MB0.1	%MB0.2	%MB0.3
WORD	%MW0.0		%MW0.1	
DWORD	%MD0.0			

2.1.5.2 AC500 V3

- %M, one common Area (4-512 kB see chapter 2.1.2.2)
  - %MB0…%MB131071
- Bits are Byte oriented
- Persistent Option: %M without init (pragma: {no init}) to be set at each variable
- No %R Area, only VAR GLOBAL PERSISTENT RETAIN

Details for converting Memory ranges from V2 to V3 can be found in chapter 5.2.2.

2.1.6 Modbus addresses / Byte order

Due to the internal different memory layout, also the Modbus addressing has changed from AC500 V2 to AC500 V3. There are no more different segments, but only one flat memory area.

2.1.6.1 AC500 V2

Detailed information about AC500 V2 Modbus addresses is available from the online help:

PLC Automation with V2 CPUs → PLC integration (hardware) → System technology for AC500 V2 products → System technology of CPU and overall system → Communication with Modbus RTU → Modbus addresses for AC500 CPUs → Modbus address table

Address assignment (bit accesses)

The address assignment for bit accesses is done according to the following table:

Modbus address		Byte	Bit (byte-oriented)	Word	Double word
HEX	DEC	BYTE	BOOL	WORD	DWORD
Line 0					
0000	0	%MB0.0	%MX0.0.0	%MW0.0	%MD0.0
0001	1		%MX0.0.1		
0002	2		%MX0.0.2		
0003	3		%MX0.0.3		
0004	4		%MX0.0.4		
0005	5		%MX0.0.5		
0006	6		%MX0.0.6		
0007	7		%MX0.0.7		
0008	8	%MB0.1	%MX0.1.0		
001E	30		%MX0.3.6		
001F	31		%MX0.3.7		
0020	32	%MB0.4	%MX0.4.0	%MW0.2	%MD0.1
0021	33		%MX0.4.1		
0022	34		%MX0.4.2		
...	...	...	...	...	...
0FFF	4095	%MB0.511	%MX0.511.7	%MW0.255	%MD0.127
1000	4096	%MB0.512	%MX0.512.0	%MW0.256	%MD0.128
...	...	...	...	...	...
7FFF	32767	%MB0.4095	%MX0.4095.7	%MW0.2047	%MD0.1023
8000	32768	%MB0.4096	%MX0.4096.0	%MW0.2048	%MD0.1024
...	...	...	...	...	...
FFFF	65535	%MB0.8191	%MX0.8191.7	%MW0.4095	%MD0.2047

Calculation of the bit variable from the hexadecimal address:

Formula:			
	Bit variable (BOOL) := %MX0.BYTE.BIT		
where:	DEC	Decimal address	
	BYTE	DEC / 8	
	BIT	DEC mod 8	(Modulo division)

2.1.6.2 AC500 V3

Detailed information about AC500 V3 Modbus addresses is available from the online help:

PLC Automation with V3 CPUs → PLC integration (hardware) → System technology for AC500 V3 products → System technology of CPU and overall system → Communication with Modbus RTU → Modbus addresses for AC500 V3 processor modules PM56xx → Modbus address table

Table 622: Modbus addresses (word accesses)

Modbus address		Byte	Bit (byte-oriented)	Word	Double word
HEX	DEC	BYTE	BOOL	WORD	DWORD
0000	0	%MB0	%MX0.0 ... %MX0.7	%MW0	%MD0
		%MB1	%MX1.0 ... %MX1.7		
0001	1	%MB2	%MX2.0 ... %MX2.7	%MW1	
		%MB3	%MX3.0 ... %MX3.7		
0002	2	%MB4	%MX4.0 ... %MX4.7	%MW2	%MD1
		%MB5	%MX5.0 ... %MX5.7		
0003	3	%MB6	%MX6.0 ... %MX6.7	%MW3	
		%MB7	%MX7.0 ... %MX7.7		
...					
FFFE	65534	%MB131068	%MX131068.0 ... %MX131068.7	%MW65534	%MD32767
		%MB131069	%MX131069.0 ... %MX131069.7		
FFFF	65535	%MB131070	%MX131070.0 ... %MX131070.7	%MW65535	
		%MB131071	%MX131071.0 ... %MX131071.7		

Table 623: Address assignment (bit accesses)

Modbus address		Byte	Bit (byte-oriented)	Word	Double word
HEX	DEC	BYTE	BOOL	WORD	DWORD
0000	0	%MB0	%MX0.0	%MW0	%MD0
0001	1		%MX0.1		
0002	2		%MX0.2		
0003	3		%MX0.3		
0004	4		%MX0.4		
0005	5		%MX0.5		
0006	6		%MX0.6		
0007	7		%MX0.7		
0008	8	%MB1	%MX1.0		
0009	9		%MX1.1		
000A	10		%MX1.2		
000B	11		%MX1.3		
000C	12		%MX1.4		
000D	13		%MX1.5		
000E	14		%MX1.6		
000F	15		%MX1.7		
0010	16	%MB2	%MX2.0	%MW1	
0011	17		%MX2.1		
0012	18		%MX2.2		
0020	32	%MB4	%MX4.0	%MW2	%MD1
0021	33		%MX4.1		
0022	34		%MX4.2		
...	...	...	...	...	...
0FFF	4095	%MB511	%MX511.7	%MW255	%MD127
1000	4096	%MB512	%MX512.0	%MW256	%MD128
...	...	...	...	...	...
7FFF	32767	%MB4095	%MX4095.7	%MW2047	%MD1023
8000	32768	%MB4096	%MX4096.0	%MW2048	%MD1024
...	...	...	...	...	...
FFFF	65535	%MB8191	%MX8191.7	%MW4095	%MD2047

Calculation of the bit variable from the hexadecimal address:

Formula:			
	Bit variable (BOOL) := %MXBYTE.BIT		
where:	DEC	Decimal address	
	BYTE	DEC / 8	
	BIT	DEC mod 8	(Modulo division)

2.1.7 Interfaces and slots

AC500 V2	AC500 V3
0-2 Onboard Ethernet	1-2 Onboard Ethernet
2 COM Ports (incl. CS31-Bus Master at COM1)	1 COM Port
	1 CAN Onboard Port
1 FBP Port	
Memory card slot	(Micro) memory card slot
Battery Slot	Battery Slot
Up to 4 external communication modules	Up to 6 external communication modules

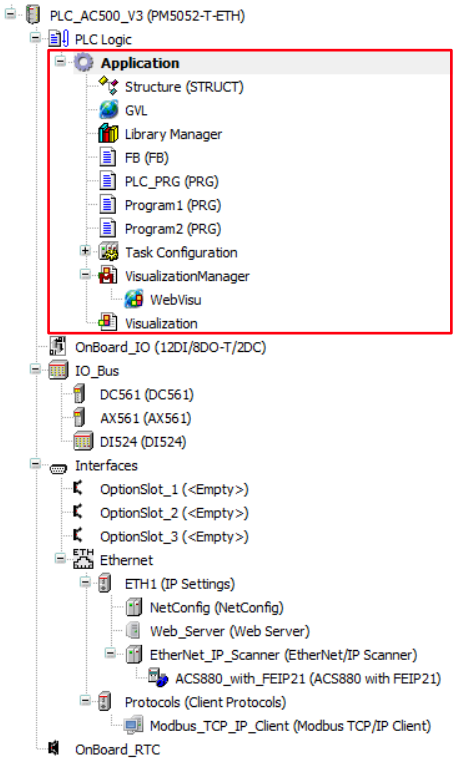
Slot number automatically configured with Automation builder	Device name will be handled by Automation Builder during inserting the communication module into the tree

2.2 Automation Builder V3 Editor

2.2.1 Configuration and programming in the same tool

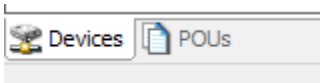
AC500 V2	AC500 V3
Configuration in Automation Builder	Configuration in Automation Builder
Programming in CODESYS	Programming in Automation Builder

In contrast to AC500 V2, where only the configuration is done in Automation Builder and the programming is done in CODESYS, the AC500 V3 PLCs are configured and programmed in Automation Builder.



2.2.2 Devices and POUs tree

In Automation Builder there is a device tree and a POU tree. In AC500 V2 only the device tree is used.



In the device tree several AC500 V2, AC500 V3 or other devices like Panels, Drive composer pro, … can be added. All devices have an own application containing variables, structures, programs, …

In the POU tree there can be also global variables, function blocks, structures, visualizations, libraries, … These POU objects can be used by all AC500 V3 PLCs in the device tree.

2.3 Library management and versioning

AC500 V2	AC500 V3
Only one library version for all Automation Builder Versions	Different library versions depending on Automation Builder Version. Multiple library versions within one Profile possible
Libraries stored in folders	Libraries installed to library repository
Library manager in CODESYS 2.3 Project	Library Manager in Automation Builder device tree and optional in the POU tree

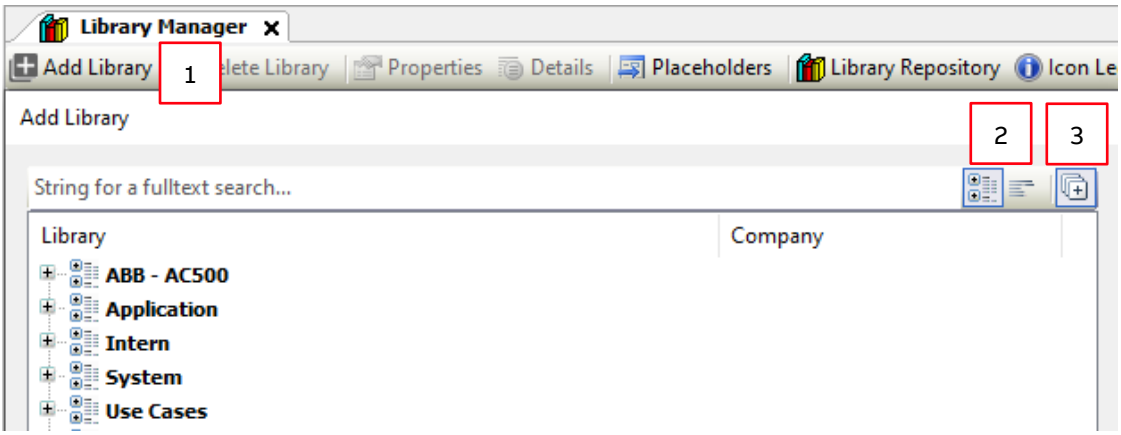
In AC500 V3 the library manager is in the device tree below the application and if needed inside the POU tree. For more details between the device tree and POU tree see chapter 2.8.

Necessary libraries required for the hardware functionality are automatically added to the library manager depending on the used hardware configuration in the device tree.

Additional libraries can be added from the *Library Manager* object by clicking on *Add Library* (1).

The libraries can be shown in a tree view or as list (2)

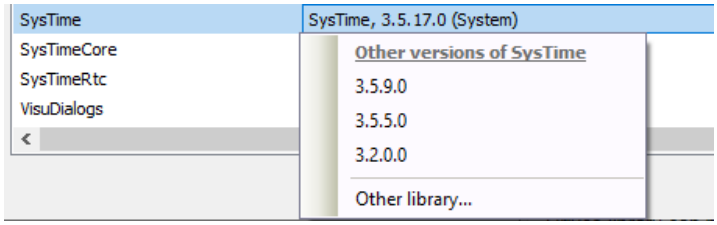
The System libraries are by default hidden. To make them visible click Display advanced libraries in the upper right corner. (3)



After selecting the library from the list, it can be added by clicking *OK*.

In contrast to V2 a library repository is handling all libraries and versions. To add an external library to a project it first needs to be installed in the library repository.

In AC500 V3 a versioning inside the library repository is possible. When adding a library to a project, the placeholder (e.g. AC500_Ethernet) is added. The device resolves this placeholder via the library repository resolves with the matching version of this library (e.g. Ethernet 1.3.0.7). It is also possible to change the used version by clicking Placeholders and double clicking the library which shall be changed. If multiple versions of this library are installed to the repository the version can be changed from the placeholder here.



It is not recommended to use older versions of a library, if multiple versions are installed.

2.4 Visualization

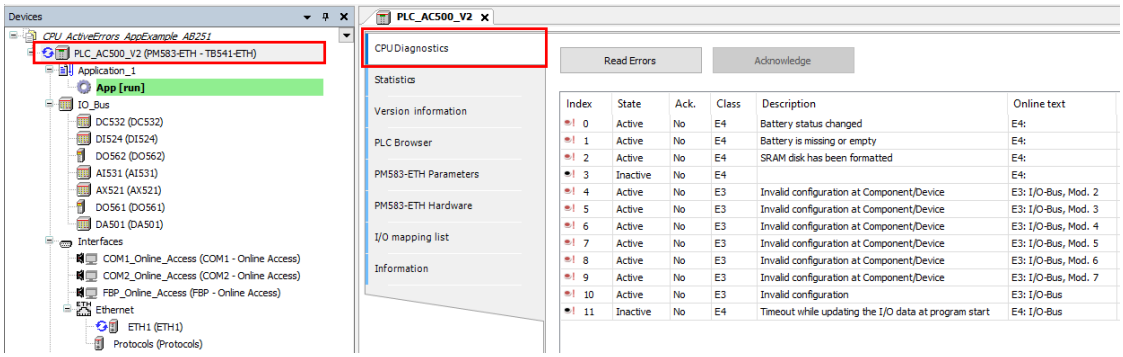
The AC500 V3 visualization provides more options in terms of configuration and widgets than AC500 V2. Main difference of the web visualization is that the generated visualization is plain HTML 5 and does not require Java Script. Detailed descriptions of the V3 features are available in the online help and the application example: AC500 V3 Web Visualization – Demonstration Example

2.5 Diagnosis

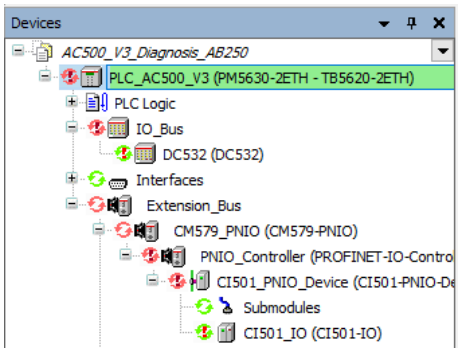
Diagnosis for commissioning or maintenance is provided either by Automation Builder or IEC code. While AC500 V2 only shows generic diagnosis information, AC500 V3 is able to receive device-, module- or channel specific diagnosis.

2.5.1 Online diagnosis within Automation Builder

Automation Builder diagnosis for AC500 V2 is only available as generic CPU diagnosis at the PLC node.



Compared to AC500 V2, the online view of Automation Builder for AC500 V3 PLCs provides first indication of faults and errors at the device tree in form of symbols in front of the faulty device and their parent objects.



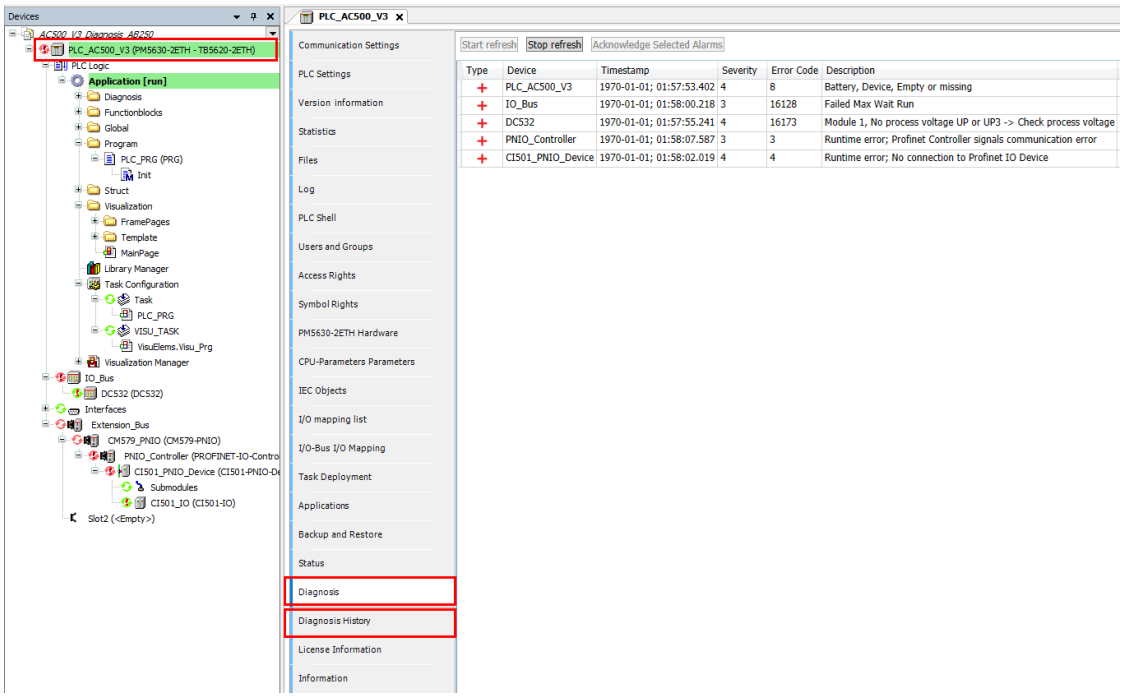
Note: Read more about *Project tree in online mode* for AC500 V3 in the Automation Builder online Help. For this, follow the content path:

PLC Automation with V3 CPUs > Diagnosis and debugging for AC500 V3 products > Online diagnosis in Automation Builder > Project tree in online mode

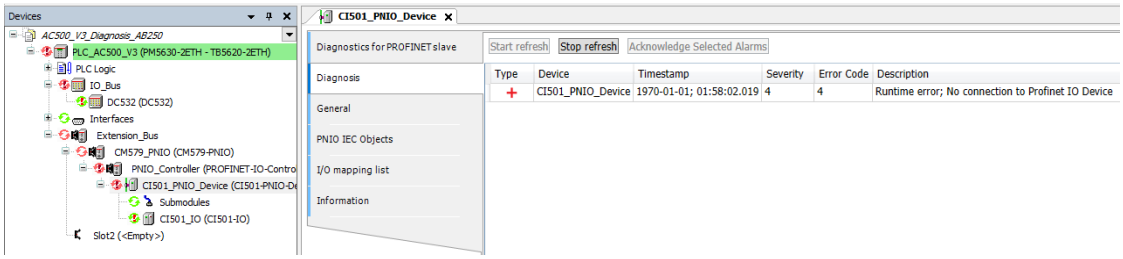
For AC500 V3 not only generic diagnosis, but also device and feature specific diagnosis is available inside Automation Builder during online mode.

The PLC provides a general overview of all diagnosis information inside the system. The tab “Diagnosis” shows only actual pending errors and faults and therefore has only incoming diagnosis data.

To see the history of incoming and outgoing diagnosis data, AC500 V3 has also a tab “Diagnosis History”.



Next to the overview of the PLC node, the diagnosis is also available at their specific device node.



Note: Read more about *Device diagnosis* and *Diagnosis history* for AC500 V3 in the Automation Builder online Help. For this, follow the content path:

	PLC Automation with V3 CPUs > Diagnosis and debugging for AC500 V3 products > The diagnosis system > Diagnosis in Automation Builder

2.5.2 Diagnosis in IEC applications

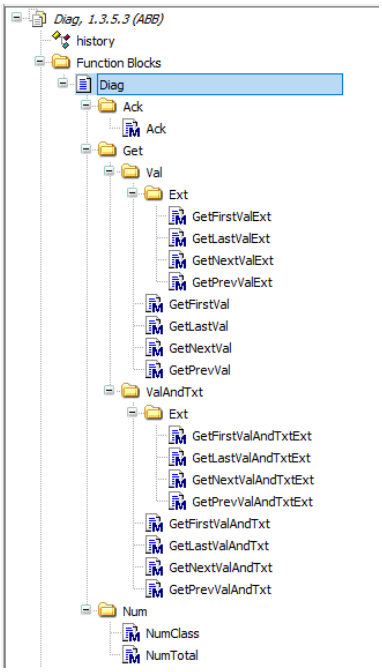
AC500 V2	AC500 V3
Generic diagnosis error codes	Device specific diagnosis error codes
-	Diagnosis available as text
-	Diagnosis history available for IEC applications
-	Editable text lists available to modify readable text
-	Multi language system available

To receive diagnosis data in IEC applications, a library is required for AC500 V2 as well as for AC500 V3:

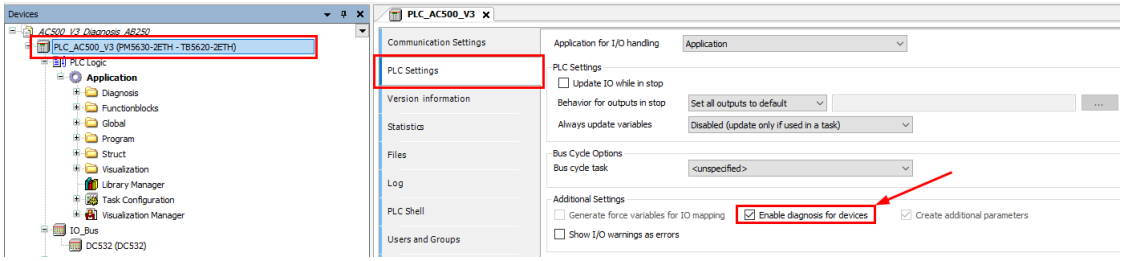
- **AC500 V2**: Diag_AC500_V20
- **AC500 V3**: AC500_Diag

  AC500_DiagHistory (not available for V2)

For V3 platform, the library uses methods to read out diagnosis data. The diagnosis of the complete system can be read with the function block *Diag* inside the library.

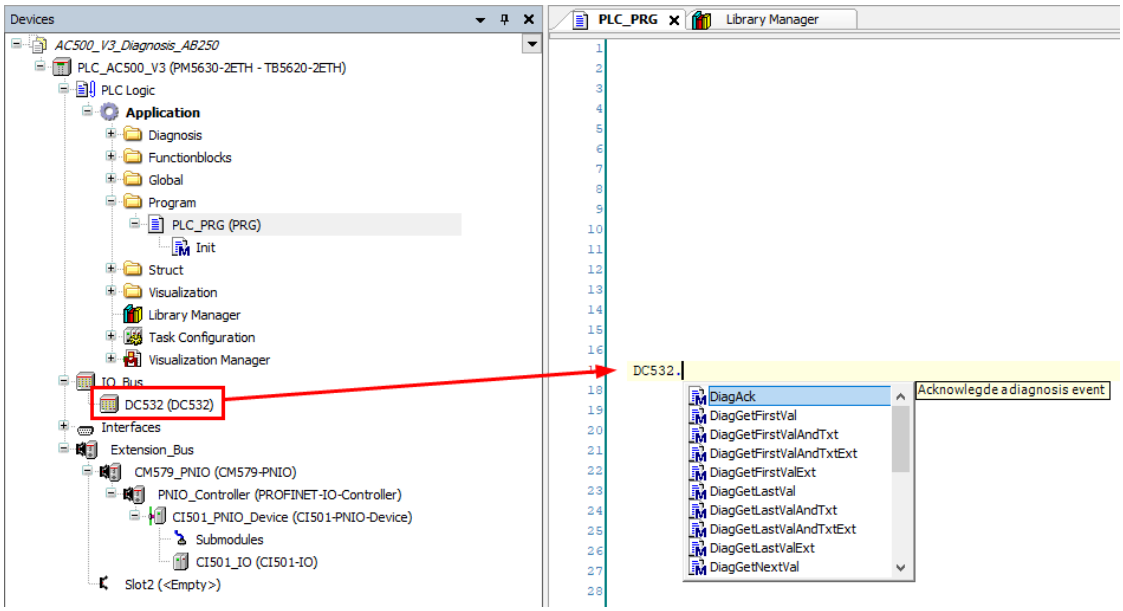


In order to receive diagnosis for specific devices, the checkbox *Enable diagnosis for devices* must be activated at the PLC Settings tab of the PLC node.



If this is done device specific diagnosis can be read, by accessing the methods for diagnosable device object inside the device tree.

For this, the object name of the device tree must be used as shown in the following picture.



	Note: Read more about *Diagnosis in IEC applications* for AC500 V3 in the Automation Builder online Help. For this, follow the content path: PLC Automation with V3 CPUs > Diagnosis and debugging for AC500 V3 products > The diagnosis system > Diagnosis in IEC application

	Note: An application example on how to use AC500 V3 diagnosis inside IEC applications can be found here. An application example for the diagnosis history can be found here.

2.6 Security

In general, please have a look at our whitepaper. This document provides a good overview about the topic Hardening and Defense in Depth strategy.

It is strongly recommended that no devices should be connected to the internet without additional security measures like encrypted communication or use of VPN. An application Note: Secure remote access via secomea gateway will explain one possibility.

2.7 User Management

	Note: Since the user managements in AC500 V2 and AC500 V3 are completely different. The user management cannot be upgraded and must be re-done in AC500 V3.

Automation Builder provides a feature to protect the project file with a password or certificate. Please have a look at the application note: AC500 USER MANAGEMENT WITH V3. This feature is available for AC500 V2 and AC500 V3 project files.

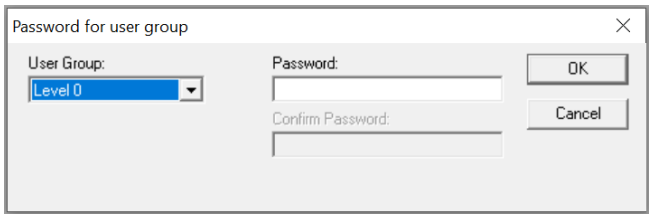
In addition, the AC500 V3 provides two further user managements for:

- PLC
- Visualization

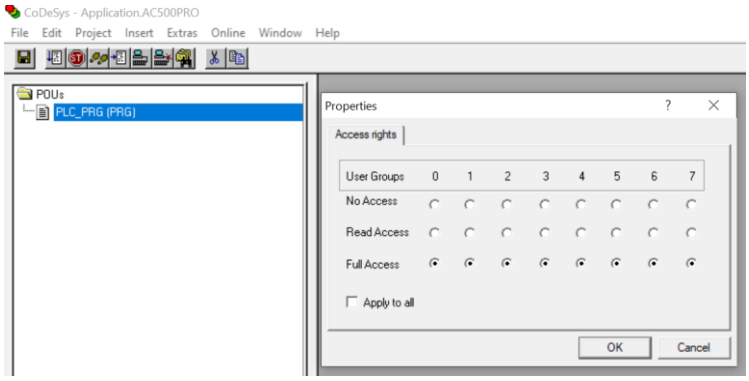
Further details can be found in the application note above or in the Automation Builder online help: PLC Automation with V3 CPUs > Programming with CODESYS > CODESYS Development System > Downloading an Application to the PLC > Handling of Device User Management

AC500 V2 PLC's user management can be accessed inside CODESYS under: Project → "User groups and passwords…". Here different passwords for different user groups (Level 0 … Level 7) can be set.

Level 0 is admin level.



Once the passwords are set, you can apply this user rights to different objects. For example, the PLC_PRG. Right click on PLC_PRG → Object Properties



The following screen will be shown. Here you can set access to different permissions for the user groups.

2.8 Additional AC500 V3 capabilities

When migrating a project from AV500 V2 to AC500 V3, it should be considered to combine that migration activity with a refactoring of the application to reduce potential technical debt.

Besides that the following AC500 V3 capabilities are available and should be considered to implement.

2.8.1 Object oriented programming (OOP)

Information is available in the online help. See: PLC Automation with V3 CPUs > Programming with CODESYS > CODESYS Development System > Programming of Applications > Object-Oriented Programming

Or in the Application Example AC500 V3 Object Oriented Programming - Working with Interfaces, Methods and Properties or in the Application Note AC500 V3 OOP Keywords

2.8.2 Static code analysis (SCA)

Information is available in the online help. See: PLC Automation with V3 CPUs > Engineering interfaces and tools > CODESYS Static Analysis

Or in the Application Example AC500 V3 Pro_Static_Analysis - Usage and Benefits For Code Optimization

2.8.3 Subversion control (SVN)

Information is available in the online help. See: PLC Automation with V3 CPUs > Engineering interfaces and tools > Professional Version Control

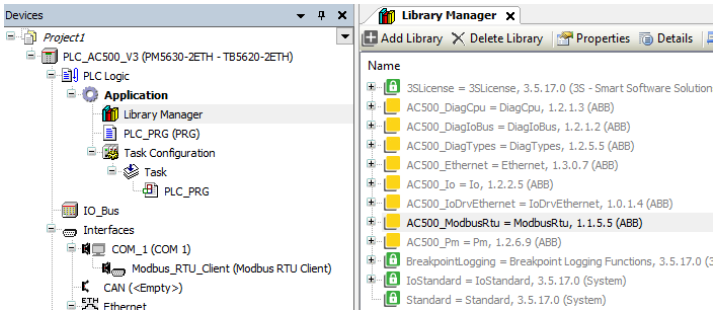
2.9 Documentation of POUs and libraries

2.9.1 Documentation of the POUs inside the library manager

The documentation of the functions, function blocks, structs etc (POU) for AC500 V3 are all included in the POUs and can be found directly in the library manager by opening the POU.

These informations are directly extracted from the POU source code, which ensures that they are always up to date.

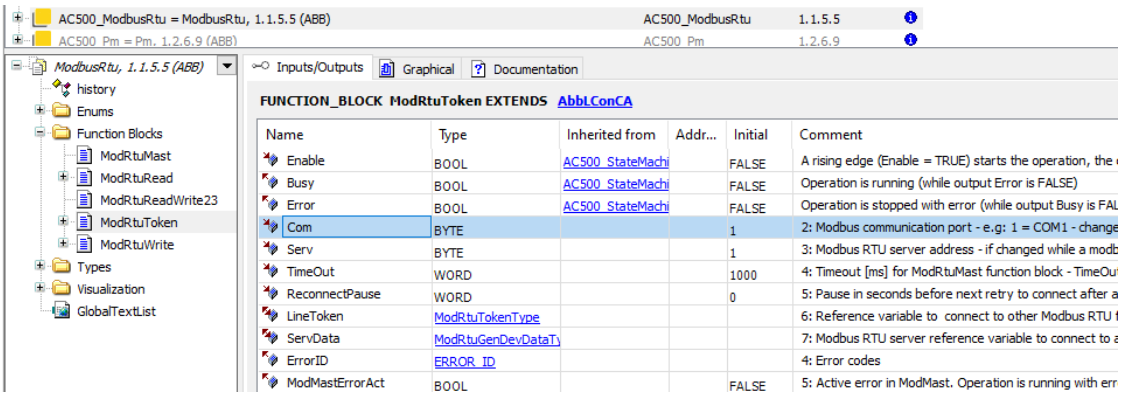
Library Manger > dedicated library >



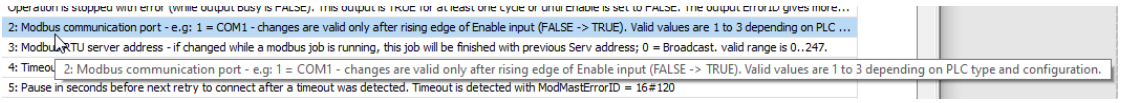
click on the POU > tabs on the right

2.9.1.1 Inputs/Outputs tab

All inputs and outputs are described with their type, inherited from, address, initial value and comment.
The inherited inputs and outputs are listed first. That means the order is not according to inputs than outputs.

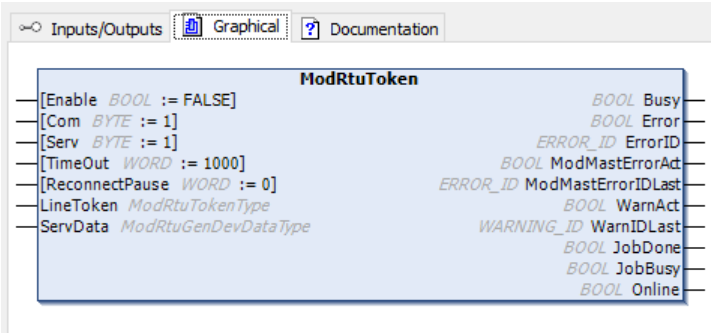


In case of longer comment a tooltip will show the whole comment.



2.9.1.2 Graphical tab

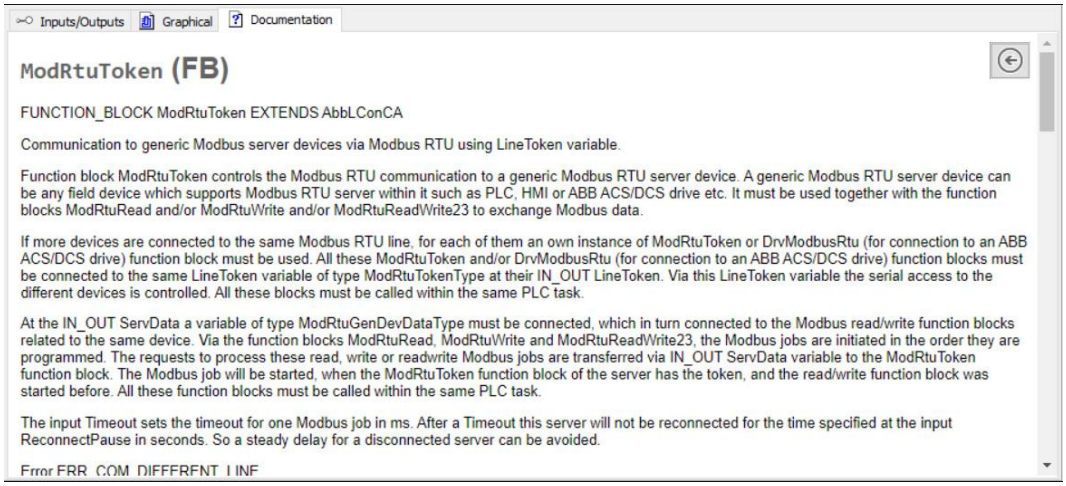
A graphical view of the POU shows all inputs, in_outputs and outputs including their type and default initial value.

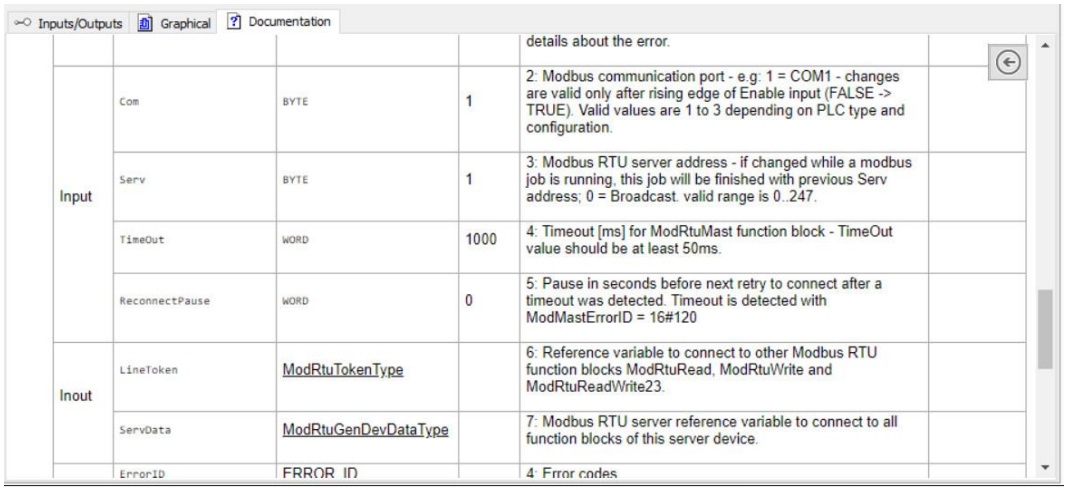


2.9.1.3 Documentation tab

In the documentation tab all comment of the declaration part of the POU is shwon.

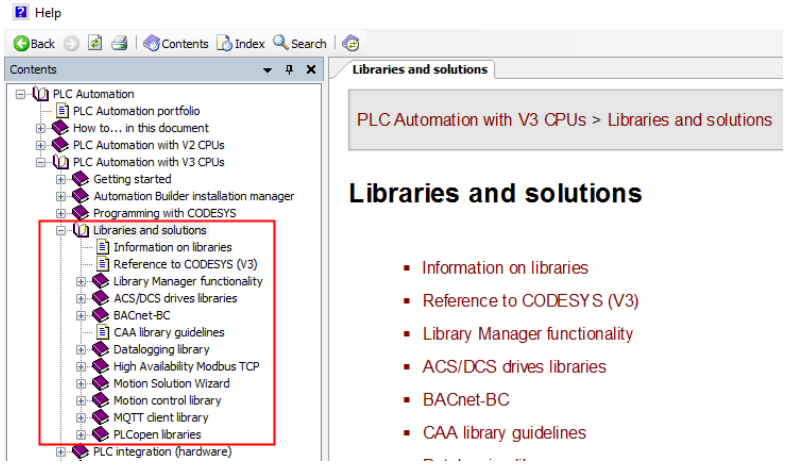
Depending on the POU, this can include a longer detailed description of how to use the POU, followed by the description of the inputs, in_outputs and outputs.





2.9.2 System technology about special libraries

For some libraries special system technology docmentation can be found in the Automation Builder help. See: PLC Automation with V3 CPUs > Libraries and solutions

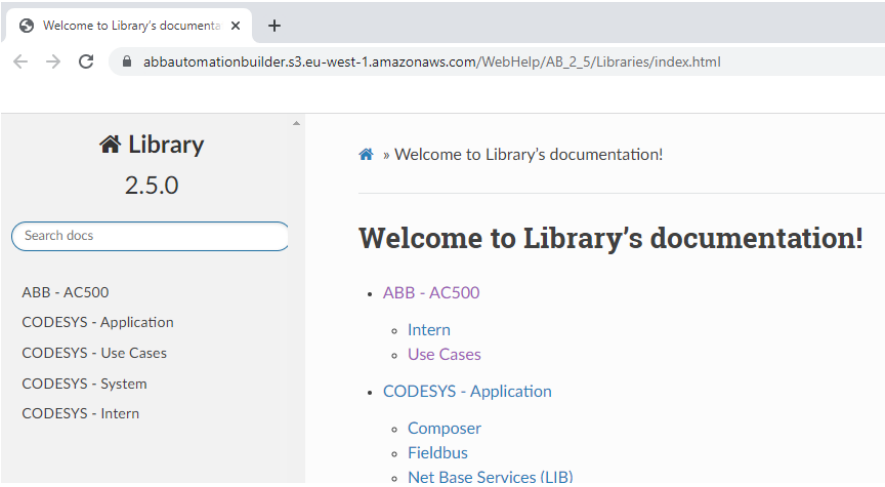


2.9.3 Online Web Library's Documentation

In this Web online documentation all POUs are documented based on an export from the information in the library manager of Automation Builder (see 2.9.1).

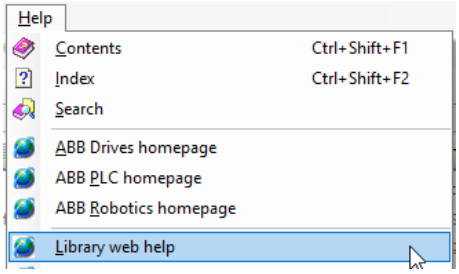
On the left side a tree view can be opened and a search option is provided.

It´s necessary to have the Automation Builder installed and open to use it.

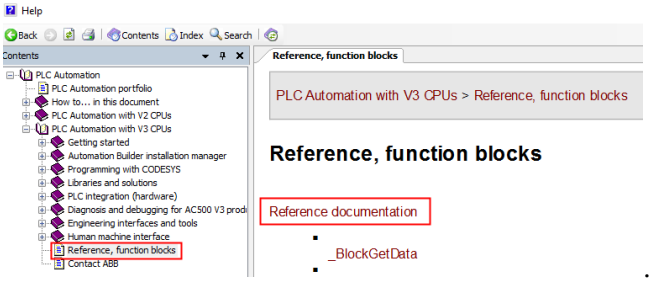


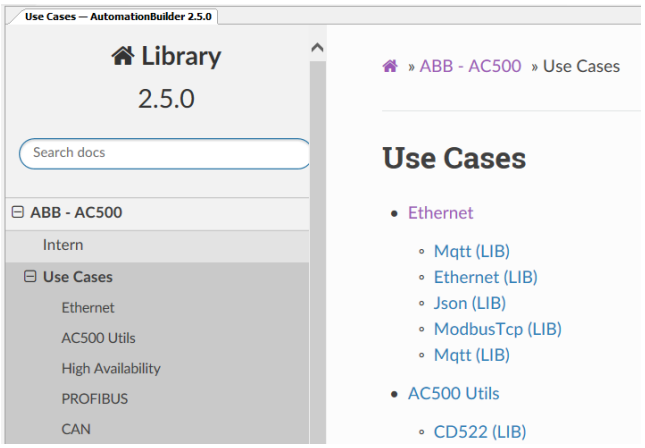
Access to this Library's documentation can be found from

- Automation Builder > Help > Library web help



- From the Automation Builder Help via PLC Automation with V3 CPUs > Reference, function blocks, then click on "Reference documentation"





3 Hardware, protocols and libraries availability

This chapter contains the information which hardware, protocols and libraries from AC500 V2 are ready to be used with AC500 V3 and which are not yet there available.

	This chapter gives an overview but will not be updated regularly. As we are working with high priority to fill all gaps. This list might be outdated, and features might be available already, even if marked as not in here.

3.1 Hardware

3.1.1 CPU overview

The following tables give a rough overview of possible replacements, but the detailed HW feature list needs to be checked to find out, if all needed HW features are available.

AC500 CPUs

CPU type V2	CPU type V3	Remark
PM572, PM582	PM5630-2ETH / PM5650-2ETH	Check features see 3.1.2.1
PM573-ETH, PM583-ETH	PM5630-2ETH / PM5650-2ETH / PM5670-2ETH	Check features see 3.1.2.1
PM585-ETH	PM5630-2ETH / PM5650-2ETH / PM5670-2ETH	Check features see 3.1.2.1
PM590-ETH, PM591-ETH	PM5650-2ETH / PM5670-2ETH	Check features see 3.1.2.1
PM592-ETH	PM5670-2ETH / PM5675-2ETH	Check features see 3.1.2.1
PM592-ETH CMS	Not yet available	Check features see 3.1.2.1
PM595-4ETH	PM5670-2ETH / PM5675-2ETH	Check features see 3.1.2.1

AC500-eCo CPUs

CPU type V2	CPU type V3	Remark
PM554 / PM564	PM5012/ PM5032	Check features in 3.1.2.2
PM554-ETH / PM564-ETH	PM5032/ PM5052	Check features in 3.1.2.2
PM556-ETH / PM566-ETH	PM5052/ PM5072	Check features in 3.1.2.2

3.1.2 CPUs detailed HW feature lists

This chapter contains high level overview of the availability of AC500 V2 features on potential AC500 V3 substitute PLCs. A green background color means that the feature is available, wither with same or extended scope. A yellow background means that there might be issues in migrating the PLC application. Therefore, it is highly recommended to clarify the options with your ABB sales representative.

3.1.2.1 Overview feature list of AC500 V2 with potential AC500 V3 substitute PLCs

3.1.2.1.1 PM572 / PM582

1	2	A	B	C	D	E	F	G	H	M	N
	1	**Type**					**PM572**	**PM582(-XC)**		**PM5630-2ETH(-XC)**	**PM5650-2ETH(-XC)**
	2										
	3	**Supply voltage**				Better					
	4		24 V DC			Slighty better	Yes	Yes		Yes	Yes
	5		100-240 V AC			Identical / Similar	No	No		No	No
	6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz	Freescale ARM Processor 32-bit /84 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	TI ARM Cortex-A9 32-bit-RISC / 600 MHz
	7	Total RAM memory / Total Flash memory				Not available	32 MB / 16 MB	32 MB / 16 MB		128 MB / 128 MB	256 MB / 512 MB
+	8	**Total user program memory (3)**					256 kB	928 kB		8 MB	80 MB
+	20	**Real-time clock**					Yes	Yes		Yes	Yes
+	22	**Plug-in memory card**					Yes	Yes		Yes	Yes
+	24	**Program execution**									
+	35	**Onboard digital inputs**					No	No		No	No
+	42	**Onboard digital outputs**					No	No		No	No
+	56	**Onboard digital input /output, configurable channels**					No	No		No	No
+	68	**Onboard analog outputs**					No	No		No	No
+	72	**Onboard analog inputs**					No	No		-	-
+	76	**Max. number of centralized inputs/outputs**									
+	85	**Max. number of decentralized inputs/outputs**					depends on the used standard fieldbus	depends on the used standard fieldbus		depends on the used standard fieldbus	depends on the used standard fieldbus
+	88	**Option board slots for extension**					No	No		No	No
	99	**Internal interfaces**									
+	100	**COM1**					Yes	Yes		Yes	Yes
+	110	**COM2**					Yes	Yes		No	No
+	119	**FieldbusPlug**					*Yes, but FieldBusPlug is out phased*	*Yes, but FieldBusPlug is out phased*		No	No
+	126	**CAN Interface**					No	No		Yes	Yes
+	136	**Ethernet**					-	-		Yes	Yes
+	185	**Terminal base for connection / CMS**					Yes / No	Yes / No		Yes / No	Yes / No
	195	**Supported communication with Communication Module**									
+	196	**Serial communication based**					Yes, CM574-RS	Yes, CM574-RS		*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*
+	203	**PROFIBUS DP**					Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP		Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP
+	208	**CAN**					Yes, CM588-CN, CM598-CN	Yes, CM588-CN, CM598-CN		Yes, CM598-CN	Yes, CM598-CN
+	215	**Ethernet TCP/IP**					Yes, CM597-ETH	Yes, CM597-ETH		*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*
+	232	**EtherCAT**					Yes, CM579-ETHCAT	Yes, CM579-ETHCAT		Yes, CM579-ETHCAT	Yes, CM579-ETHCAT
+	236	**PROFINET IO RT / PROFISAFE**					Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO		Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO
+	242	**Safety module SM560-S**					Yes, SM560-S-FD-1, SM560-S-FD-4	Yes, SM560-S-FD-1, SM560-S-FD-4		Yes, SM560-S-FD-1, (-FD-4 in preparation)	Yes, SM560-S-FD-1, (-FD-4 in preparation)
+	246	**Diagnostic and function**					Yes	Yes		Yes	Yes
	255	**Approvals**					Yes	Yes		Yes	Yes

3.1.2.1.2 PM573-ETH / PM583-ETH

	A	B	C	D	E	F	G	H	M	N	O
1	**Type**					**PM573-ETH(-XC)**	**PM583-ETH(-XC)**		**PM5630-2ETH(-XC)**	**PM5650-2ETH(-XC)**	**PM5670-2ETH(-XC)**
2											
3	**Supply voltage**				Better						
4		24 V DC			Slighty better	Yes	Yes		Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No	No		No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz	Freescale ARM Processor 32-bit /84 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	TI ARM Cortex-A9 32-bit-RISC / 600 MHz	TI ARM Cortex-A9 32-bit-RISC / 1 GHz
7	Total RAM memory / Total Flash memory				Not available	32 MB / 16 MB	32 MB / 16 MB		128 MB / 128 MB	256 MB / 512 MB	512 MB / 1024 MB
8	**Total user program memory (3)**					2048 kB	6144 kB		8 MB	80 MB	160 MB
20	**Real-time clock**					Yes	Yes		Yes	Yes	Yes
22	**Plug-in memory card**					Yes	Yes		Yes	Yes	Yes
24	**Program execution**										
35	**Onboard digital inputs**					No	No		No	No	No
42	**Onboard digital outputs**					No	No		No	No	No
56	**Onboard digital input /output, configurable channels**					No	No		No	No	No
68	**Onboard analog outputs**					No	No		No	No	No
72	**Onboard analog inputs**					No	No		-	-	-
76	**Max. number of centralized inputs/outputs**										
85	**Max. number of decentralized inputs/outputs**					depends on the used standard fieldbus	depends on the used standard fieldbus		depends on the used standard fieldbus	depends on the used standard fieldbus	depends on the used standard fieldbus
88	**Option board slots for extension**					No	No		No	No	No
99	**Internal interfaces**										
100	**COM1**					Yes	Yes		Yes	Yes	Yes
110	**COM2**					Yes	Yes		No	No	No
119	**FieldbusPlug**					*Yes, but FieldBusPlug is out phased*	*Yes, but FieldBusPlug is out phased*		No	No	No
126	**CAN Interface**					No	No		Yes	Yes	Yes
136	**Ethernet**					Yes	Yes		Yes	Yes	Yes
185	**Terminal base for connection / CMS**					Yes / No	Yes / No		Yes / No	Yes / No	Yes / No
195	**Supported communication with Communication Module**										
196	**Serial communication based**					Yes, CM574-RS	Yes, CM574-RS		*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*
203	**PROFIBUS DP**					Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP		Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP
208	**CAN**					Yes, CM588-CN, CM598-CN	Yes, CM588-CN, CM598-CN		Yes, CM598-CN	Yes, CM598-CN	Yes, CM598-CN
215	**Ethernet TCP/IP**					Yes, CM597-ETH	Yes, CM597-ETH		*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*
232	**EtherCAT**					Yes, CM579-ETHCAT	Yes, CM579-ETHCAT		Yes, CM579-ETHCAT	Yes, CM579-ETHCAT	Yes, CM579-ETHCAT
236	**PROFINET IO RT / PROFISAFE**					Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO		Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO
242	**Safety module SM560-S**					Yes, SM560-S-FD-1, SM560-S-FD-4	Yes, SM560-S-FD-1, SM560-S-FD-4		Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation)
246	**Diagnostic and function**					Yes	Yes		Yes	Yes	Yes
255	**Approvals**					Yes	Yes		Yes	Yes	Yes

3.1.2.1.3 PM585-ETH

	A	B	C	D	E	F	G	L	M	N
1	**Type**					**PM585-ETH**		**PM5630-2ETH(-XC)**	**PM5650-2ETH(-XC)**	**PM5670-2ETH(-XC)**
2										
3	**Supply voltage**				Better					
4		24 V DC			Slighty better	Yes		Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No		No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit /400 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	TI ARM Cortex-A9 32-bit-RISC / 600 MHz	TI ARM Cortex-A9 32-bit-RISC / 1 GHz
7	Total RAM memory / Total Flash memory				Not available	64 MB / 32 MB		128 MB / 128 MB	256 MB / 512 MB	512 MB / 1024 MB
8	**Total user program memory (3)**					7680 kB		8 MB	80 MB	160 MB
20	**Real-time clock**					Yes		Yes	Yes	Yes
22	**Plug-in memory card**					Yes		Yes	Yes	Yes
24	**Program execution**									
35	**Onboard digital inputs**					No		No	No	No
42	**Onboard digital outputs**					No		No	No	No
56	**Onboard digital input /output, configurable channels**					No		No	No	No
68	**Onboard analog outputs**					No		No	No	No
72	**Onboard analog inputs**					No		-	-	-
76	**Max. number of centralized inputs/outputs**									
85	**Max. number of decentralized inputs/outputs**					depends on the used standard fieldbus		depends on the used standard fieldbus	depends on the used standard fieldbus	depends on the used standard fieldbus
88	**Option board slots for extension**					No		No	No	No
99	**Internal interfaces**									
100	**COM1**					Yes		Yes	Yes	Yes
110	**COM2**					Yes		No	No	No
119	**FieldbusPlug**					*Yes, but FieldBusPlug is out phased*		No	No	No
126	**CAN Interface**					No		Yes	Yes	Yes
136	**Ethernet**					Yes		Yes	Yes	Yes
185	**Terminal base for connection / CMS**					Yes / No		Yes / No	Yes / No	Yes / No
195	**Supported communication with Communication Module**									
196	**Serial communication based**					Yes, CM574-RS		*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*
203	**PROFIBUS DP**					Yes, CM582-DP, CM592-DP		Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP
208	**CAN**					Yes, CM588-CN, CM598-CN		Yes, CM598-CN	Yes, CM598-CN	Yes, CM598-CN
215	**Ethernet TCP/IP**					Yes, CM597-ETH		*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*
232	**EtherCAT**					Yes, CM579-ETHCAT		Yes, CM579-ETHCAT	Yes, CM579-ETHCAT	Yes, CM579-ETHCAT
236	**PROFINET IO RT / PROFISAFE**					Yes, CM589-PNIO, CM579-PNIO		Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO
242	**Safety module SM560-S**					Yes, SM560-S-FD-1, SM560-S-FD-4		Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation)
246	**Diagnostic and function**					Yes		Yes	Yes	Yes
255	**Approvals**					Yes		Yes	Yes	Yes

3.1.2.1.4 PM590-ETH / PM591-ETH

	A	B	C	D	E	F	G	H	N	O	P
1	**Type**					**PM590-ETH**	**PM591-ETH(-XC)**		**PM5650-2ETH(-XC)**	**PM5670-2ETH(-XC)**	**PM5675-2ETH(-XC)**
2											
3	**Supply voltage**				Better						
4		24 V DC			Slighty better	Yes	Yes		Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No	No		No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 400 MHz	Freescale ARM Processor 32-bit /400 MHz		TI ARM Cortex-A9 32-bit-RISC / 600 MHz	TI ARM Cortex-A9 32-bit-RISC / 1 GHz	TI ARM Cortex-A9 32-bit-RISC / 1 GHz
7	Total RAM memory / Total Flash memory				Not available	64 MB / 32 MB	64 MB / 32 MB		256 MB / 512 MB	512 MB / 1024 MB	512 MB / 1024 MB
8	**Total user program memory (3)**					13316 kB	17924 kB		80 MB	160 MB	160 MB
20	**Real-time clock**					Yes	Yes		Yes	Yes	Yes
22	**Plug-in memory card**					Yes	Yes		Yes	Yes	Yes
24	**Program execution**										
35	**Onboard digital inputs**					No	No		No	No	No
42	**Onboard digital outputs**					No	No		No	No	No
56	**Onboard digital input /output, configurable channels**					No	No		No	No	No
68	**Onboard analog outputs**					No	No		No	No	No
72	**Onboard analog inputs**					No	No		-	-	-
76	**Max. number of centralized inputs/outputs**										
85	**Max. number of decentralized inputs/outputs**					depends on the used standard fieldbus	depends on the used standard fieldbus		depends on the used standard fieldbus	depends on the used standard fieldbus	depends on the used standard fieldbus
88	**Option board slots for extension**					No	No		No	No	No
99	**Internal interfaces**										
100	**COM1**					Yes	Yes		Yes	Yes	Yes
110	**COM2**					Yes	Yes		No	No	No
119	**FieldbusPlug**					*Yes, but FieldBusPlug is out phased*	*Yes, but FieldBusPlug is out phased*		No	No	No
126	**CAN Interface**					No	No		Yes	Yes	Yes
136	**Ethernet**					Yes	Yes		Yes	Yes	Yes
185	**Terminal base for connection / CMS**					Yes / No	Yes / No		Yes / No	Yes / No	Yes / No
195	**Supported communication with Communication Module**										
196	**Serial communication based**					Yes, CM574-RS	Yes, CM574-RS		*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*	*No, in preparation CM5610-2RS*
203	**PROFIBUS DP**					Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP		Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP	Yes, CM582-DP, CM592-DP
208	**CAN**					Yes, CM588-CN, CM598-CN	Yes, CM588-CN, CM598-CN		Yes, CM598-CN	Yes, CM598-CN	Yes, CM598-CN
215	**Ethernet TCP/IP**					Yes, CM597-ETH	Yes, CM597-ETH		*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*	*No, in preparation CM5640-2ETH*
232	**EtherCAT**					Yes, CM579-ETHCAT	Yes, CM579-ETHCAT		Yes, CM579-ETHCAT	Yes, CM579-ETHCAT	Yes, CM579-ETHCAT
236	**PROFINET IO RT / PROFISAFE**					Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO		Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO	Yes, CM589-PNIO, CM579-PNIO
242	**Safety module SM560-S**					Yes, SM560-S-FD-1, SM560-S-FD-4	Yes, SM560-S-FD-1, SM560-S-FD-4		Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation	Yes, SM560-S-FD-1, (-FD-4 in preparation)
246	**Diagnostic and function**					Yes	Yes		Yes	Yes	Yes
255	**Approvals**					Yes	Yes		Yes	Yes	Yes

3.1.2.2 Overview feature list of AC500-eCo V2 with potential AC500-eCo V3 substitute PLCs

3.1.2.2.1 PM554-xP

	A	B	C	D	E	F	G	H	M	N	O	P
1	**Type**					**PM554-TP**	**PM554-RP(-AC)**		**PM5012-T-ETH**	**PM5012-R-ETH**	**PM5032-T-ETH**	**PM5032-R-ETH**
2												
3	**Supply voltage**				Better							
4		24 V DC			Slighty better	Yes	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No	Yes		No	No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz			TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory				Not available	16 MB / 4 MB			128 MB / 128 MB		128 MB / 128 MB	
8	**Total user program memory (3)**					142 kB			1 MB		2 MB	
20	**Real-time clock**					Optional with TA561-RTC or TA562-RS-RTC			Optional, use option board TA5131-RTC		Yes	
22	**Plug-in memory card**					Optional with MC503 option board			Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	**Program execution**											
35	**Onboard digital inputs**					Yes			Yes		Yes	
42	**Onboard digital outputs**					Yes			Yes		Yes	
56	**Onboard digital input /output, configurable channels**					No			No		Yes	
68	**Onboard analog outputs**					No, specific CPU version			No onboard analog I/O, but possible using additional option board TA5126-2AO-UI		No onboard analog I/O, but possible using additional option board TA5126-2AO-UI	
72	**Onboard analog inputs**					No, specific CPU version			With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	**Max. number of centralized inputs/outputs**					Little bit more analog possible			No local I/O extension except with option boards		More digital / Less analog	
85	**Max. number of decentralized inputs/outputs**					depends on the used standard fieldbus (1)			depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	**Option board slots for extension**					2 Dedicated slots			Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	**Internal interfaces**								No onboard serial interface, only using additional option boards for serial communication			
100	**COM1**					Yes			Optional with option boards		Optional with option boards	
110	**COM2**					Optional with option boards			No		Optional with option boards	
119	**COM3**					No			No		No	
128	**Ethernet**					No, but with special versions			Yes		Yes	
176	**Diagnostic and function**					Yes			Yes		Yes	
181	**Approvals**							See detailled page 272 or www.abb.com/plc				
182												
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.											
184	(3) Total user program memory: contains user program code, data and web server											

3.1.2.2.2 PM564-xP

	A	B	C	D	E	F	G	H	M	N	O	P
1	Type					PM564-TP	PM564-RP(-AC)		PM5012-T-ETH	PM5012-R-ETH	PM5032-T-ETH	PM5032-R-ETH
2												
3	Supply voltage				Better							
4		24 V DC			Slighty better	Yes	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No	Yes		No	No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz			TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory				Not available	16 MB / 4 MB			128 MB / 128 MB		128 MB / 128 MB	
8	Total user program memory (3)					142 kB			1 MB		2 MB	
20	Real-time clock					Optional with TA561-RTC or TA562-RS-RTC			Optional, use option board TA5131-RTC		Yes	
22	Plug-in memory card					Optional with MC503 option board			Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	Program execution											
35	Onboard digital inputs					Yes			Yes		Yes	
42	Onboard digital outputs					Yes			Yes		Yes	
56	Onboard digital input /output, configurable channels					No			No		Yes	
68	Onboard analog outputs					Yes			No onboard analog I/O, but possible using additional option		No onboard analog I/O, but possible using additional option	
72	Onboard analog inputs					Yes			With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	Max. number of centralized inputs/outputs					Little bit more analog possible			No local I/O extension except with option boards		More digital / Less analog	
85	Max. number of decentralized inputs/outputs					depends on the used standard fieldbus (1)			depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	Option board slots for extension					2 Dedicated slots			Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	Internal interfaces								No onboard serial interface, only using additional option boards for serial communication			
100	COM1					Yes			Optional with option boards		Optional with option boards	
110	COM2					Optional with option boards			No		Optional with option boards	
119	COM3					No			No		No	
128	Ethernet					No, but with special versions			Yes		Yes	
176	Diagnostic and function					Yes			Yes		Yes	
181	Approvals								See detailled page 272 or www.abb.com/plc			
182												
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.											
184	(3) Total user program memory: contains user program code, data and web server											

3.1.2.2.3 PM554-xP-ETH

	A	B	C	D	E	F	G	L	M	N	O
1	Type					PM554-TP-ETH		PM5032-T-ETH	PM5032-R-ETH	PM5052-T-ETH	PM5052-R-ETH
2											
3	Supply voltage				Better						
4		24 V DC			Slighty better	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No		No	No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory				Not available	16 MB / 4 MB		128 MB / 128 MB		128 MB / 128 MB	
8	Total user program memory (3)					654 kB		2 MB		4 MB	
20	Real-time clock					Optional with TA561-RTC or TA562-RS-RTC		Yes		Yes	
22	Plug-in memory card					Optional with MC503 option board		Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	Program execution										
35	Onboard digital inputs					Yes		Yes		Yes	
42	Onboard digital outputs					Yes		Yes		Yes	
56	Onboard digital input /output, configurable channels					No		Yes		Yes	
68	Onboard analog outputs					No, specific CPU version		No onboard analog I/O, but possible using additional option		No onboard analog I/O, but possible using additional option	
72	Onboard analog inputs					No, specific CPU version		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	Max. number of centralized inputs/outputs					Little bit more analog possible		More digital / Less analog		More digital / More analog	
85	Max. number of decentralized inputs/outputs					depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	Option board slots for extension					2 Dedicated slots		Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	Internal interfaces										
100	COM1					Yes		Optional with option boards		Optional with option boards	
110	COM2					Optional with option boards		Optional with option boards		Optional with option boards	
119	COM3					No		No		Optional with option boards	
128	Ethernet					Yes		Yes			
176	Diagnostic and function					Yes		Yes		Yes	
181	Approvals										
182											
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.										
184	(3) Total user program memory: contains user program code, data and web server										

3.1.2.2.4 PM564-xP-ETH

	A	B	C	D	E	F	G	H	M	N	O	P
1	Type					PM564-TP-ETH	PM564-RP-ETH(-AC)		PM5032-T-ETH	PM5032-R-ETH	PM5052-T-ETH	PM5052-R-ETH
2												
3	Supply voltage				Better							
4		24 V DC			Slighty better	Yes	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No	Yes		No	No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz			TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory				Not available	16 MB / 4 MB			128 MB / 128 MB		128 MB / 128 MB	
8	Total user program memory (3)					654 kB			2 MB		4 MB	
20	Real-time clock					Optional with TA561-RTC or TA562-RS-RTC			Yes		Yes	
22	Plug-in memory card					Optional with MC503 option board			Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	Program execution											
35	Onboard digital inputs					Yes			Yes		Yes	
42	Onboard digital outputs					Yes			Yes		Yes	
56	Onboard digital input /output, configurable channels					No			Yes		Yes	
68	Onboard analog outputs					Yes			No onboard analog I/O, but possible using additional option		No onboard analog I/O, but possible using additional option	
72	Onboard analog inputs					Yes			With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	Max. number of centralized inputs/outputs					Little bit more analog possible			More digital / Less analog		More digital / More analog	
85	Max. number of decentralized inputs/outputs					depends on the used standard fieldbus (1)			depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	Option board slots for extension					2 Dedicated slots			Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	Internal interfaces											
100	COM1					Yes			Optional with option boards		Optional with option boards	
110	COM2					Optional with option boards			Optional with option boards		Optional with option boards	
119	COM3					No			No		Optional with option boards	
128	Ethernet					Yes			Yes			
176	Diagnostic and function					Yes			Yes		Yes	
181	Approvals											
182												
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.											
184	(3) Total user program memory: contains user program code, data and web server											

3.1.2.2.5 PM556-TP-ETH

	A	B	C	D	E	F	G	L	M	N	O
1	Type					PM556-TP-ETH		PM5052-T-ETH	PM5052-R-ETH	PM5072-T-2ETH	PM5072-T-2ETHW(2)
2											
3	Supply voltage				Better						
4		24 V DC			Slighty better	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC			Identical / Similar	No		No	No	No	No
6	Type of processor / Processor clock frequency				Lower feature	Freescale ARM Processor 32-bit / 50 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory				Not available	16 MB / 4 MB		128 MB / 128 MB		128 MB / 128 MB	
8	Total user program memory (3)					1666 kB		4 MB		8 MB	
20	Real-time clock					Optional with TA561-RTC or TA562-RS-RTC		Yes		Yes	
22	Plug-in memory card					Optional with MC503 option board		Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	Program execution										
35	Onboard digital inputs					Yes		Yes		Yes	
42	Onboard digital outputs					Yes		Yes		Yes	
56	Onboard digital input /output, configurable channels					No		Yes		Yes	
68	Onboard analog outputs					No, specific CPU version		No onboard analog I/O, but possible using additional option		No onboard analog I/O, but possible using additional option	
72	Onboard analog inputs					No, specific CPU version		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	Max. number of centralized inputs/outputs					Little bit more analog possible		More digital / More analog		More digital / More analog	
85	Max. number of decentralized inputs/outputs					depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	Option board slots for extension					2 Dedicated slots		Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	Internal interfaces										
100	COM1					Yes		Optional with option boards		Optional with option boards	
110	COM2					Optional with option boards		Optional with option boards		Optional with option boards	
119	COM3					No		Optional with option boards		Optional with option boards	
128	Ethernet					Yes		Yes		Yes	
176	Diagnostic and function					Yes		Yes		Yes	
181	Approvals										
182											
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.										
184	(3) Total user program memory: contains user program code, data and web server										

3.1.2.2.6 PM566-TP-ETH

1	Type				PM566-TP-ETH		PM5052-T-ETH	PM5052-R-ETH	PM5072-T-2ETH	PM5072-T-2ETHW(2)
2										
3	Supply voltage			Better						
4		24 V DC		Slighty better	Yes		Yes	Yes	Yes	Yes
5		100-240 V AC		Identical / Similar	No		No	No	No	No
6	Type of processor / Processor clock frequency			Lower feature	Freescale ARM Processor 32-bit / 50 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz		TI ARM Cortex-A9 32-bit-RISC / 300 MHz	
7	Total RAM memory / Total Flash memory			Not available	16 MB / 4 MB		128 MB / 128 MB		128 MB / 128 MB	
8	Total user program memory (3)				1666 kB		4 MB		8 MB	
20	Real-time clock				Optional with TA561-RTC or TA562-RS-RTC		Yes		Yes	
22	Plug-in memory card				Optional with MC503 option board		Yes, onboard micro memory card socket		Yes, onboard micro memory card socket	
24	Program execution									
35	Onboard digital inputs				Yes		Yes		Yes	
42	Onboard digital outputs				Yes		Yes		Yes	
56	Onboard digital input /output, configurable channels				No		Yes		Yes	
68	Onboard analog outputs				Yes		No onboard analog I/O, but possible using additional option		No onboard analog I/O, but possible using additional option	
72	Onboard analog inputs				Yes		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD		With opt. boards TA5120-2AI-UI, TA5122-2AI-TC, TA5123-2AI-RTD	
76	Max. number of centralized inputs/outputs				Little bit more analog possible		More digital / More analog		More digital / More analog	
85	Max. number of decentralized inputs/outputs				depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)		depends on the used standard fieldbus (1)	
88	Option board slots for extension				2 Dedicated slots		Each slot can be used for all type of existing option boards, same option board can be used on several slots per CPU			
99	Internal interfaces									
100	COM1				Yes		Optional with option boards		Optional with option boards	
110	COM2				Optional with option boards		Optional with option boards		Optional with option boards	
119	COM3				No		Optional with option boards		Optional with option boards	
128	Ethernet				Yes		Yes		Yes	
176	Diagnostic and function				Yes		Yes		Yes	
181	Approvals									
182										
183	(1) Real-time clock requires optional TA561-RTC or TA562-RS-RTC. (2) COM2 requires TA562-RS-RTC, TA562-RS or new TA569-RS-ISO.									
184	(3) Total user program memory: contains user program code, data and web server									

3.1.3 Safety CPUs

CPU / Feature	AC500 V2	AC500 V3
SM560-S	On all coupler slots: SM560-S, SM560-S-FD1, SM560-S-FD4	On all coupler slots: SM560-S

3.1.4 AC31-Replacement

Device	AC500 V2	AC500 V3
AC31-Replacement CPUs	07KT98-xx-yy	
AC31 Replacement CS31-IOs	07AC9x-yy, 07AI91-AD, 07DC9x-AD	

3.1.5 Hardware Interfaces

Fieldbus / Interface	AC500 V2	AC500 V3	AC500-eCo V3
Serial communication * protocols see chapter below	COM1(terminal block), - RS232, RS422, RS485 COM2(D-SUB) - RS232, RS422, RS485	COM1(terminal block) CAN terminal block	Option boards (terminal block) - RS232, RS232 isolated - RS485, RS485 isolated
Ethernet interfaces * protocols see chapter below	ETH1 and coupler CM597-ETH PM591-2ETH additional ETH2, PM595-4ETH additional ETH2, ETH3, ETH4,	ETH1 and ETH2	ETH1 PM5072 additional ETH2

3.1.6 Fieldbus communication modules (CM)

Fieldbus / Protocol	AC500 V2	AC500 V3	AC500-eCo V3
Profinet Controller	CM579-PNIO with PM595-4ETH optional on board of ETH3 and ETH4.	CM579-PNIO	Not available
Profinet Device	CM589-PNIO and CM589-PNIO-4	CM589-PNIO and CM589-PNIO-4	Not available
Profibus Master	CM592-DP	CM592-DP	Not available
Profibus Slave	CM582-DP	CM582-DP	Not available

CANopen Manager (Master)	CM578-CN CM598-CN	Onboard (terminal block) and CM598-CN	Not available
CANopen Device (Slave)	CM588-CN	Not available	Not available
CAN 2A/2B	CM598-CN	Onboard (terminal block) and CM598-CN	Not available
EtherCAT	CM579-ETHCAT with PM595-4ETH optional on board of ETH3 and ETH4.	CM579-ETHCAT	Not yet available
Modbus TCP	CM577-ETH CM597-ETH	Only onboard Interface	Only onboard Interface

3.1.7 Communication Interface modules

Fieldbus / Protocol	AC500 V2	AC500 V3	AC500-eCo V3
Profinet	CI501-PNIO CI502-PNIO CI504-PNIO CI506-PNIO	CI501-PNIO CI502-PNIO	n.a.
Profibus Master	CI541-DP CI542-DP	CI541-DP CI542-DP	n.a.
CANopen	CI581 CI582	CI581 * CI582 * *) only to onboard CANopen Manager	
EtherCAT	CI511-ETHCAT CI512-ETHCAT	CI511-ETHCAT CI512-ETHCAT	
Modbus TCP	CI521-MODTCP CI522-MODTCP	CI521-MODTCP CI522-MODTCP	CI521-MODTCP CI522-MODTCP

3.1.8 Other devices

Module(s) / Device(s)	AC500 V2	AC500 V3	AC500-eCo V3
IO-Modules	All	All	All
Function Modules	DC541-CM	Not supported	Not available
	FM562-PTO	Not supported	Use on board PTO outputs

3.1.9 Memory Card

	AC500 V2	AC500 V3
Size	Max. 2 GB (ABB tested Cards)	Max. 8 GB from ABB
Format	Standard	SDHC supported, SDXC is not supported
File System	DOS 8.3 – Filesystem (Short Names)	FAT 32, Long Names
Card Function	Function Description of Card in Init File SDCARD.INI	Function Description of Card in Init File SDCARD.INI
Functionality	One Card for • FW Update (CPU, CM Module, Display) • User Data • Retain-/Persistent-Data • Project Source Code • Application Prog. • CM Module Conf.	One Card for • FW Update (System, Boot, Update, Display) • User Data • Retain-/Persistent- Data • Project Source Code • Application Prog. • Trends and Alarms • Backup and Restore

Hint: For AC500 V3 the memory card is optional. All functionalities can also be performed via Automation Builder or the internal userdisk memory.

For AC500 V3 there is no SD Card mandatory. All Functionalities can also be performed via Automation Builder or the internal userdisk memory in the AC500.

3.2 Protocols

3.2.1 Serial protocols

Fieldbus / Protocol	AC500 V2	AC500 V3	AC500-eCo V3
Modbus RTU Client and Sever	COM1 and COM2 Coupler: CM574-RS	Onboard COM1	Optional Up to 3 option boards available (terminal block)
CS31	COM1 CM574-RS	Not yet available	Not yet available
RCOM	CM574-RCOM	Not available	Not available
ASCII	COM1 and COM2	Not yet available	Not yet available
MULTI	COM1 and COM2	Not yet available	Not yet available

SysLibCom – free prog	COM1 and COM2	CAA SerialCom	CAA SerialCom
BACNET MS/TP	COM1 and COM2	COM1 and COM2	Option board TA5142-RS485I-BacNet
CANopen Manager (Master)	CM598-CN	Onboard (terminal block) and CM598-CN	Not available
CANopen Device (Slave)	CM588-CN	Not available	Not available
CAN 2A/2B	CM598-CN	Onboard (terminal block) and CM598-CN	Not available

3.2.2 Ethernet based protocols

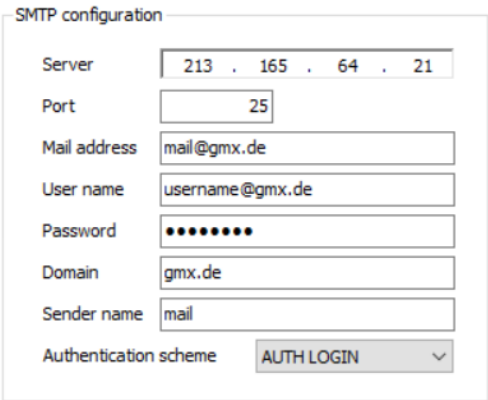
AC500 V2	AC500 V3
Modbus TCP, Client + Server	Modbus TCP, Client + Server
TCP/IP Sockets	TCP/IP Sockets (CAA)
UDP (Standard UDP + ABB UDP)	TCP/UDP
Online access (configurable by the user, 3S and ABB drivers)	3S Online access (configurable only regarding security settings)
IEC60870-5-104 (Control - & Substation)	IEC60870-5-104 (Control - & Substation) Control station depending on PLC type
	IEC61850 (MMS Server, GOOSE, Edition 1)
SNTP (Client + Server)	SNTP, NTP (Client + Server)
SMTP	SMTP (including encryption)
SNMP	SNMP
FTP Server (Client as Lib)	FTP(s) Server (connected to both Interfaces)
WEBserver	CODESYS HTML5 WEB Visualization
	CODESYS network variables
OPC DA (Server installed at PC)	OPC DA (Server installed at PC) OPC UA Server OPC UA Client
MQTT	MQTT
HTTP	HTTP + HTTPS
	Client Protocols added to Protocols Server Protocols added below dedicated Ethernet Port
BACnet B-ASC	BACnet B-BC

3.2.3 SMTP protocol

The biggest difference between V2 and V3 is, that v2 does not support TLS or SSL encryption. As all mail provider use encryption today, this functionality is only usable for local mail sending using a local unencrypted mail software.

The second differnce is that in V3 the functionality is done in function blocks which must be prepared properly. In V2 there is a mask to put in everything. This is easier to use but on the contrary the possibilities are more limited than a function bock based programming.

The V2 SMTP has a mask to put in the parameters:



In V3, function blocks are used:

```
(*PREPARE MAIL*)
BodyToSend                  := MailContent;         // Content of the mail
mailToSend.sSubject         := MailSubject;         // Subject of the mail
mailToSend.sMailTo          := MailTo;              // Destination mail address
mailToSend.sMailFrom        := MailFrom;
mailToSend.sMailCC          := '';
mailToSend.sMailBCC         := '';
userInfo.sUserName:=Settings.UserName;
userInfo.sPassword:=Settings.Password;
mailToSend.Body.pString     := ADR(BodyToSend);
mailToSend.Body.uiSize      := INT_TO_UINT(LEN(BodyToSend));
```

3.3 Libraries

The following tables can be taken as rough orientation where to find the required functions and function blocks. Please consider:

- In some cases, the AC500 V3 do not offer the complete functionality as the AC500 V2 library. Typically, mostly used functions and function blocks are available so that a project conversion can be done.
- There is not always a one-to-one relation between AC500 V2 and AC500 V3 libraries.
- They are not complete and might have changed since last release of this document. New libraries for AC500 V3 are continuously developed.
- Final decision, if all required functionality is available with AC500 V3 has to be made on function / function block level. The full documentation of all libraries that are installed together with Automation Builder is available online from the internet. Latest version is the Automation Builder 2.5 library documentation.

3.3.1 AC500 System libraries

Topic	AC500 V2	AC500 V3	Remark
AC500	SysExt_AC500_V10.lib, SysInt_AC500_V10.lib, SysIntExt_AC500_V13.lib	Pm, IO	Processor Module and IO Utilities
	BusDiag.lib, Diag_AC500_V20.lib	Diag, DiagCpu, DiagUtil, DiagIoBus, DiagTypes, DiagBase, DiagHistory, DiagS500	Diagnosis libraries
	CD522_AC500_V13.lib	CD522	supporting functions of encoder & PWM module CD522
	Counter_AC500_V20.lib	IO	
	RTC_AC500_V20.lib	Pm	
	FBP.lib	n.a.	
	DC541_AC500_V11.lib	n.a.	
	Serie90_AC500_V10.lib	n.a.	
AC500 (Eco)	n.a.	EcoUtils	utility functions for AC500 ECO PLCs
	OnBoardIO_AC500_V13.lib	OnboardIO	

3.3.2 Communication libraries

Topic	AC500 V2	AC500 V3	Remark
FTP client	FTPClient_AC500_V22.lib	Planned for Q3 2023	
BACnet	BACnet_BASC_AC500_V28.lib	BACnet	V3 has B-BC profile plus AB plugin (V2 only B-ASC library)
IEC 61850	n.a.	IEC61850Server	
IEC 60870	IEC60870_AC500_V20.lib	IEC60870_5_104	
EtherCAT	EtherCAT_AC500_V13.lib, EtherCAT_CS_AC500.lib	EtherCAT, EcatBase	
	CAMSWITCH_AC500_V13.lib, MCX_AC500_Vx.lib		
CAN	CANopen_AC500_V11.lib	CM598Can	CM598-CAN specific function blocks
Serial	ASCII_AC500_V10.lib	Com	Serial Communication for AC500 V3
Modbus RTU	Modbus_AC500_V10.lib, MODBUS_Ext_AC500_V20.lib	ModbusRtu	
Modbus Ethernet	Ethernet_AC500_V10.lib	ModbusTcp	
Com. Profibus	PROFIBUS_AC500_V10.lib	CM592Profibus	
Com. Profinet	PROFINET_AC500_V13.lib, PROFINET_Ext_AC500_V20.lib	PnioCntrl	
Com. MQTT	MqttClient_AC500_V28.lib	Mqtt	
	JSON_AC500_V28.lib	Json	
Com. Ethernet	Ethernet_AC500_V10.lib	Ethernet	Ethernet communication with AC500 devices.
CS31	CS31_AC500_V20.lib	n.a.	
Arcnet	ARCNET_AC500_V12.lib, ARCNETExt_AC500_V12.lib	n.a.	
Rcom	RCOM_AC500_V13.lib	n.a.	
DeviceNet	DeviceNet_AC500_V11.lib	n.a.	

3.3.3 Safety libraries

The Safety libraries run exclusively inside the Safety PLC which is independent of the used AC500 PLC (AC500 V2 or AC500 V3).

3.3.4 Application / Product libraries

Topic	AC500 V2	AC500 V3	Remark
Motion base	MC_Base_AC500_V11.lib, MC_Blocks_AC500_V11.lib, CompactMotionControl_AC500_V12.lib	MotionControl	
	ECAT_AC500_APPL_V21.lib	Ecat_CiA402	
	MathFunctions_AC500_V23.lib	MathFunctions	
Motion Eco	n.a.	MotionContro-lEco	
Motion PTO	PTO_FM562_MC_support_Vxx.lib	MotionContro-lEco	FM562 is not supported in V3. Use onboard PTO of AC500-eCo V3.
Motion Load control	n.a.	MotionControl-Load	
Motion coordi-nated	CoordinatedMotion_AC500_V23.lib, MC_CoBlocks_AC500_V23.lib, CMC_Ext_AC500_V23.lib, CMC_Transformationen_AC500_V23.lib	Not yet available	Scheduled for the future
Motion drive based	ACS350_AC500_V11.lib, ACS350_MC_support_AC500_V11.lib, ACS800_AC500_V11.lib, ACS800_MC_support_AC500_V11.lib, ACSM1_AC500_V11.lib, ACSM1_MC_support_AC500_V11.lib	Not yet available	Scheduled for the future
Temperature control	ADCTRL_AC500_V24.lib, TECT_EXT_AC500_V24.lib, TECT_TEMP_CONTROL_AC500_V24.lib	Not yet available	Scheduled for the future
Pumping (V2 Wa-ter)	PMP_AC500_V25.lib	Pump	
Logging (V2 Wa-ter)	LogData_AC500_V23.lib	DataLogger	
	n.a.	DataLoggerEco	
	n.a.	DataLoggerMulti	
Solar	Solar_AC500_V22.lib, SolarNREL_AC500_V22.lib	Solar	

Drives	ACSDrivesBase_AC500_V20, ACSDrivesComModRTU_AC500_V20.lib, ACSDrivesComModTCP_AC500_V22.lib, ACSDrivesCom-ModTCP_Ext_AC500_V24.lib, ACSDrivesComPB_AC500_V24.lib, ACSDrivesComPN_AC500_V24.lib, DCSDrives_AC500_V24.lib	Drives	
High Availability CS31	HA_CS31_AC500_V23.lib	n.a.	
High Availability Modbus TCP	HAModbus_AC500_V26.lib	HaModbus	
High Availability Modbus TCP	CI52x_AC500_V26.lib	CI52x	
Condition Moni-toring System	CMS_IO_AC500_V24.lib, WAV_FILE_AC500_V24.lib, SP_AC500_V24_App.lib	Not yet available	Scheduled for the future
HVAC	CTRL_AC500_App_V22.lib, HVAC_AC500_App_V22.lib	Not yet available	Scheduled for the future
Process Control Object library	Pco_AC500_V28.lib	Prototype (ask support)	

3.3.5 CAA libraries

V2	**V3**	**Remark**
CAA_Async_Man.lib	CAA Async Manager	
CAA_Tick.lib	CAA Tick	
CAA_TickUtil.lib	CAA TickUtil	
CAA_Types.lib	CAA Types	
CAA_Callback.lib	CAA Callback	
CAA_File.lib	CAA File	Most used functions are con-verted to V3. Some functions to work with Archives and on the Disk are not available in V3.

3.3.6 System libraries

Before using function blocks of the CODESYS system libraries, the documentation has to be checked for any changes in functionality. E.g. some rarely used function blocks will be block execution of the PLC application until they are done, which might lead to watchdog exceptions.

V2	V3	Remark
SysLibSockets.lib	SysSocket2	All FUN and FBs are converted to V3 The V3 library provide much more functionalities then the V2 library
SysLibCom.lib	Com	Same functions are available in V3 library but with different names and interface configuration No 1:1 use possible
SysLibCallback.lib	n.a.	
SysLibInitLibrary.lib	n.a.	
SysLibMem.lib	SysMem	All FUNs are converted to V3 The V3 library provide much more functionalities then the V2 library for example SysMemIsValidPointer
SysTaskInfo.lib	CmpIecTask	Need to be tested
SysLibTime.lib	AC500_Pm SysEvent	Both libraries provide time functions.
SysLibAlarmTrend.lib		Handled by visu library
SysLibEvent.lib	CmpIecTask	Same functions are available in V3 library but must be tested
SysLibFile.lib	SysFile	All FUNs are converted to V3 The V3 library provide much more functionalities then the V2 library
SysLibIecTasks.lib	CmpIecTask	Need to be tested!
SysLibPLCConfig.lib	n.a.	
SysLibPlcCtrl.lib	n.a.	See CmpApp
SysLibProjectInfo.lib	CmpApp	Need to be tested!
SysLibRtc.lib	SysTimeRtc	Need to be tested!
SysLibSem.lib	SysSem_Implementation	Need to be tested!
SysLibStr.lib	StringUtils	

SysLibTasks.lib	SysTask	Need to be tested!
SysLibVisu.lib		Different libraries for V3
SysLib OnlineAccess.lib		Handled by runtime

3.3.7 CODESYS libraries

V2	V3	Remark
standard.lib	Standard	All FUN and FBs are converted to V3 except SEMA (Fb) is not available in V3 Standard library
Iecsfc.lib	IecSfc	All FUN and FBs are converted to V3
util.lib	Util	All FUN and FBs are converted to V3 The V3 library provide much more functionalities then the V2 library

3.3.8 Application libraries

V2	V3	Remark
MSSQL_AC500_V24.lib	MsSQL	All FUN and FBs are converted to V3. The library is not working with AC500 V3 eCo PLC
MYSQL_AC500_App_V21.lib	MySQL	All FUN and FBs are converted to V3
HTTP_AC500_V24.lib	Http	All FUN and FBs are converted to V3. In addition TLS is supported in the V3 library and further function blocks are available

3.3.9 C-code libraries

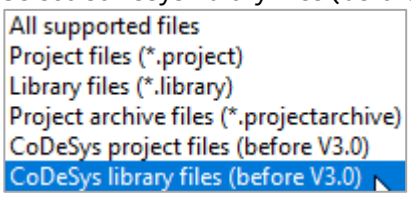
Not yet available in AC500 V3.

4 Conversion of custom libraries

4.1 Upgrade to V3.5

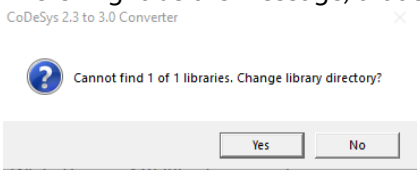
Before converting the project from V2 to V3 custom libraries needs to be upgraded.

1. Open Automation Builder
2. Click File → Open Project
3. Select CoDeSys library files (before V3.0) and open the library from your hard disk

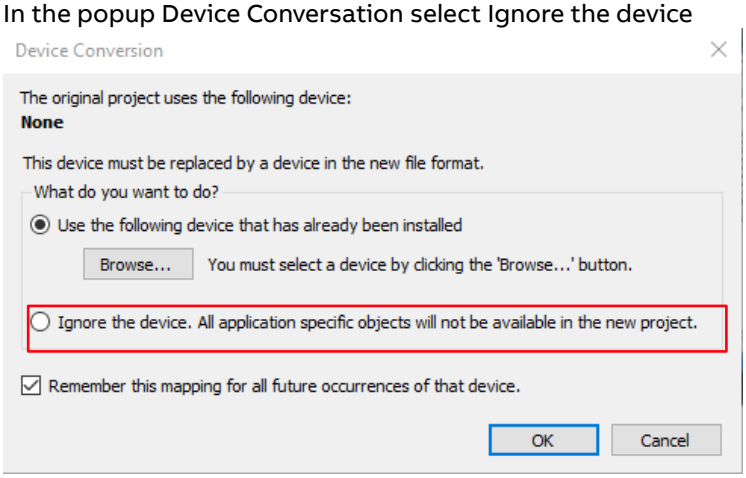


> In Automation Builder 2.5.X the opening of library files is by default not possible.
>
> To open a Codesys V2.3 library either open Automation Builder in Profile 2.4 or download and install the CODESYS V2.3 Converter from the codesys store:
> https://store.codesys.com/en/codesys-v2-3-converter.html
> This needs to be installed via package manager (Tools → Package Manager)

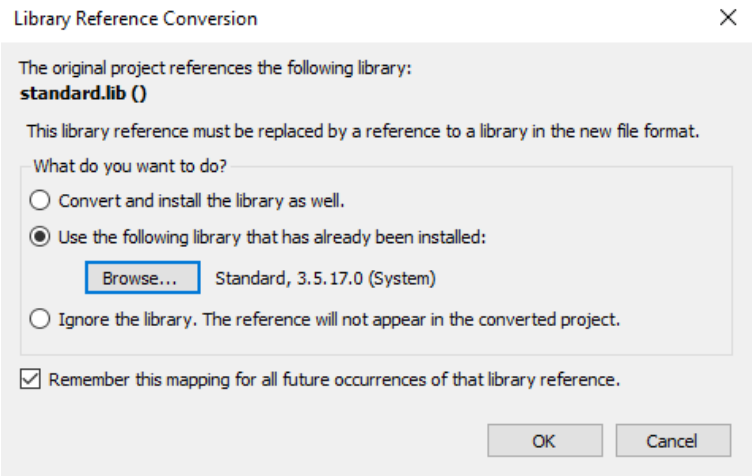
4. There might be the message, that some libraries are not found. Click No to continue



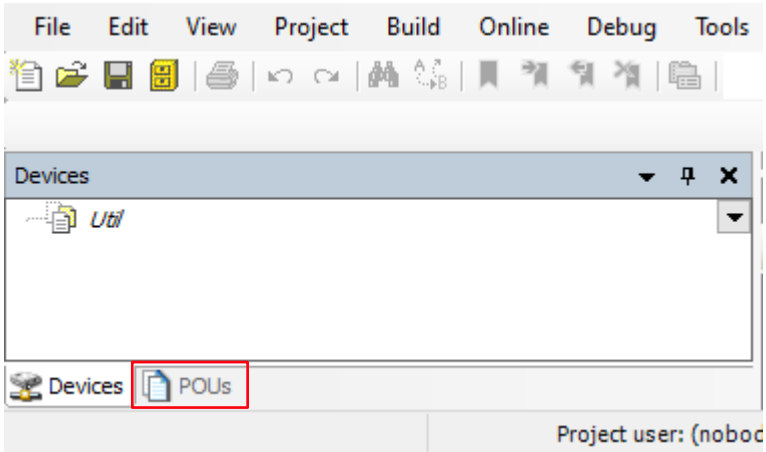
5. In the popup Device Conversation select Ignore the device



6. If other libraries are used in the library, you need to select the corresponding V3 Version. Select use the following library that has already been installed and browse to find the right library in the repository

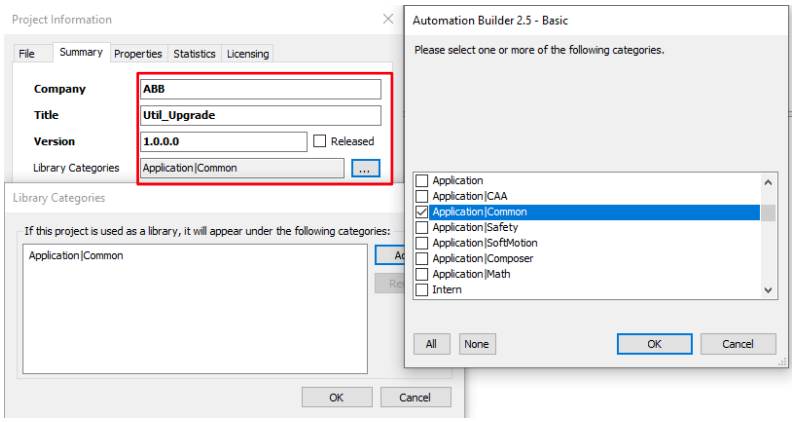


7. Automation Builder will update the library to version 3 and open afterwards. To see the elements, change from Devices to POUs



8. Double click Project information

   a) Insert your Company, a Title and the Version
   b) Set the category of the library. The generic Library Category Base description file can be used or adapted to the needs. LibraryCategoryBase.libcat.xml



   c) Set a namespace for accessing the library

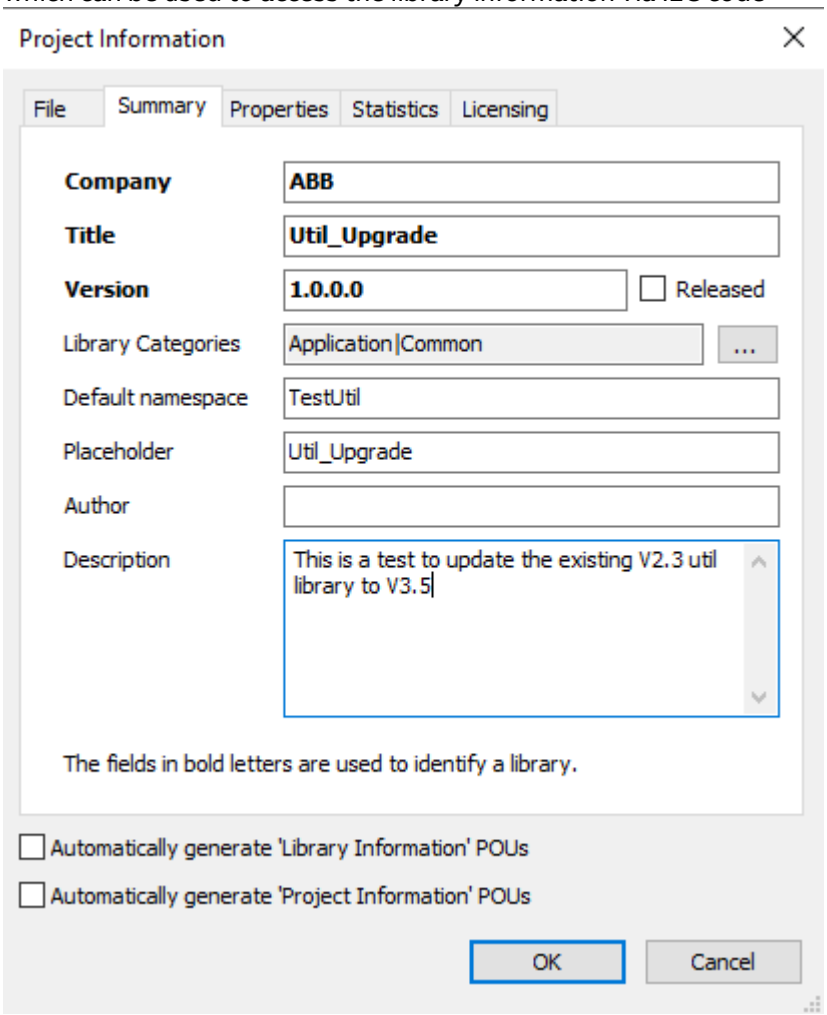
	The namespace defines how the internal function blocks and functions are accessed. E.g. when using the namespace TestUtil the function block BLINK can be accessed via the namespace TestUtil.BLINK

d) Set a Placeholder for the library

	The placeholder is the name of the library inside the repository. Different versions of a library can then be handled and selected via the placeholder of the library. In many cases the Title of the library is also used as Placeholder.

e) Fill in Author (optional) and Description (recommended)

f) By clicking Automatically generate 'Library Information' POUs functions are created which can be used to access the library information via IEC code



9. Switch to the tab Properties of the Project Information.

   a) Click Version string and Remove this Key
   b) Type Key *DocFormat*, Type *Text*, Value *reStructuredText* and click Add
   c) Optional the key *LanguageModelAttribute*, Type *Text*, Value *qualified-access-only* can be added

	The key LanguageModelAttribute defines if the internal objects can be accessed without the namespace or not.

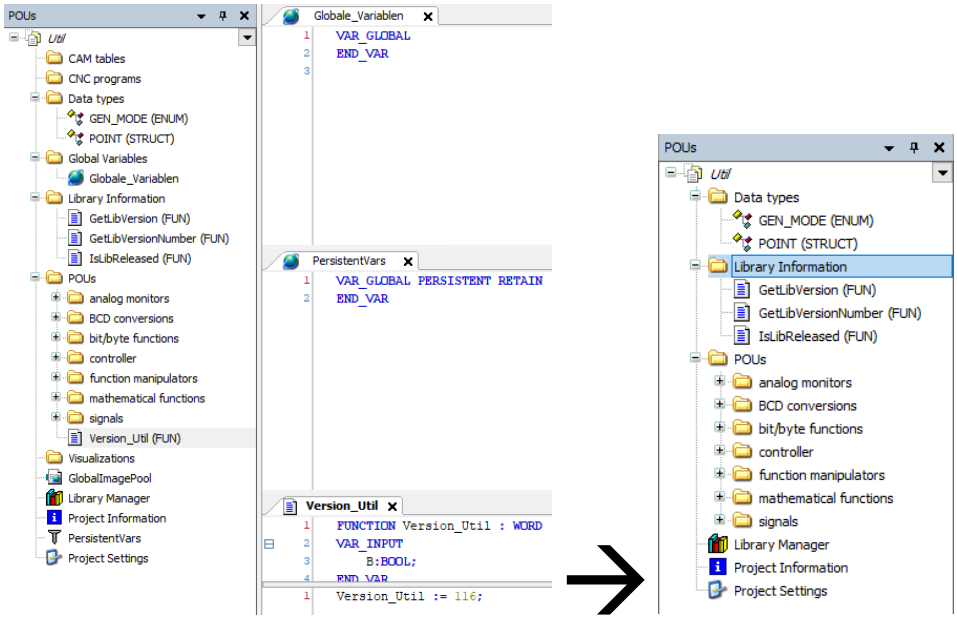
	If the attribute is not set all objects like function blocks, functions, structures, Variables, … can be accessed without the namespace (normal behavior in V2) If the attribute is added like described in c) the access is only possible via the namespace e.g. TestUtil.BLINK instead of BLINK If no qualified access is used all objects inside the library should have a unique prefix. According to the example of the Util library upgrade. If this upgraded library as well as the original library are both added to the project and BLINK is instantiated there will be no error message like in V2.3, but the function block of the library which is more on top of the library manager is used.

d) Click OK to save the changes

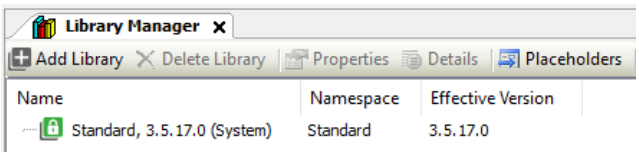
4.2 Changes in POUs, Libraries, …

After upgrading the library there will be empty or not required objects in the POUs tree which were added during the upgrade can be deleted. Furthermore the existing code should be checked

1. Delete all empty folders (Here: CAM tables, CNC programs, Visualization)
2. Delete all empty objects (Here Globale_Variablen, GlobalImagePool, PersistentVars)
3. Delete all not required POUs (Here Version_Util replaced by GetLibVersion)



4. Open the Library Manager and check that all Library Placeholders are resolved



5. Click Build → Check all Pool Object

   The check should run error free and only showing warnings

If the custom library is using non basic CODESYS functionalities additional code changes might be required. The needed steps for different protocols, visualizations, … are described in the next chapters focusing on the project upgrade.

Solve as many warnings from the build message as possible e.g. by replacing implicit type conversations to explicit type conversations.

6. Check the documentation of the function blocks and change them if required

4.3 Install and share the custom V3 library

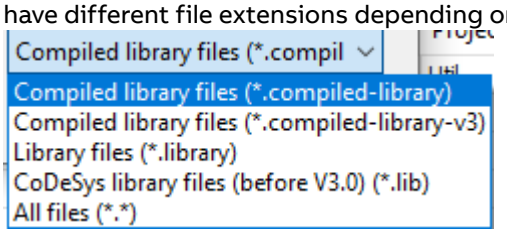
In V2 custom libraries could be read and write protected with a password. In V3 there is the differentiation between source and compiled libraries.

A source library *.library is readable by anybody. A source library can be installed into the library repository. If this is the case debugging like step into or breakpoints can be used inside a library

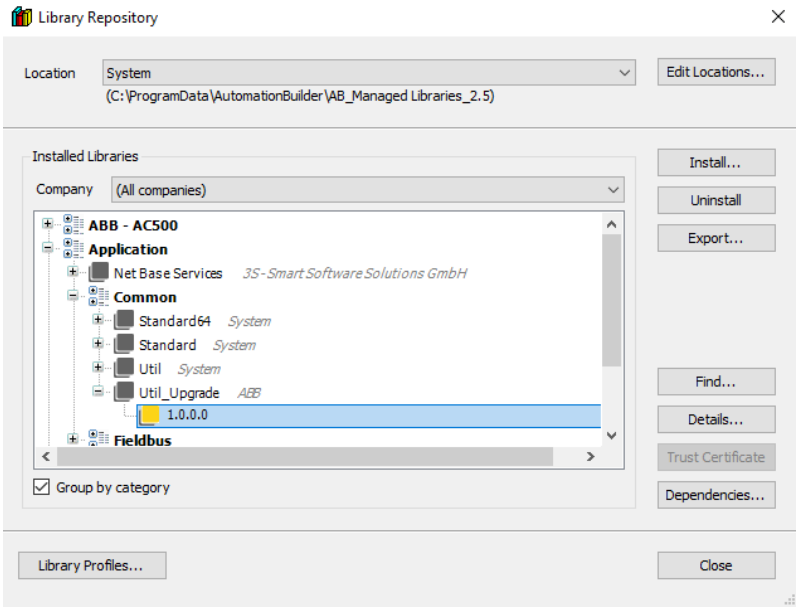
A compiled library *.compiled-library is only binary code. This is neither readable nor editable. There is no way to come from the compiled library back to the source library.

Only compiled libraries should be shared with customers. The access to the source library should be limited to a small group of persons. Please always save the source library somewhere as there is no way to get the source code from a compiled library.

1. Save the library
   a) Click File → Save Project as… to save the source library on the hard drive
   b) Click File → Save Project as Compiled Library… to save the compiled library on the hard drive
2. Install the library
   a) Click Tools → Library Repository
   b) Click Install… and search navigate to the file on the hard drive
      In the drop down menu it might be useful to select All files, as compiled libraries can have different file extensions depending on version, signing, … of the library



   c) Click open to install the library to the library repository



Now the placeholder Util_Upgrade can be used on this PC in all projects.

	Note: Read more about *Libraries* for AC500 V3 in the Automation Builder online Help. For this, follow the content path: PLC Automation > PLC Automation with V3 CPUs > Programming with CODESYS> Libraries

After upgrading the custom libraries used in a V2 project the V2 project itself can be upgraded to V3 as described in the next chapter.

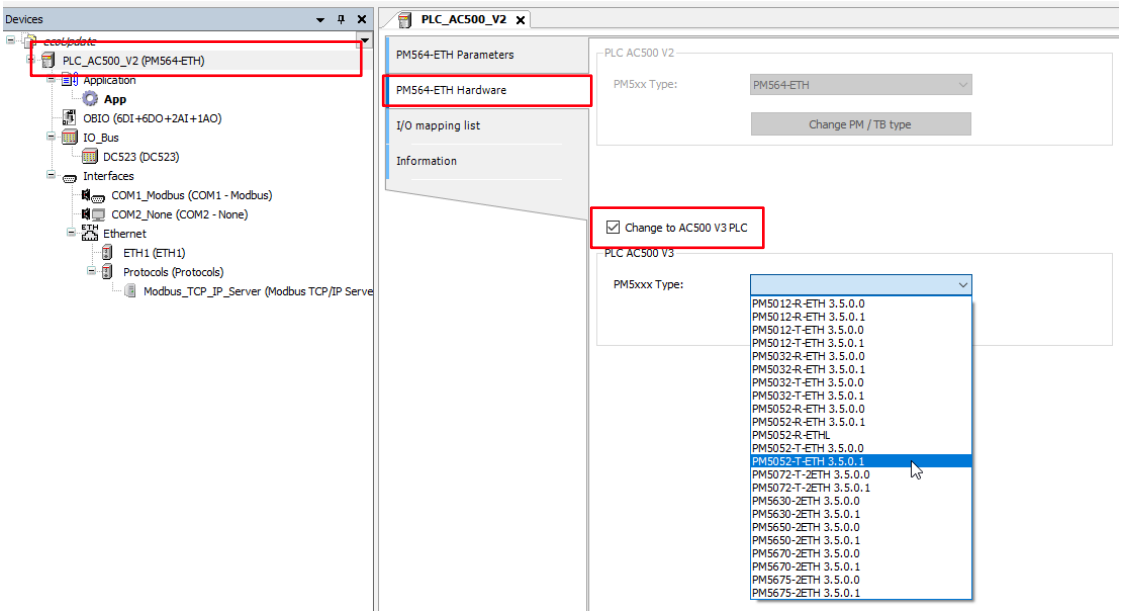
5 Conversion of the project

This chapter describes how to upgrade an AC500 V2 project to an AC500 V3 project. The sub-chapters are focusing on additional adaptions for the hardware configuration, the IEC Application and the visualization.

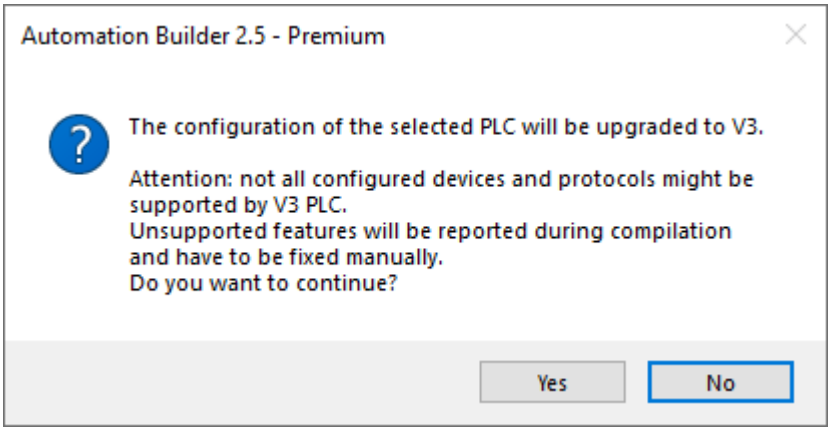
This chapter is describing the general steps, which need to be done for all projects to upgrade.

The chapters 6 and 7 are then focusing more specific on the fieldbus and SCADA communication protocols and which changes needs to be done there.

1. Open the existing AC500 V2 project
2. Double click the PLC to open and select Hardware in the menu



3. Click Change to V3 PLC and select the used V3 PLC from the drop-down menu
4. Click Create V3 PLC and confirm the dialog



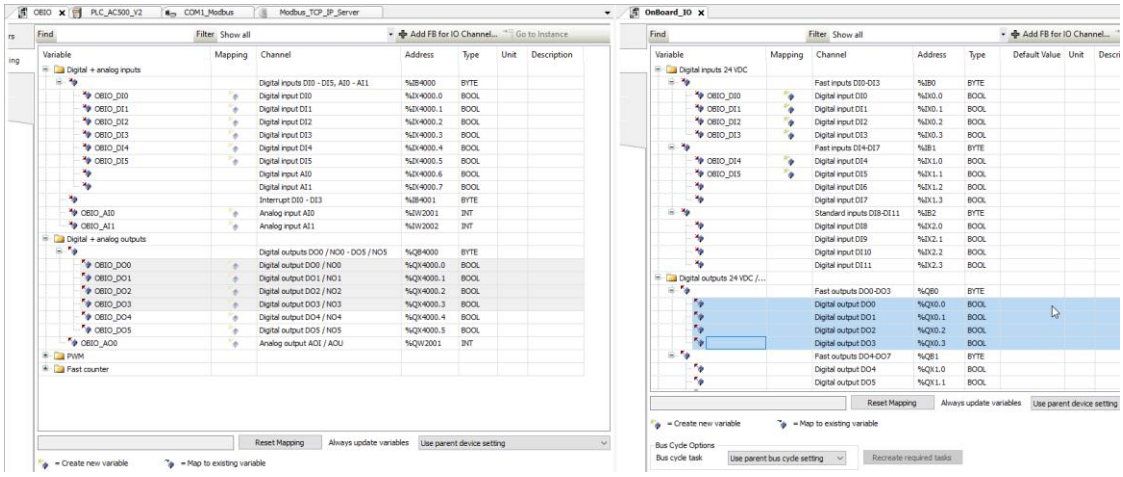
5. The V3 PLC is created in the same project and all devices (including configuration), POUs, visualizations and other elements are automatically added.

5.1 Typical adaptions in the Hardware configuration

5.1.1 IOs

S500 and S500-eCo modules added to the IO bus are also available in the AC500 V3 PLC. All settings and mapping variables remain the same.

Onboard IO settings and mapping, used for AC500-eCo V2 PLCs are not copied and need to be adapted manually. The input and outputs can be just copy & pasted from the AC500 V2 onboard IO to the AC500 V3 onboard IO.



If a AC500-eCo V2 PLC with analog inputs & outputs was used, additional hardware devices are required for these inputs / outputs. These can be realized with spare analog channels in S500(-eCo) modules on the IO bus or additional analog option boards[1] TA5120/ TA5126.

5.1.2 Alarms, Traces & Recipes

After upgrading the project the AlarmConfiguration, Recipe Manager and Sampling_Trace are automatically added to the Project. In case these elements were configured in the AC500 V2 project the elements should be rechecked in the AC500 V3 project.

In case these elements were not used in the existing AC500 V2 program, they can be deleted.

5.1.3 Library Manager

General information regarding library management in AC500 V3 PLCs can be found in chapter 2.3.

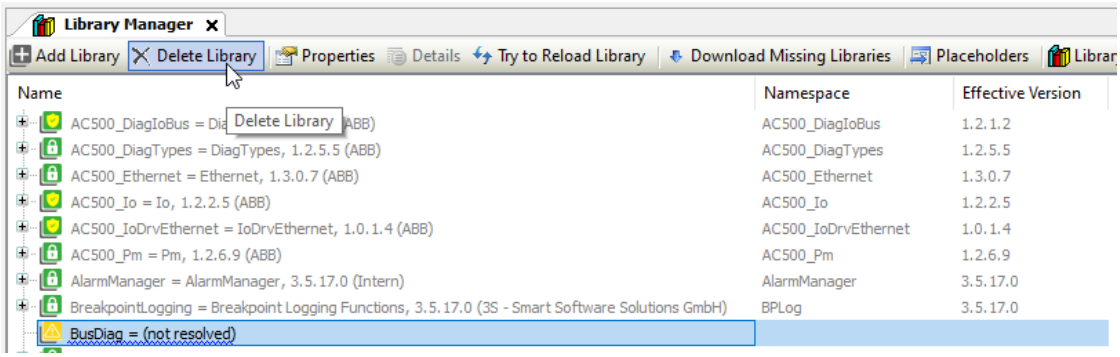
Inside the library manager there are already some automatically added libraries depending on the hardware configuration in the project.

Some AC500 V2 libraries might be directly upgraded to a AC500 V3 library with the same functionality. Some libraries have a successor library with similar functionality. For more details check chapter 3.3.

All libraries which could not be resolved needs to be deleted or changed.

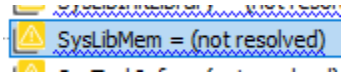
[1] Planned for Q4 2022

In almost all cases there are unresolved placeholders in the library manager, which results into error messages. For some of these unresolved placeholders there are already correctly resolved placeholders added, e.g. *DiagIoBus* provides the functionality of *BusDiag*.

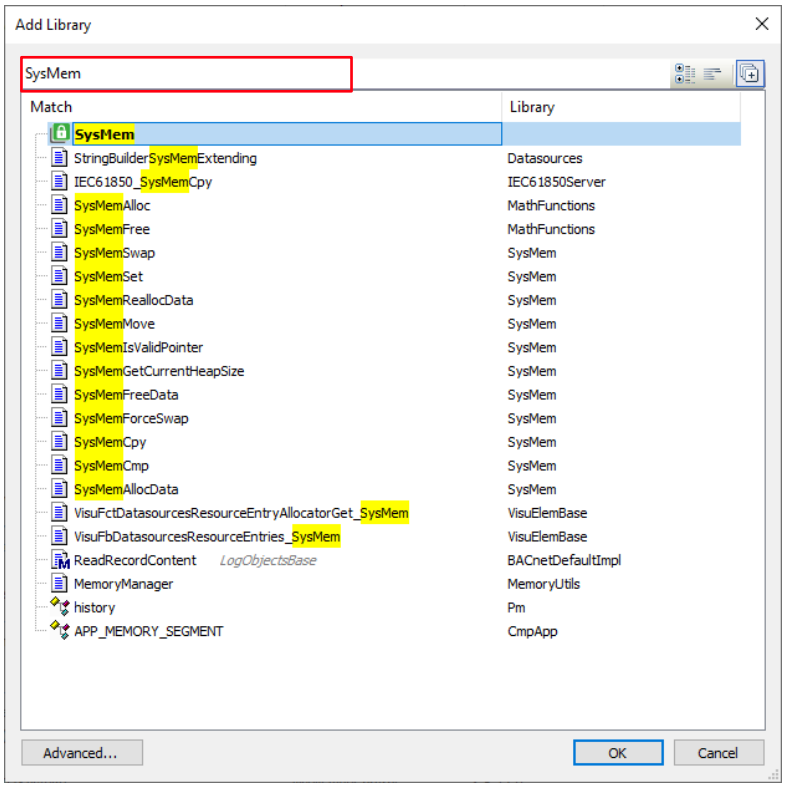


Other libraries need to be exchanged to a V3 successor.

For example for the SysLibMem



The easiest way to remove all errors concerning the libraries is to remove all unresolved placeholders from the library manager. After that, Automation Builder will indicate function blocks that cannot be resolved. The names of these function blocks can be entered in the “Add Library” dialog to add the related library. E.g. in case of start typing *SysMemCopy* it will be quickly indicated that the *SysMem* placeholder has to be added.

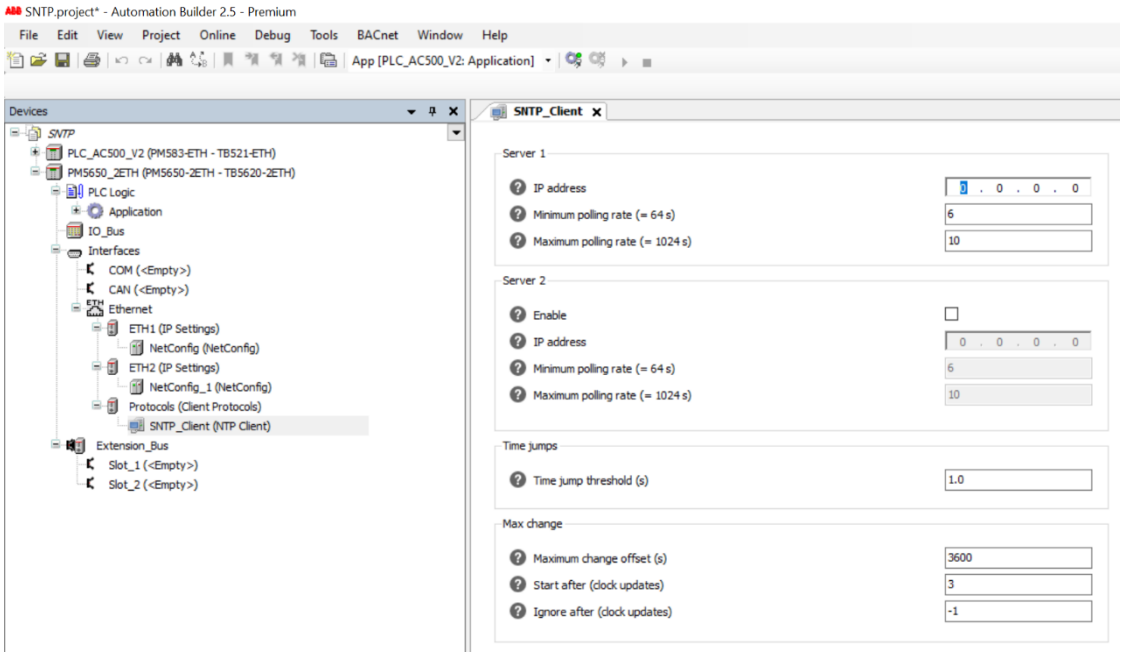


5.1.4 SNTP

AC500 V3 PLC is supporting NTP (Network Time Protocol) instead of SNTP(Simple Network Time Protocol). The difference between these both names are the high level of accuracy for NTP.

Client:

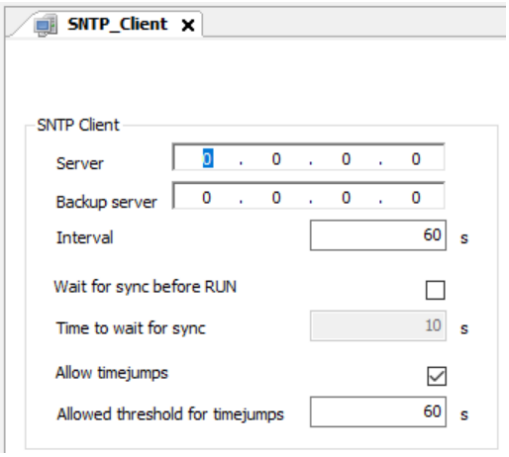
When changing the PLC from AC500 V2 to AC500 V3 the protocol for SNTP **client** is automatically added on the Ethernet interface under "*Protocols* (*Client Protocols*)"



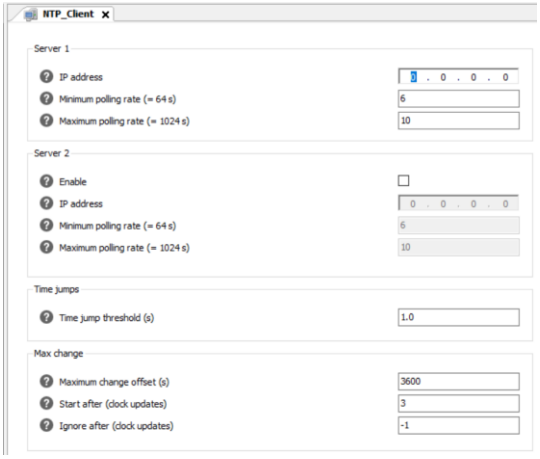
	Note: The settings like IP address, Backup Server etc. are not converted. Please set the NTP settings in the AC500 V3 PLC again.

The settings for the client are similar but in a better readable structure

AC500 V2: SNTP



AC500 V3: NTP



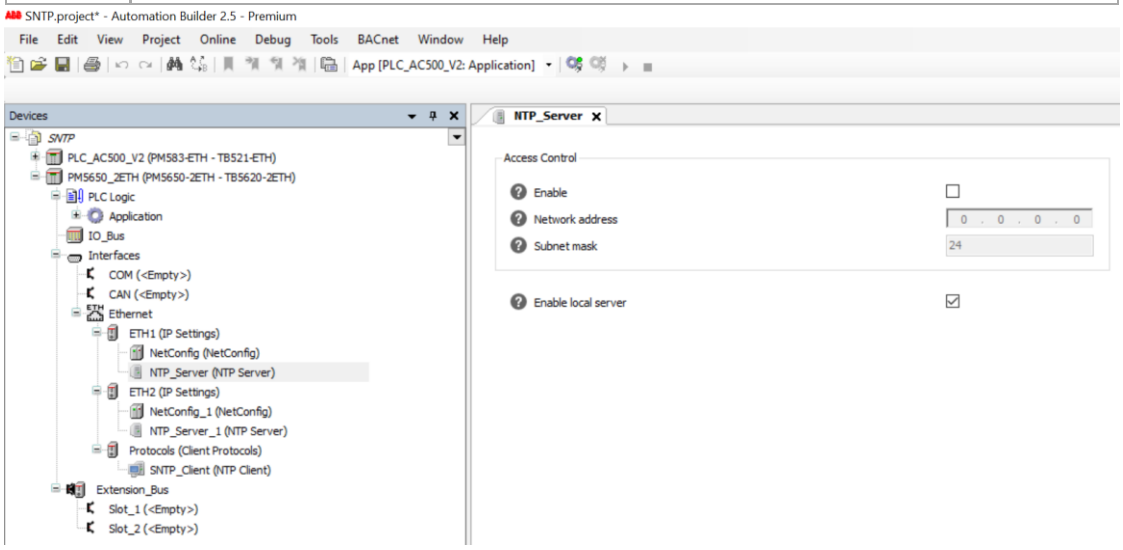
In both configurations you can set an IP address for the server who will be contacted to receive the current UTC time. Also a Backup Server or Server 2 is supported when Server or Server 1 is not reachable.

Further information can be found in the online help: PLC Automation with V3 CPUs > PLC integration (hardware) > Configuration in Automation Builder for AC500 V3 products > Protocols and special servers > NTP/SNTP protocol > Configuration of the (S)NTP protocol > (S)NTP client configuration

Server:

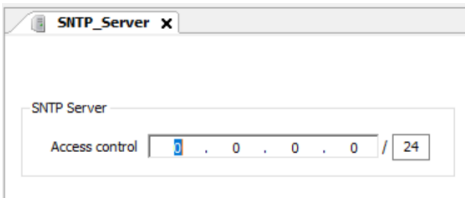
When changing the PLC from AC500 V2 to AC500 V3 the protocol for SNTP **server** need to be added on the Ethernet interface (*ETH1 or ETH2 or both interfaces*).

	Note: This protocol is not automatically added when converting the project from AC500 V2 to AC500 V3. The reason is, that the converter does not know at what ETH interface the server needs to be added.

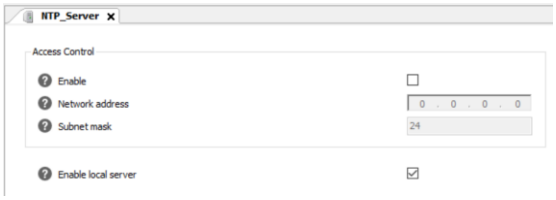


The settings for the server are very similar from AC500 V2 to AC500 V3.

AC500 V2: SNTP



AC500 V3: NTP



Access control and Network address have the same behavior. You can set an IP address from a device that is allowed to access and request the time. If the IP is set to 0.0.0.0 means all IP addresses within the same Subnet are allowed. The Subnet mask can be set via /24.

The subnet mask 24 means: 255.255.255.0

The subnet mask 16 means: 255.255.0.0

The NTP Server for AC500 V3 has in addition an Enable checkbox to activate the server.

Further information can be found in the online help: PLC Automation with V3 CPUs > PLC integration (hardware) > Configuration in Automation Builder for AC500 V3 products > Protocols and special servers > NTP/SNTP protocol > Configuration of the (S)NTP protocol > (S)NTP server configuration

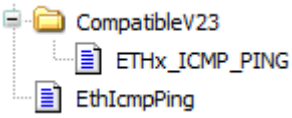
5.2 Typical adaptions in the IEC application

5.2.1 PLCopen conform function blocks

Between AC500 V2 and AC500 V3 the style of the function blocks and functions has changed. The AC500 V3 POUs are according to the PLCopen guidelines.

AC500 V2	AC500 V3
SNAKE_STYLE e.g. ETHx_ICMP_PING	CamelCaseStyle e.g. EthIcmpPing
Function blocks have an enable input “EN”	Function blocks are either edge controlled “Execute” input or level controlled “Enable” input
Function block outputs are “DONE”, “ERR” and “ERNO”	Function block outputs are “Done”, “Error” and “ErrorId”
In case an error occurs DONE and ERR are true	In case an error occurs only ERR is true (exclusive)

In some AC500 V3 Libraries there are AC500 V2 compatibility function blocks in a folder called CompatibleV23. For other libraries there is a separate compatibility library with the extension 23 available.



These function blocks can be used to reuse AC500 V2 code in a AC500 V3 PLC without any modification.

The compatibility function blocks are a wrapper around the PLCopen function block with renamed inputs and outputs and the changed behavior in case of errors.

For memory or CPU optimization it is always recommended to use the PLCopen conform AC500 V3 function block instead of the compatibility version.

5.2.2 Modbus addresses %M and %R

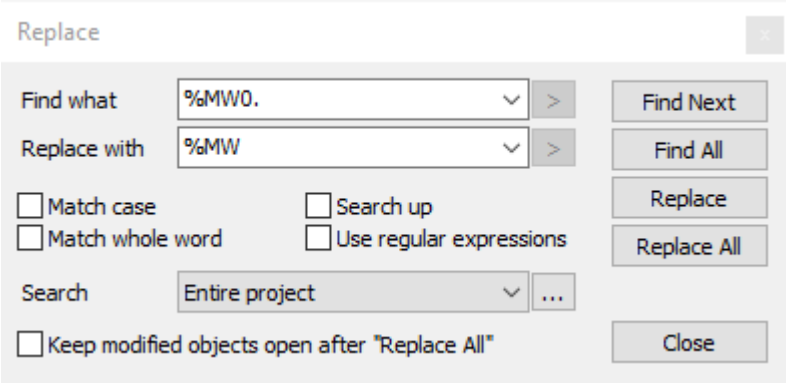
As described in chapter 2.1.5 the AC500 V3 PLCs have one single large memory segment instead of several segments like in V2.

The addresses need to be changed accordingly.

%M Addresses

Press CTRL + H for replacing. The goal is to delete the ‘0.’ in the memory addresses.

- Find what: %MX0. Replace with: %MX
- Find what: %MB0. Replace with: %MB
- Find what: %MW0. Replace with: %MW
- Find what: %MD0. Replace with: %MD



Afterwards all %M memory addresses which were in the 0 segment in V2 have correct V3 addresses.

In case also higher segments were used these need to be changed manually, to avoid overlapping memory areas.

%R Addresses

The %R memory is addressed remanent memory. This area is not existing anymore in AC500 V3.

If the %R area was only used to retain the data and not for Modbus addressing delete the %R address and move the variable to a VAR (GOLBAL) RETAIN PERSISTENT declaration. Either within the Global PersistentVars list or any local declaration.

If the %R area was used for retained Modbus memory, the %R variable must be addressed to any free %M memory address. To retain this %M memory address the pragma {attribute 'noinit'} is used.

```
VAR_GLOBAL
  iA AT %MW0 : INT;
  {attribute 'no_init'}
  iB AT %MW1 : INT;
END_VAR
```

5.2.3 ACS/DCS Drives Library

There are no AC500 V2 compatible function blocks available. A comparison of AC500 V2 and AC500 V3 functionality and which function blocks should be used can be found in the Online-Help:

PLC Automation with V3 CPUs > Libraries and solutions > ACS/DCS drives libraries > Introduction > Comparison of V2 and V3 drives library

PS553-Drives Library package (V2)		PS5605-Drives Library package (V3)	
Library Name	**Function Block**	**Library**	**Function Block**
ACSDrivesBase_AC500_V2	ACS3XX_DRIVES_CTRL_BASIC	Not supported – use DrvControlModbusACS	
	ACS_DRIVES_CTRL_ENG	ABB_Drives_AC500	DrvControlModbusEng
	ACS_DRIVES_CTRL_STANDARD		DrvControlModbusACS
	ACS_DRIVES_CTRL_STANDARD_GEN		DrvControlACS
	ACS_MOD_READ_N_PRM		DrvModbusRead
	ACS_MOD_WRITE_N_PRM		DrvModbusWrite
	ACS_REF_SCALING		DrvScaling
ACSDrivesComModRTU_AC500_V20	ACS3XX_COM_MOD_RTU	Not supported	
	ACS_COM_MOD_RTU	ABB_Drives_AC500	DrvModbusRtu
	ACS_COM_MOD_RTU_ENHANCED		DrvModbusRtu
	ACS_COM_MOD_RTU_GEN	ABB_ModbusRtu_AC500	ModRtuToken
	ACS_COM_MOD_RTU_GEN_READ_N_PRM		ModRtuRead
	ACS_COM_MOD_RTU_GEN_WRITE_N_PRM		ModRtuWrite
			ModRtuReadWrite23
ACSDrivesComModTCP_AC500_V22	ACS_COM_MOD_TCP	ABB_Drives_AC500	DrvModbusTcp
	ACS_COM_MOD_TCP_ENHANCED		DrvModbusTcp
ACSDrivesCom-ModTCP_Ext_AC500_V24	ACS_COM_MOD_TCPx	ABB_Drives_AC500	DrvModbusTcp
	ACS_COM_MOD_TCPx_ENHANCED		DrvModbusTcp
DCSDrives_AC500_V24	DCS_DRIVES_CTRL	ABB_Drives_AC500	DrvControlModbusDCS
	DCS_DRIVES_CTRL_GEN		DrvControlDCS
ACSDrivesComPN_AC500_V24	ACS_PN_WRITE_N_PRM_DPV1	ABB_Drives_AC500	DrvPnWrite
	ACS_PN_READ_N_PRM_DPV1		DrvPnRead
ACSDrivesComPB_AC500_V24	ACS_PB_READ_N_PRM_DPV1	Will be supported in next Release	
	ACS_PB_WRITE_N_PRM_DPV1		
	ACS_COM_PB	Not supported	
	ACS_COM_PB_PZD		
	ACS_PB_READ_PRM_DPV0		
	ACS_PB_WRITE_PRM_DPV0		
		ABB_Drives_AC500	DrvModbusReadWrite23
		ABB_Drives_AC500	DrvModbusRtuBroadcast
		ABB_Drives_AC500	DrvControlCANCiA402

5.3 Typical migration for 3rd party devices

There is a shared device repository for AC500 V2 and AC500 V3. Therefore, all devices that have been available for AC500 V2 can be used with AC500 V3 without any constraints.

6 Reestablishing communication to field devices

6.1 Modbus TCP

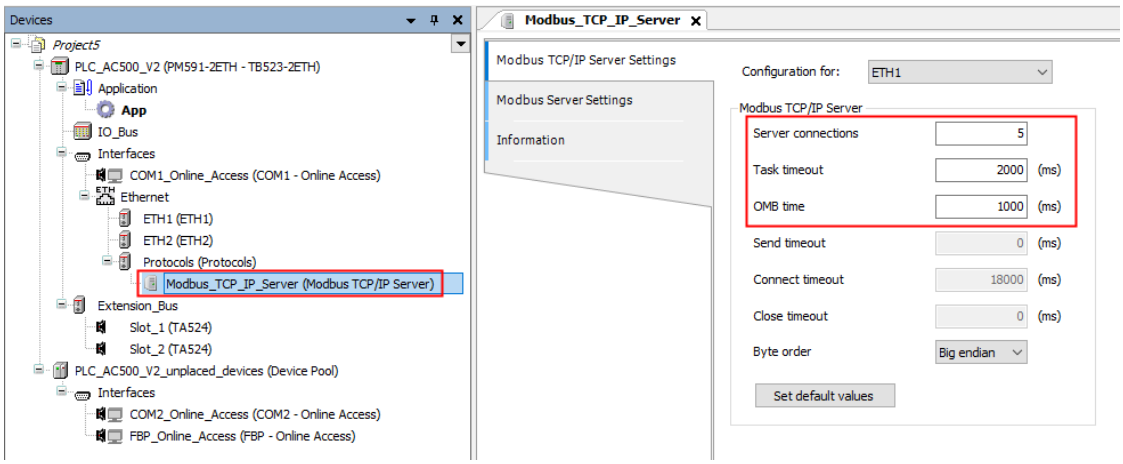
The main difference regarding the use of Modbus in AC500 V3 is the difference in the memory structure and therefore in the absolute addressing of the Modbus variables already described in chapter 2.1.5 “Addressable ” and chapter 2.1.6 “Modbus addresses / Byte order”. Tipps for the conversion of the Modbus addresses are in chapter 5.2.2 “Modbus addresses %M and %R”.

Following other differences have to be taken into account while changing from AC500 V2 to Ac500 V3 with a Modbus TCP connection.

6.1.1 AC500 V2

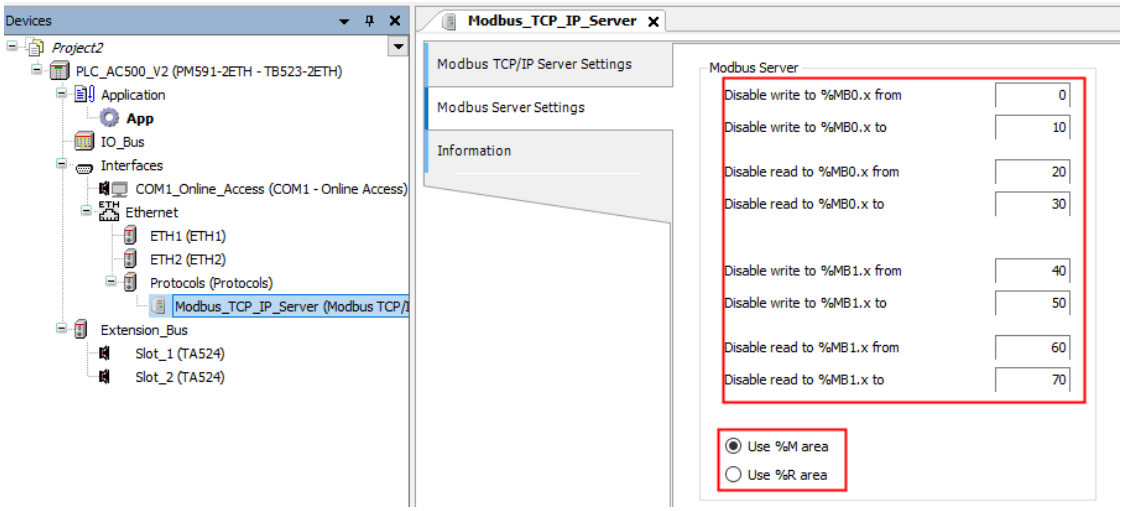
6.1.1.1 Configuration

In V2 it is necessary to add a “Modbus_TCP_IP_Server” for both: server and client functionality.
The number of reserved server sockets, task timeout and OpenModBus time can be set in the “Modbus TCP/IP Sever Settings” tab



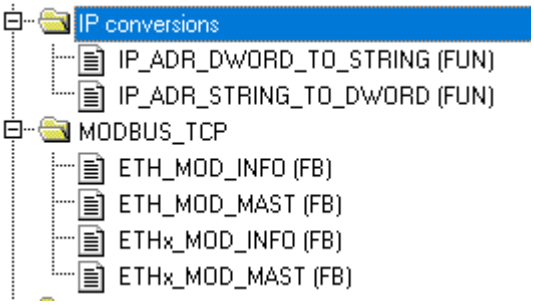
It is possible to protect two memory areas from access (read / write) of other clients access, but it is not possible to prevent the access of the PLC itself.

As the V2 PLCs have a special %R (retain) memory this can be chosen to be used instead of the %M memory area for the Modbus access.



6.1.1.2 Function blocks

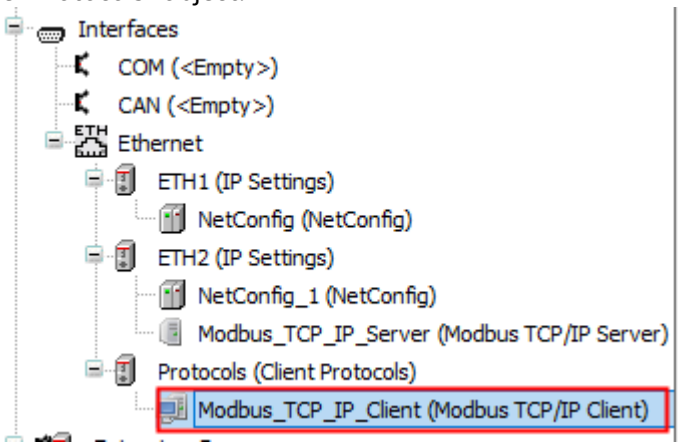
The following main function blocks can be used to create Modbus jobs to read/write from other servers:



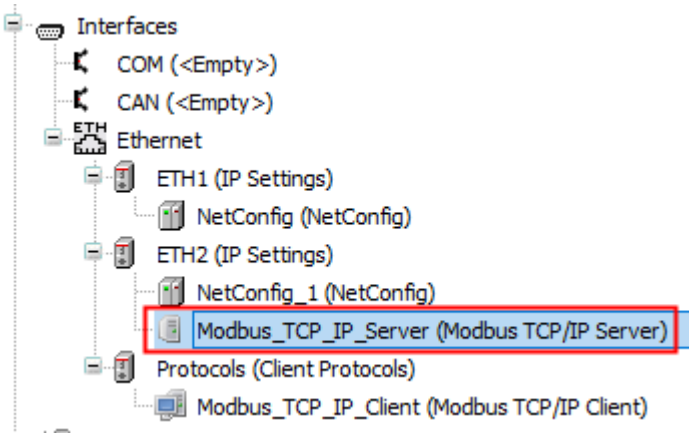
6.1.2 AC500 V3

6.1.2.1 Configuration

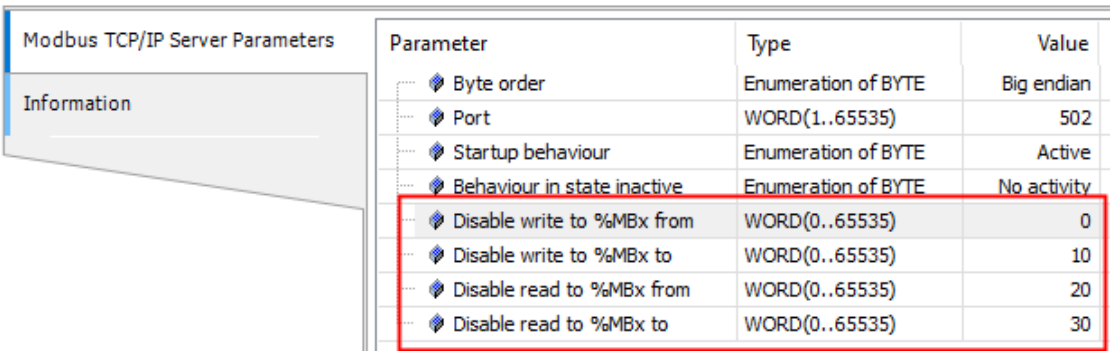
To configure the PLC as a Modbus client, the “Modbus_TCP_IP_Client” must be added below the “Protocols” object.



To configure the PLC as a Modbus server, the “Modbus_TCP_IP_Server” must be added below the appropriate ethernet port *ETH1* or *ETH2* (default after conversion is *ETH2*!).



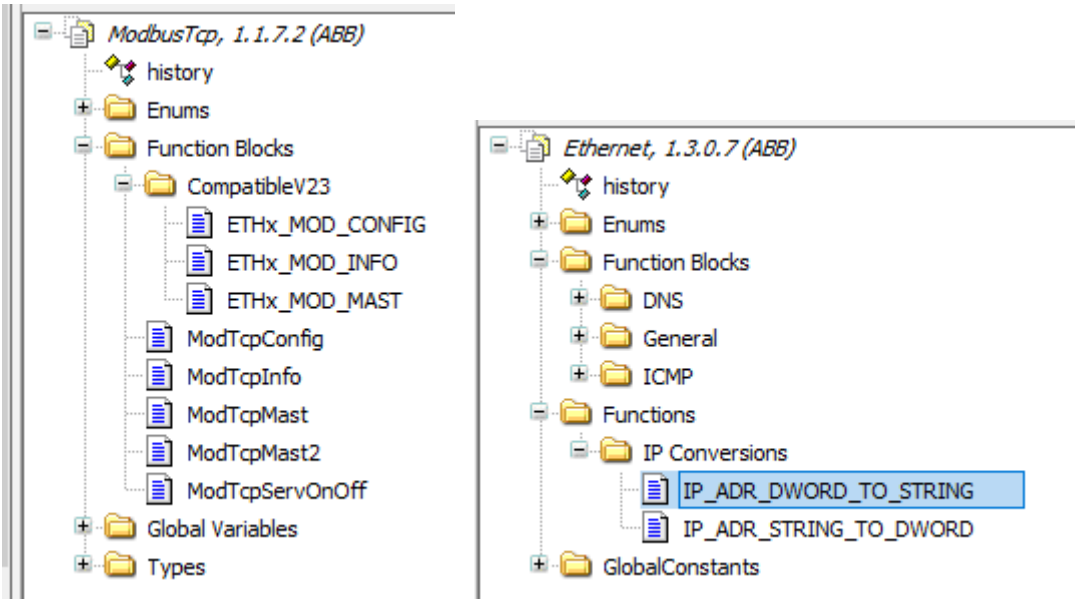
Two memory areas (read / write) can be protected from access from other clients.



Other configurations are made via function blocks ModTcpConfig, ModTcpMast2 and ModTcpServOnOff. See chapter 6.1.2.2 below for details.

6.1.2.2 Function blocks

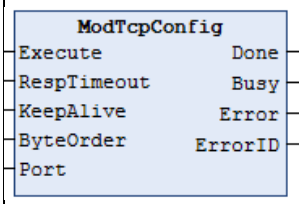
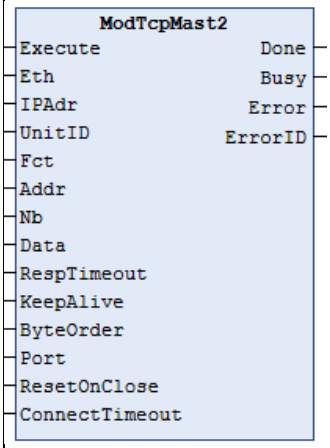
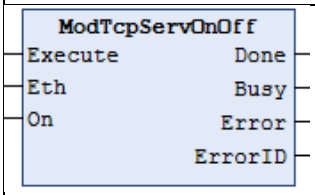
The following main function blocks can be used to create Modbus jobs to read/write from other servers:



The function blocks beneath the folder “Compatiblev23” can be used directly in the same way as the have been used in AC500 V2.

The ETH_MOD_xx are not available. They must be replaced by the ETHx_MOD_xx, where the only difference is the input “ETH” instead of “Slot”.

Additional functionality and configuration is given by the following function blocks.

<table>
<tr><th>Function block</th><th>Remark / part of documentation</th></tr>
<tr><td> 
</td><td>Configuration of Modbus TCP for whole PLC
<table>
<tr><td rowspan="4">Input</td><td>RespTimeout</td><td>WORD</td><td>Slave response timeout in ms. Maximum time waiting for a response after having sent a request</td></tr>
<tr><td>KeepAlive</td><td>DWORD</td><td>Open Modbus Time in ms. A connection to a Server stays open after sending a request
until the time is expired. If there is another request within this time, the timer is restarted. On expiration the connection is closed.</td></tr>
<tr><td>ByteOrder</td><td>CAA.ENDIANESS</td><td>Endianess of data, little or Big Endian; Modbus Standard is Big Endian</td></tr>
<tr><td>Port</td><td>WORD</td><td>TCP Port</td></tr>
</table></td></tr>
<tr><td> 
</td><td>Function block serves to send Modbus on TCP/IP requests to a server and evaluate the response.
<table>
<tr><td rowspan="7">Input</td><td>RespTimeout</td><td>WORD</td><td>Slave response timeout in ms. Maximum time waiting for a response after having sent a request</td></tr>
<tr><td>KeepAlive</td><td>DWORD</td><td>Open Modbus Time in ms. A connection to a Server stays open after sending a request
until the time is expired. If there is another request within this time, the timer is restarted. On expiration the connection is closed.</td></tr>
<tr><td>ByteOrder</td><td>CAA.ENDIANESS</td><td>Endianess of data, little or Big Endian; Modbus Standard is Big Endian</td></tr>
<tr><td>Port</td><td>WORD</td><td>TCP Port</td></tr>
<tr><td>ResetOnClose</td><td>BOOL</td><td>Reset connection when closing it (TRUE) or gaceful close (FALSE) like ModTcpMast</td></tr>
<tr><td>ConnectTimeout</td><td>DWORD</td><td>Connect timeout [ms]. Value 0 corresponds to OS default</td></tr>
</table></td></tr>
<tr><td> 
</td><td>Function block serves to set TCP server to active/passive.
<table>
<tr><td rowspan="2">Input</td><td>Eth</td><td>BYTE</td><td>Index number of the assigned interface</td></tr>
<tr><td>On</td><td>BOOL</td><td>Switch on (TRUE) or off (FALSE) server</td></tr>
</table></td></tr>
</table>

6.1.3 Summary of steps for migration from AC500 V2 to AC500 V3

- Add the “MODBUS_TCP_CLIENT” below the “Protocols” object, if Modbus client is needed.
- Check the ethernet port for the server (default is ETH2, should be probably ETH1) if Modbus server is needed.
- In the IEC-Program change the ETH_MOD_yy function blocks to ETHx_MOD_yy function blocks, which are available in same style as in AC500 V2.
- To configure the Modbus behavior of PLC use function block “ModTcpConfig”
  See chapter 6.1.2.2

- To configure the behavior of dedicated server connections, use the function block “ModTcpMast2”, which has special inputs to configure the connection. See chapter 6.1.2.2
- To enable or disable the server functionality of a port use the function block “ModTcpServOnOff”. See chapter 6.1.2.2

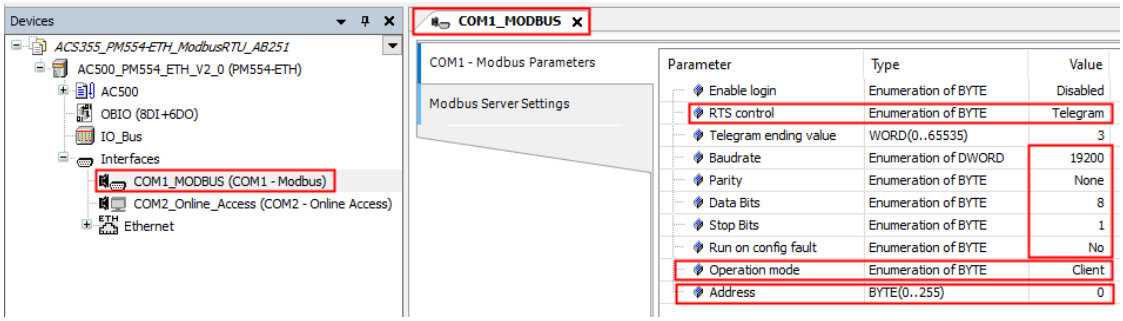
6.2 Modbus RTU

The main difference regarding the use of Modbus in AC500 V3 is the difference in the memory structure and therefore in the absolute addressing of the Modbus variables already described in chapter 2.1.5 “Addressable ” and chapter 2.1.6 “Modbus addresses / Byte order”. Tipps for the conversion of the Modbus addresses are in chapter 5.2.2 “Modbus addresses %M and %R”.

6.2.1 Modbus RTU HW configuration

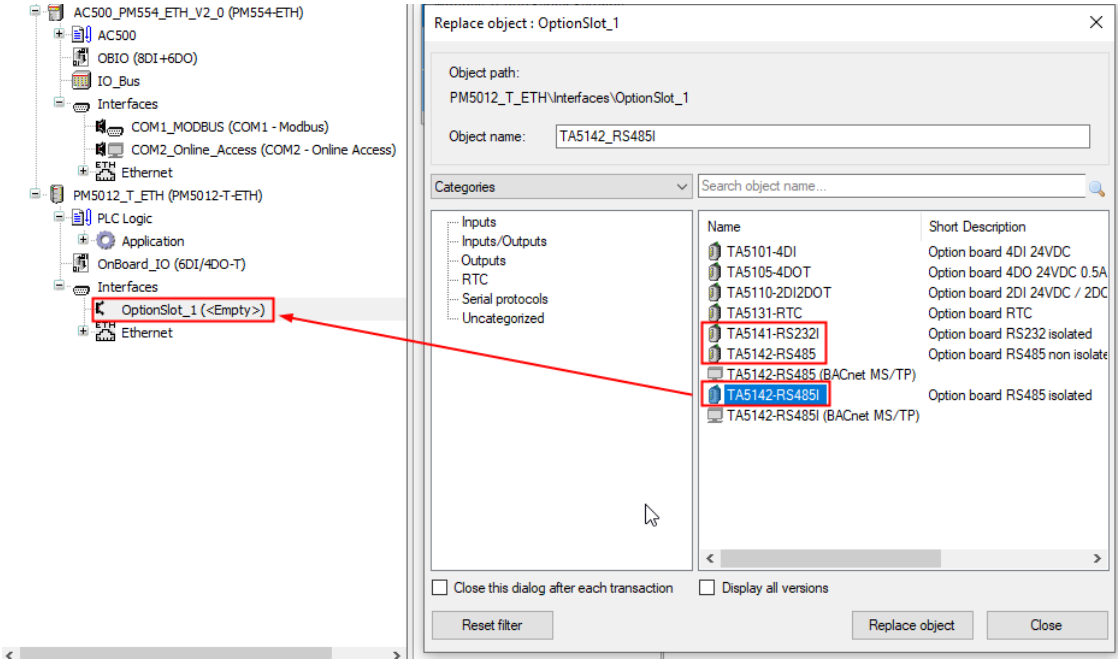
To change from an AC500 V2 PLC with Modbus RTU to an AC500 V3 PLC, the following steps must be performed after the change of the CPU type from AC500 V2 to AC500 V3.

1. Note the communication parameters of the AC500 V2 PLC



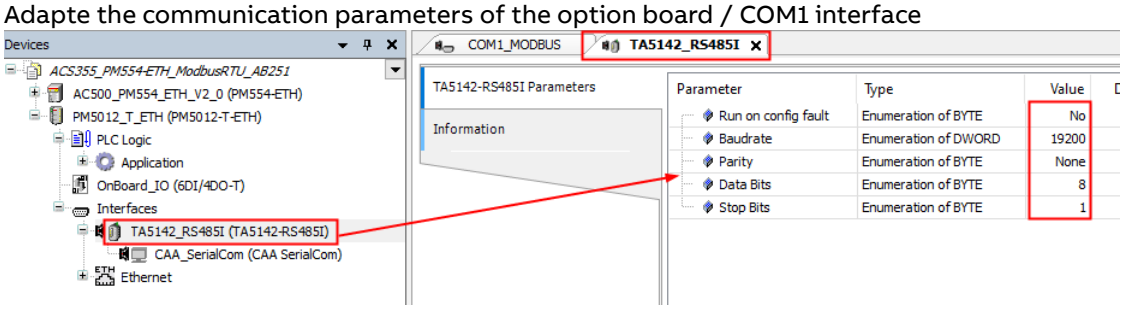
RTS control = Telegram ➔ RS485 is used. Otherwise RS232.
Client mode has always the address “0”

2. For AC500-**eCo V3** insert an **option board** for RS232 or RS485 (with or without isolation) on the dedicated Option Slot.

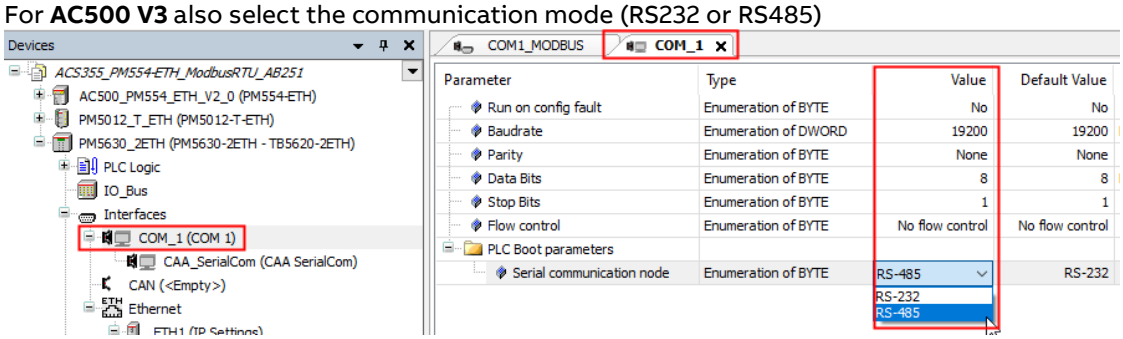


For **AC500 V3** insert **COM1** for the COM interface

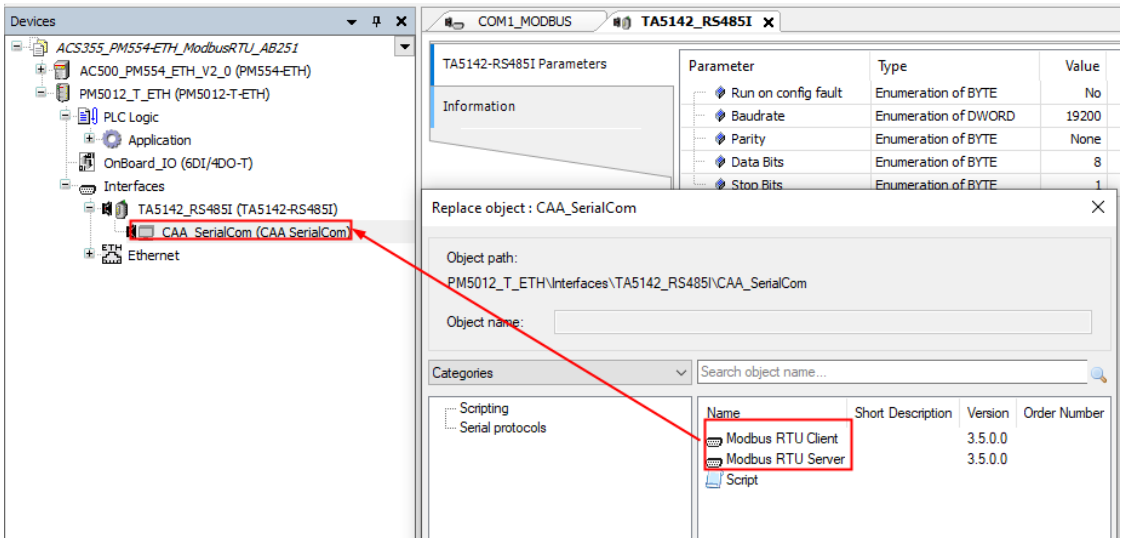
3. Adapte the communication parameters of the option board / COM1 interface

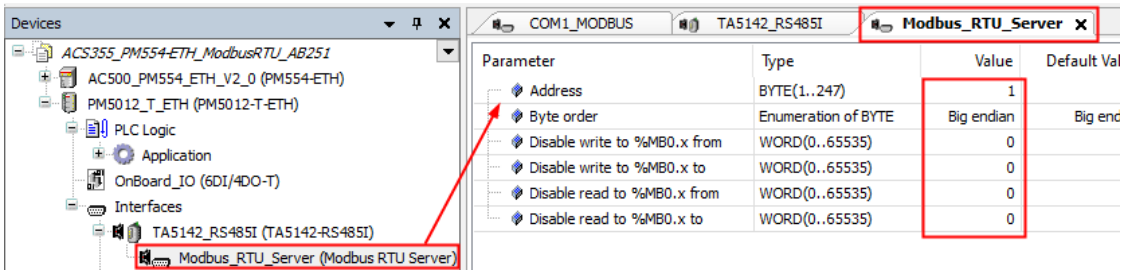


For **AC500 V3** also select the communication mode (RS232 or RS485)

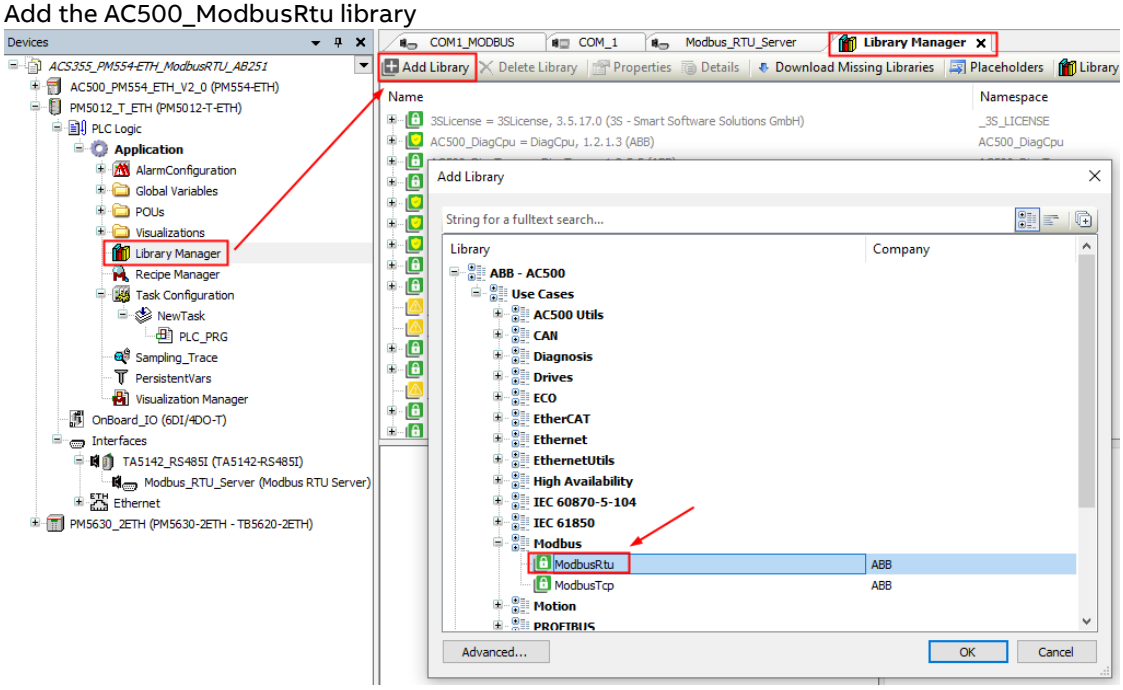


4. Add a Modbus client or Modbus server.
   If a Modbus server is used the address has to set and the restricted data areas, if needed.





5. Add the AC500_ModbusRtu library

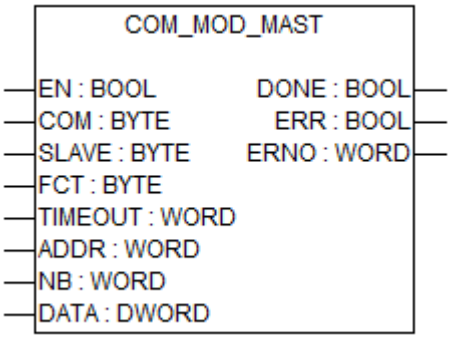


6.2.2 Modbus RTU IEC program adaption

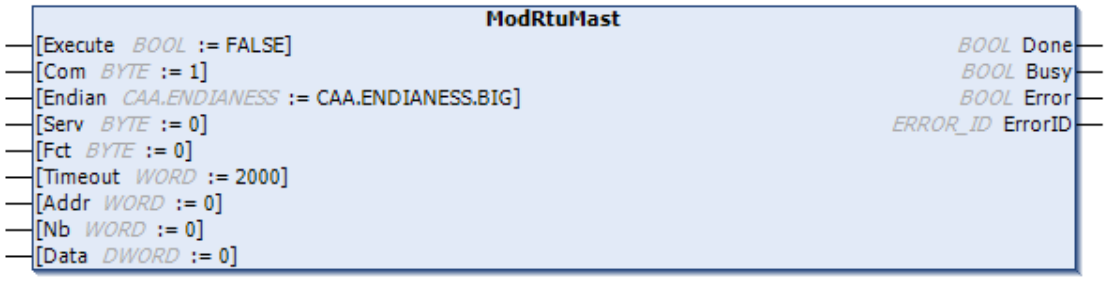
6.2.2.1 Modbus RTU Function blocks

The master function block in AC500 V3 is similar to AC500 V2, but in PLCopen style and has an additional input “Endian” to select the endinaness (byte order)

AC500 V2:

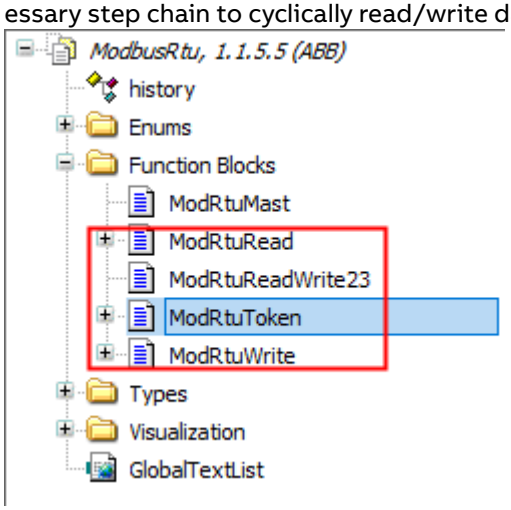


AC500 V3:

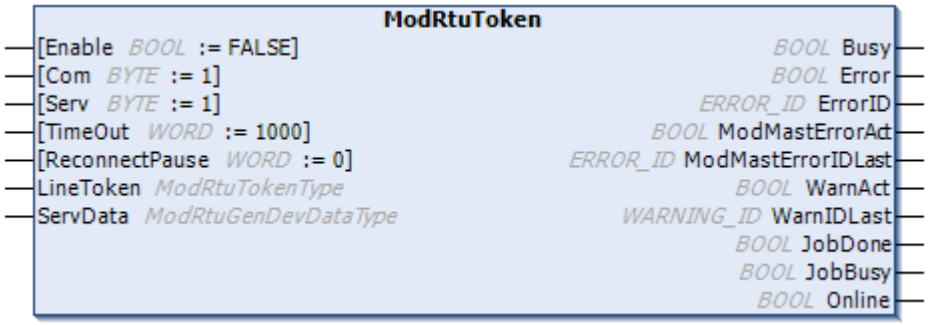


Additional function blocks in AC500 V3:

The AC500 V3 library contains some more valuable function blocks that encapsulate the necessary step chain to cyclically read/write data from one or several servers:



Using the ModRtuToken together with one or more of the blocks ModRtuRead, ModRtuWrite or ModRtuReadWrite23 establishes cyclic read/write jobs of all servers connected to the same COM line.



Connection between the different servers is made via the common variable connected to the input “LineToken”.

Connection to the read/write blocks within one server is made via the common variable “ServerData”.

6.2.2.2 Modbus data in AC500 V3

If the AC500 V3 is used as a Modbus server, the absolute addressing of the variables need to be checked / changed.

If special conversion of received or send data were made in AC500 V2, those have to be checked due to the different byte order in AC500 V3 PLCs.

See chapter 2.1.5 “Addressable ” and chapter 2.1.6 “Modbus addresses / Byte order”

6.3 Profinet

6.3.1 General

All devices and couplers are kept as long as a AC500 V3 CPU is selected, that has enough communication modules slots available (no AC500-eCo V3).

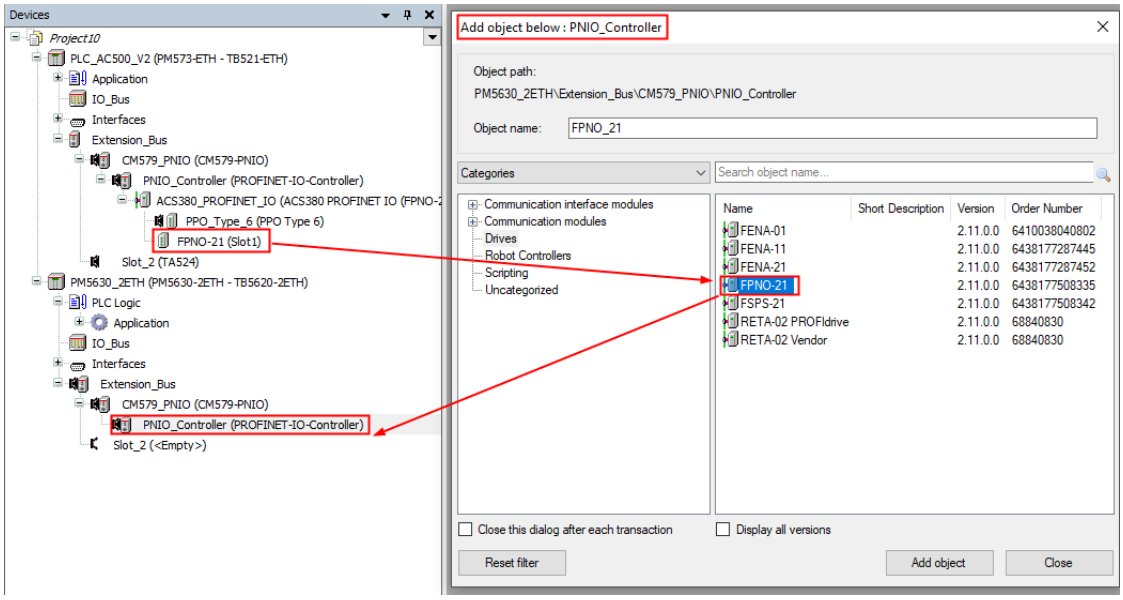
Exceptions are:

- There are no DriveManager editors, as they are not supported by the AC500 V3 PLCs. See chapter 6.3.2 how to keep the data exchange functionality.
- **CI504 and CI506** are not yet supported by AC500 V3.

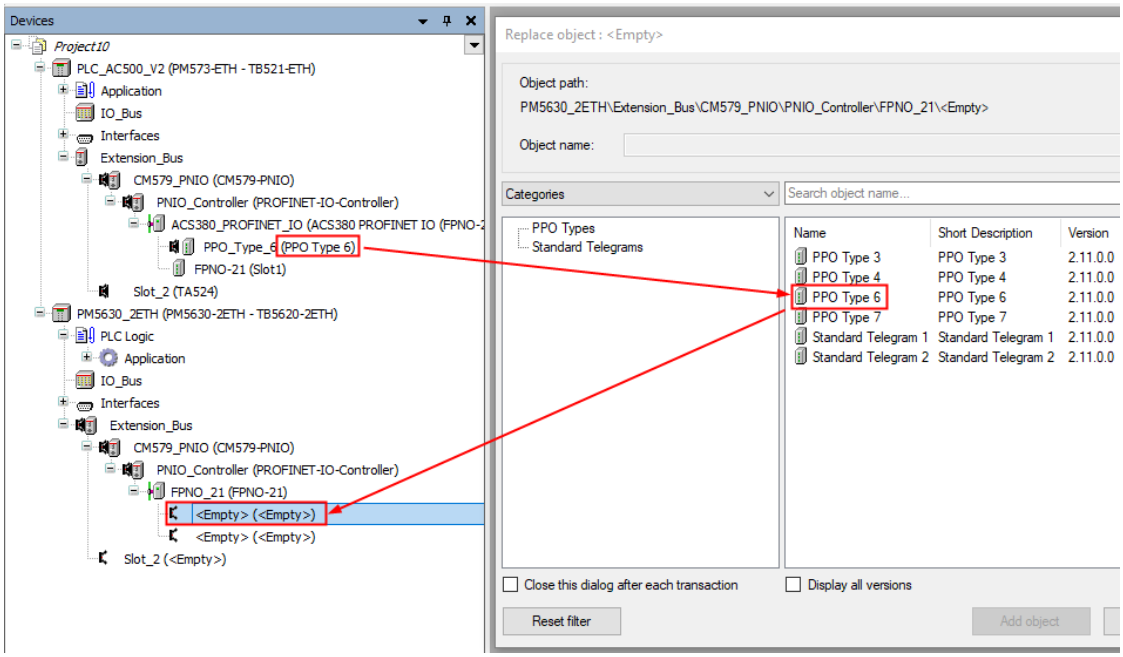
6.3.2 DriveManager

The DriveManager elements for ACS/DCS drives are not supported by AC500 V3.
To keep the data exchange functionality, it´s necessary to do some manual steps for each drive.

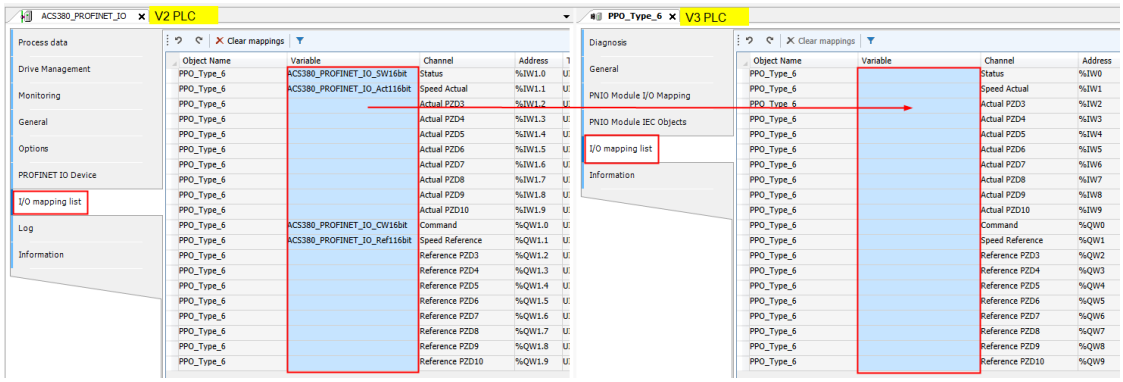
Insert a FPNO-21 or FENA-x1 node below the Profinet_Controller according to what was selected in the AC500 V2 PLC.



Then add the same PPO-Type as used in the AC500 V2 PLC to the FPNO-21 or FENA-x1 node.

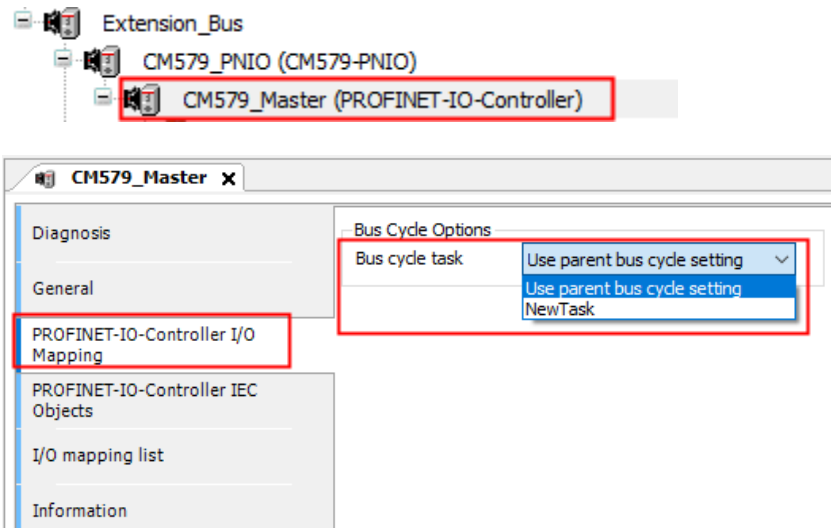


Then copy all variable mappings from the AC500 V2 PLC to the AC500 V3 PLC.
Best to use the I/O mapping list.
Mark the whole column with all variables, copy them, go to the V3 I/O mapping list, mark the whole column and paste the variables.



6.3.3 Bus cycle

The bus cycle can be adapted in AC500 V3 to a specific task. This will be set by the parameter “Bus cycle task” of the Profinet Master IO-Controller



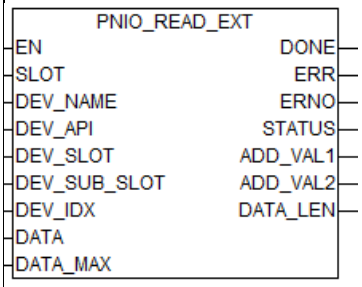
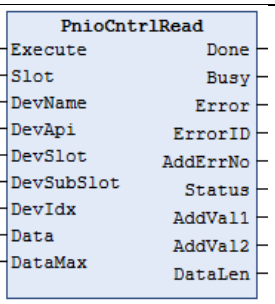
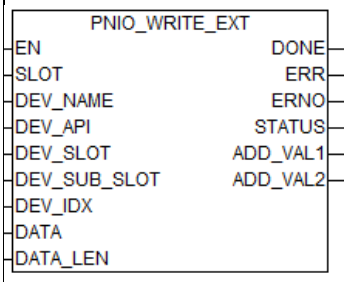
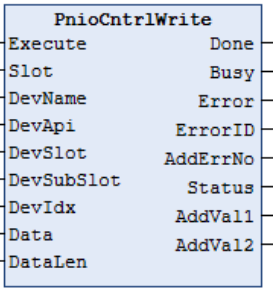
6.3.4 Cyclic data exchange

After the conversion from AC500 V2 to AC500 V3 PLC the Profinet controller and device functionality regarding it´s cyclic data exchange is kept as in V2 and can be used without any changes, as long as all needed variables are mapped and used within the IEC program code.

Absolute addresses might change and should not be used in the IEC program code.

6.3.5 Acyclic data exchange

The acyclic data exchange needs to be adapted slightly as the function blocks have changed to PLCopen style and have some improvements in outputs Status, AddVal1 and AddVal2.

V2	V3
	
	

6.3.6 Bus control

The bus control function blocks have kept the functionality but are now available in PLCopen style.

V2	V3
PNIO_CNTL_START_COM EN, SLOT, TOUT DONE, ERR, ERNO, ADD_ERNO	PnioCntrlStartCom Execute, Slot Done, Busy, Error, ErrorID, AddErrNo
PNIO_CNTL_STOP_COM EN, SLOT, RESET_IO, TOUT DONE, ERR, ERNO, ADD_ERNO	PnioCntrlStopCom Execute, Slot Done, Busy, Error, ErrorID, AddErrNo

6.3.7 Diagnosis

The AC500 V3 diagnosis system gives detailed information about the bus state and S500 module messages (see chapter 2.5). Additional bus scans can be done via Automation Builder.

The Profinet diagnosis function block, which are used in AC500 V2 must be adapted to the function blocks in AC500 V3, which are now available in PLCopen style and partly include functionality of multiple blocks from AC500 V2 in one block in AC500 V3.

V2	V3
PNIO_STATE EN : BOOL, SLOT : BYTE DONE : BOOL, ERR : BOOL, ERNO : WORD, MST_STATE : PNIO_MST_STATE_TYPE, COMM_ERNO : PNIO_COMM_ERNO_TYPE, NUM_ERRS : DWORD	PnioCntrlGetCntrlState Execute, Slot Done, Busy, Error, ErrorID, AddErrNo, CommunicationCOS, MstState, CommErno, NumErrs, Version, Watchdog, SlaveState, NumConfigSlaves, NumActiveSlaves, NumDiagSlaves
PNIO_SYS_DIAG EN, SLOT DONE, ERR, ERNO, DEV_STATE, NUM_CFG, NUM_ACT, NUM_DIAG	

PNIO_DEV_INFO_EXT EN, SLOT, DEV_NAME DONE, ERR, ERNO, MAC, IP, VENDOR, DEVICE, ALARM, ACTIVE, FLAGS	PnioCntrlGetDevState Execute, Slot, DevName Done, Busy, Error, ErrorID, AddErrNo, Mac, Ip, Vendor, Device, Alarm, Active, Flags
PNIO_IM0 EN, SLOT, DEV_NAME, DEV_SLOT DONE, ERR, ERNO, VENDOR_ID, ORDER_ID, SER_NO, HW_REV, SW_REV, REV_CNT, PROFILE_ID, PROFILE_SPEC, IM_VERSION, IM_SUPPORTED	PnioCntrlGetDevIM0Data Execute, Slot, DevName, DevSlot Done, Busy, Error, ErrorID, AddErrNo, VendorId, OrderId, SerNo, HwRev, SwRev, RevCnt, ProfileId, ProfileSpec, ImVersion, ImSupported
PNIO_DEV_ALARM EN, SLOT, DEV_NAME, DATA DONE, ERR, ERNO, API, DEV_SLOT, DEV_SUB_SLOT, DEV_MODULE, DEV_SUB_MOD, PRIO, ALARM_TYPE, SPECIFIER, STRUCT_ID, DATA_LEN, NEW, ACTIVE PNIO_DEV_SPECIFIER SPECIFIER SEQUENCE, CHANNEL, MANUFACT, SUB_MOD, RES, AR_STATE	Please use standard diagnose of node name. <node_name>.diag... Or use generic function block from dignosis library
PNIO_DEV_DIAG EN, SLOT, DEV_NAME, DATA DONE, ERR, ERNO, FLAGS, DATA_LEN	Please use standard diagnose of node name. <node_name>.diag... Or use generic function block from dignosis library

6.4 Profibus

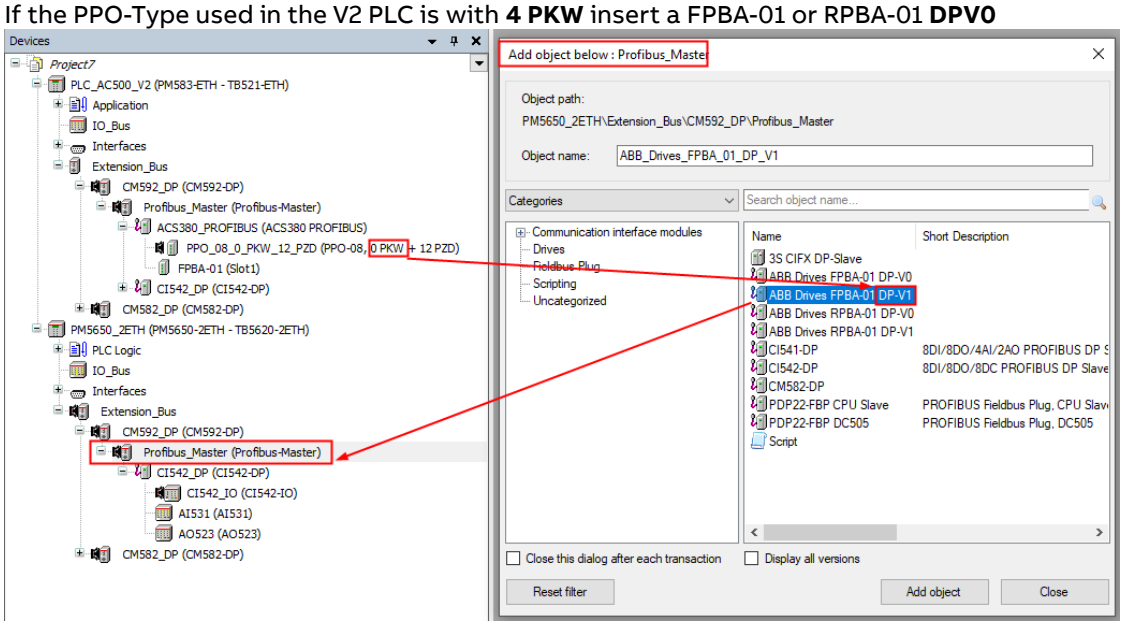
6.4.1 General

All devices and couplers are kept as long as a AC500 V3 CPU is selected, that has enough communication modules slots available (no AC500-eCo V3). The only exception is that there are no DriveManager editors, as they are not supported by the AC500 V3 PLCs. See chapter 6.4.2 how to keep the data exchange functionality.

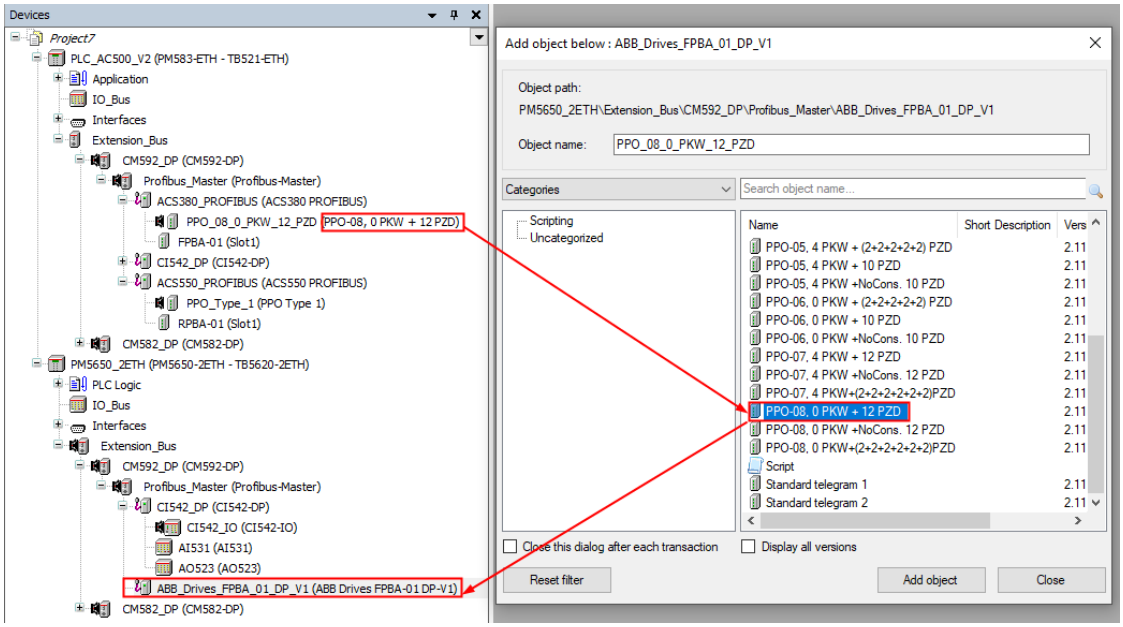
6.4.2 DriveManager

The DriveManager elements for ACS/DCS drives are not supported by AC500 V3.
To keep the data exchange functionality it´s necessary to do some manual steps for each drive.

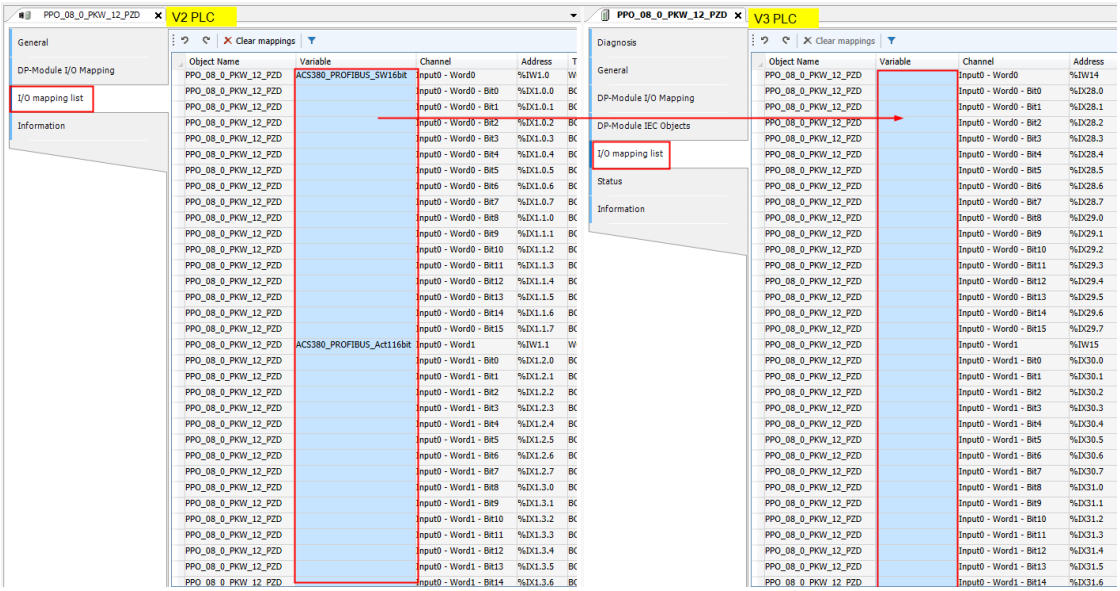
Insert a FPBA-01 or RPBA-01 node below the Profibus_Master.
If the PPO-Type used in the V2 PLC is with **0 PKW** insert a FPBA-01 or RPBA-01 **DPV1**
If the PPO-Type used in the V2 PLC is with **4 PKW** insert a FPBA-01 or RPBA-01 **DPV0**



Then add the same PPO-Type as used in the V2 PLC to the FPBA-01 or RPBA-01 node.

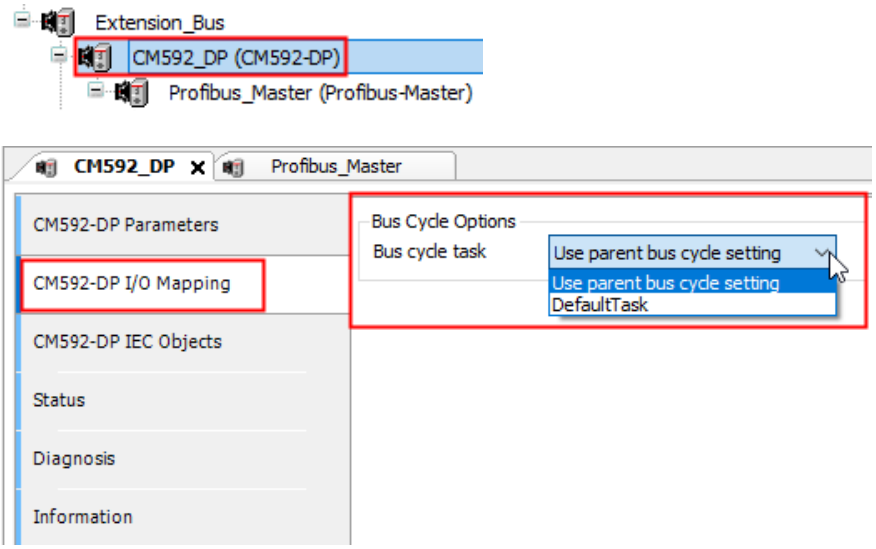


Then copy all variable mappings from the AC500 V2 PLC to the AC500 V3 PLC.
Best to use the I/O mapping list.
Mark the whole column with all variables, copy them, go to the AC500 V3 I/O mapping list, mark the whole column and paste the variables.



6.4.3 Bus cycle

The bus cycle can be adapted in AC500 V3 to a specific task. This will be set by the parameter “Bus cycle task” of the CM592_DP in the CM592-DP I/O Mapping section.



6.4.4 Cyclic data exchange

After the conversion from AC500 V2 to AC500 V3 PLC the Profibus master and slave functionality regarding it´s cyclic data exchange is kept as in AC500 V2 and can be used without any changes, as long as all needed variables are mapped and used within the IEC program code.

Absolute addresses might change and should not be used in the IEC program code.

6.4.5 Acyclic data exchange

The acyclic data exchange needs to be adapted slightly as the function blocks have changed to PLCopen style.

V2	V3
DPV1_MSAC1_READ EN : BOOL SLOT : BYTE SLV : BYTE SLV_SLOT : BYTE SLV_IDX : BYTE LEN : BYTE DATA : DWORD DONE : BOOL ERR : BOOL ERNO : WORD ERNO1 : BYTE ERNO2 : BYTE DATA_LEN : BYTE	CM592DPV1Masc1Read Execute Device SlaveAddr SlaveSlot SlaveIndex DataLen Data Done Busy Error ErrorID AddErrorID ErrNo1 ErrNo2 DataLenRead
DPV1_MSAC1_WRITE EN : BOOL SLOT : BYTE SLV : BYTE SLV_SLOT : BYTE SLV_IDX : BYTE LEN : BYTE DATA : DWORD DONE : BOOL ERR : BOOL ERNO : WORD ERNO1 : BYTE ERNO2 : BYTE DATA_LEN : BYTE	CM592DPV1Masc1Write Execute Device SlaveAddr SlaveSlot SlaveIndex DataLen Data Done Busy Error ErrorID AddErrorID ErrNo1 ErrNo2 DataLenWrite
DPM_READ_INPUT EN : BOOL SLOT : BYTE SLV : BYTE DATA : DWORD DONE : BOOL ERR : BOOL ERNO : WORD DATA_LEN : BYTE	CM592ReadInput Execute Device SlaveAddr DataLen Data Done Busy Error ErrorID AddErrorID DataLenRead
DPM_READ_OUTPUT EN : BOOL SLOT : BYTE SLV : BYTE DATA : DWORD DONE : BOOL ERR : BOOL ERNO : WORD DATA_LEN : BYTE	CM592ReadOutput Execute Device SlaveAddr DataLen Data Done Busy Error ErrorID AddErrorID DataLenRead

6.4.6 Bus control

The bus control function blocks have kept the functionality but are now available in PLCopen style.

V2	V3
DPM_CTRL EN : BOOL SLOT : BYTE SLV : BYTE GROUP_SEL : BYTE CLR_DATA : BOOL UNFREEZE : BOOL FREEZE : BOOL UNSYNC : BOOL SYNC : BOOL DONE : BOOL ERR : BOOL ERNO : WORD	CM592Control Execute Device SlaveAddr GroupSelect Unfreeze Freeze Unsync Sync Done Busy Error ErrorID AddErrorID

6.4.7 Diagnosis

The V3 diagnosis system gives detailed information about the bus state and S500 module messages (see chapter 2.5). Additional bus scans can be done via Automation Builder.

The profibis diagnosis function blocks, which are used in AC500 V2 must be adapted to the function blocks in AC500 V3, which are now available in PLCopen style.

V2	V3
DPM_SYS_DIAG EN : BOOL SLOT : BYTE TYP : BYTE DONE : BOOL ERR : BOOL ERNO : WORD SLV : ARRAY [0..127] OF BOOL	CM592SystemDiagnosis Enable Device SlaveStateType Done Busy Error ErrorID AddErrorID SlaveStates
DPM_STAT EN : BOOL SLOT : BYTE DONE : BOOL ERR : BOOL ERNO : WORD STATE_BITS : DPM_STATE_BITS_TYPE DPM_STATE : BYTE COM_ERR : DPM_COM_ERR_TYPE BUS_ERR : WORD TIME_OUT : WORD	CM592CommStatus Enable Device Done Busy Error ErrorID AddErrorID StateInfo CommState CommError NumBusError NumTimeout
DPM_SLV_DIAG EN : BOOL SLOT : BYTE SLV : BYTE DONE : BOOL ERR : BOOL ERNO : WORD STAT_1 : STATIONSTATUS_1_TYPE STAT_2 : STATIONSTATUS_2_TYPE STAT_3 : STATIONSTATUS_3_TYPE MSTR : BYTE EXT_DIAG_LEN : BYTE EXT_DIAG_DAT : ARRAY [1..238] OF BYTE	CM592SlaveDiagnosis Execute Device SlaveAddr Done Busy Error ErrorID AddErrorID Status_1 Status_2 Status_3 MasterAddr ExtDiagLen ExtDiagData
	Also possiblel to us <node_name>.diag…

6.5 CS31 Bus connections

There is no CS31-Bus available in AC500 V3 nor AC500-eCo V3 CPUs yet.

Please check possible availability in the future.

For the time being the IO-Modules used on a CS31 Bus must be transferred to another decentralized communication interface.

Possible Solutions:

1. Using Profibus CI541-DP or CI542-DP with AC500 V3 (serial cable might be kept)

2. Using Modbus TCP CI521-MODTCP or CI522-MODTCP with AC500-eCo V3 or AC500-eCo (can be used with eCo. Needs special engineering)
3. Using Profinet CI501-PNIO or CI502-PNIO with AC500 V3

7 Reestablishing communication to SCADA devices

7.1 Symbol File

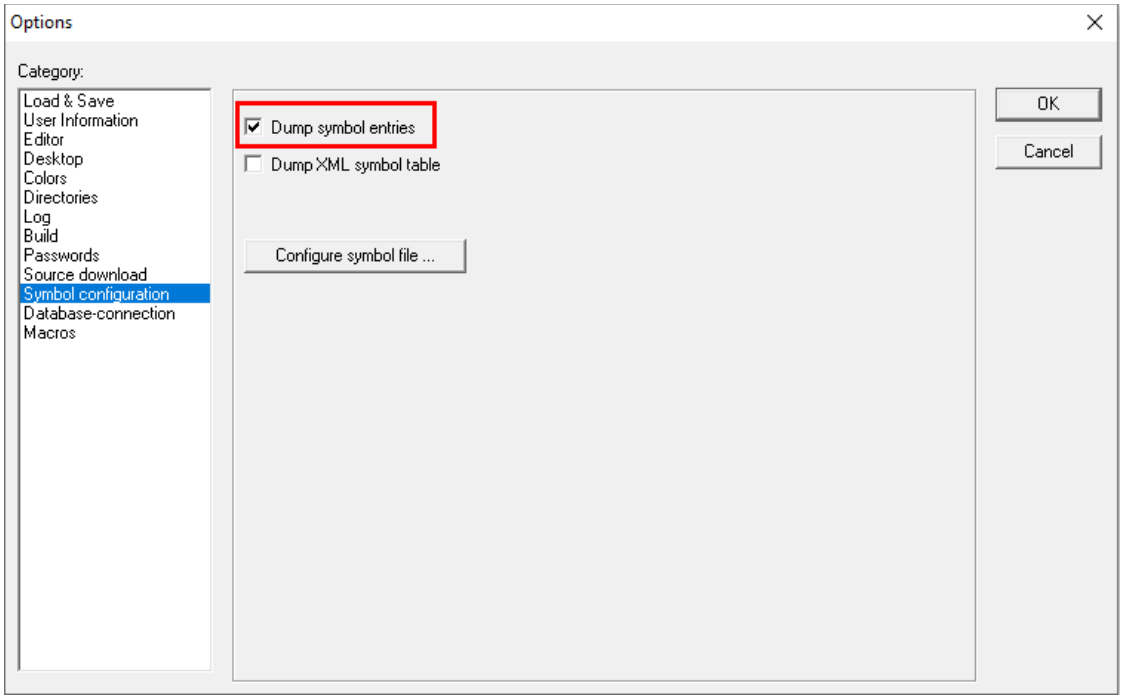
7.1.1 AC500 V2 PLC

	This chapter only describes the steps how to configure the symbol files for AC500 V2 PLCs. The next chapter 7.1.2 describes the configuration for AC500 V3 PLCs.

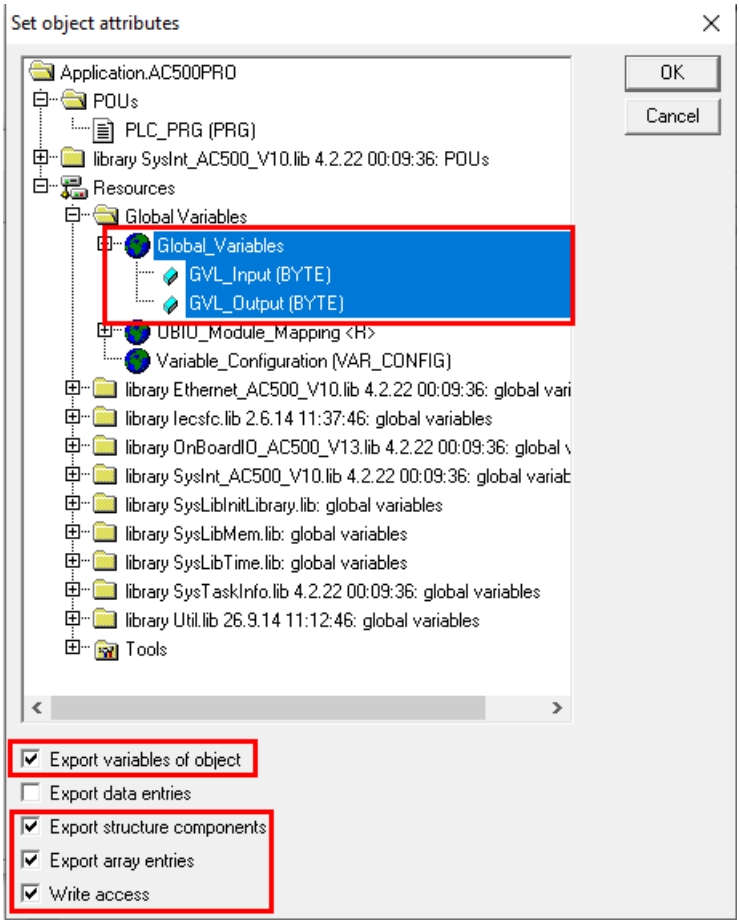
Configure Symbol File

Symbol includes the items(variables) which exchanges with PLC, this is needed for OPC communication with Panel, SCADA, etc.

Open the Symbol configuration (Project->Options->Symbol configuration) to choose ‘Dump symbol entries’, and click the button ‘Configure symbol file…’



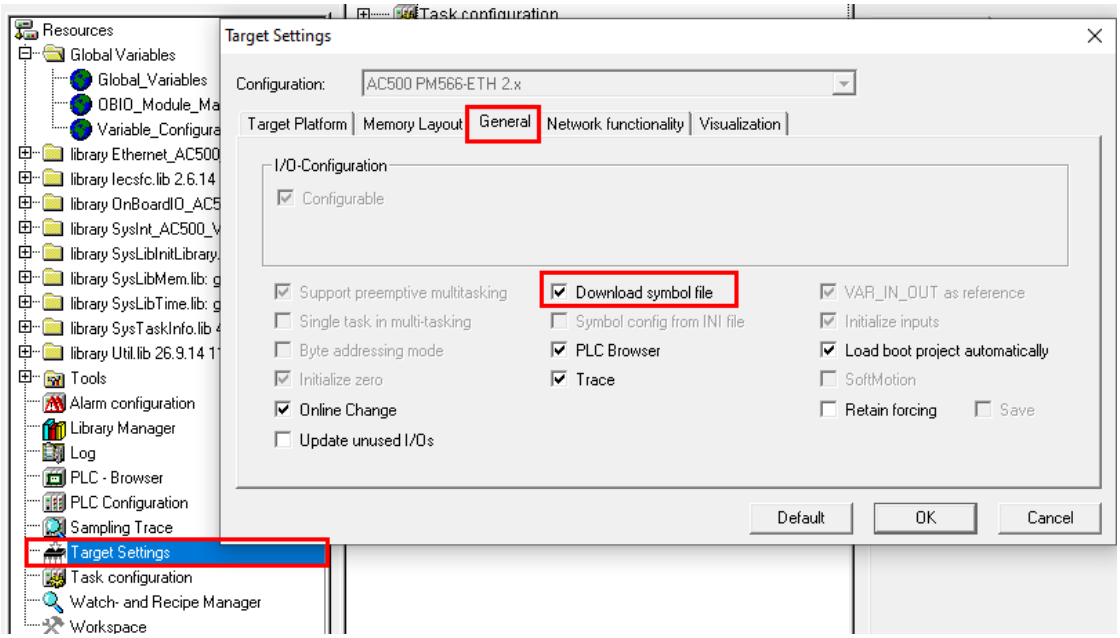
Only choose the variables which should be communicated as symbol. Then check the options except for the item ‘Export data entries ‘



Confirm the settings with pushing the OK button. Then we can rebuild the project.

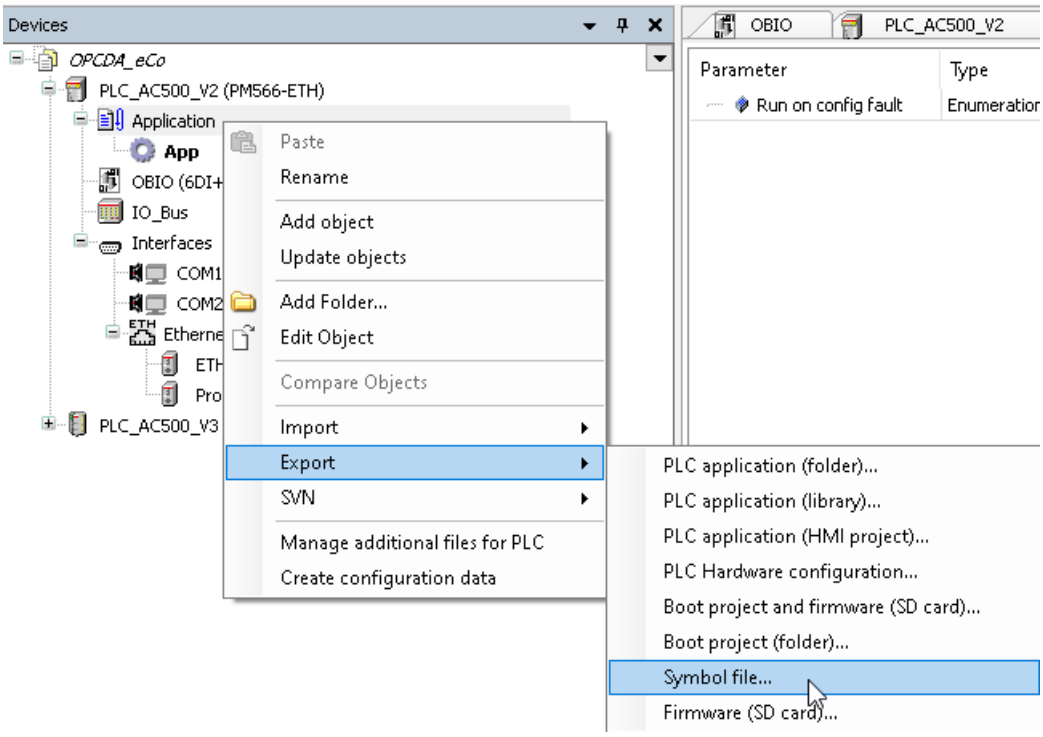
Download Symbol file to PLC

Open the target settings from resources tab and check the ‘Download symbol file ‘.

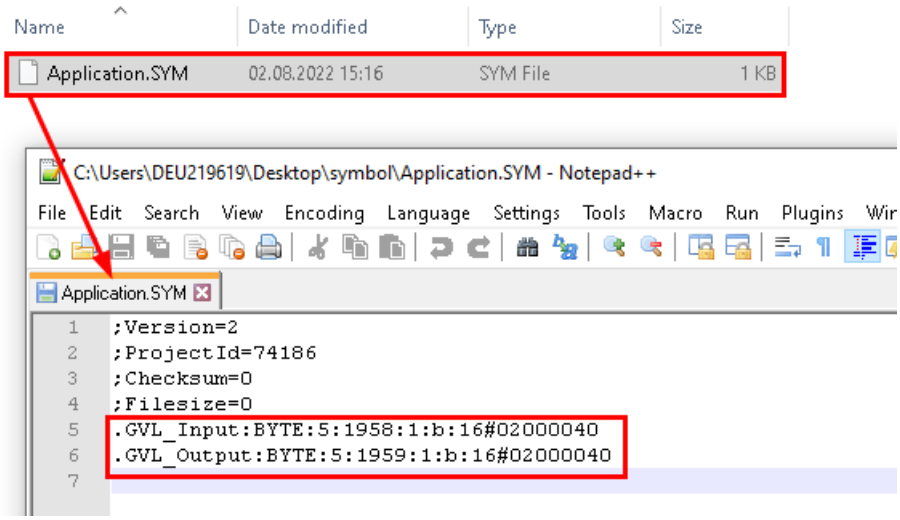


Go online and download the project into PLC. After that save and close the Codesys IEC application editor.

We can export the symbol file with text format which can be understood by engineer.



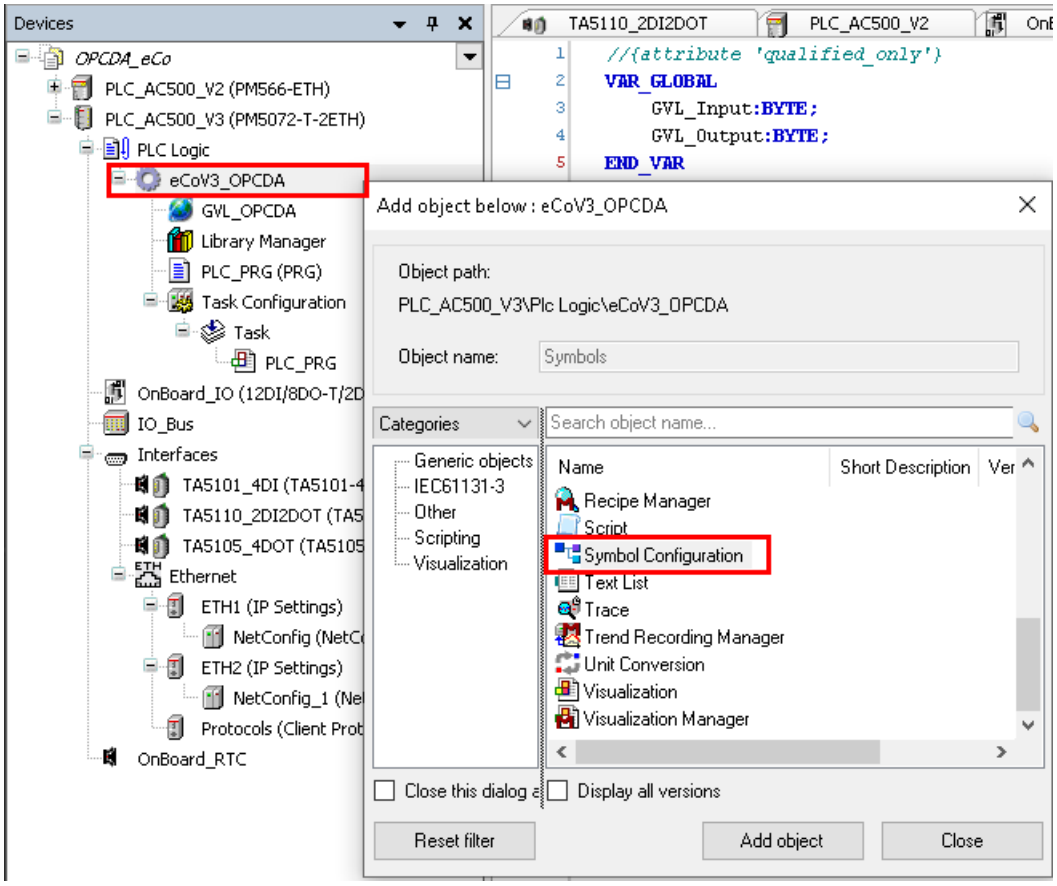
Then we can check if the symbol file is generated correctly.



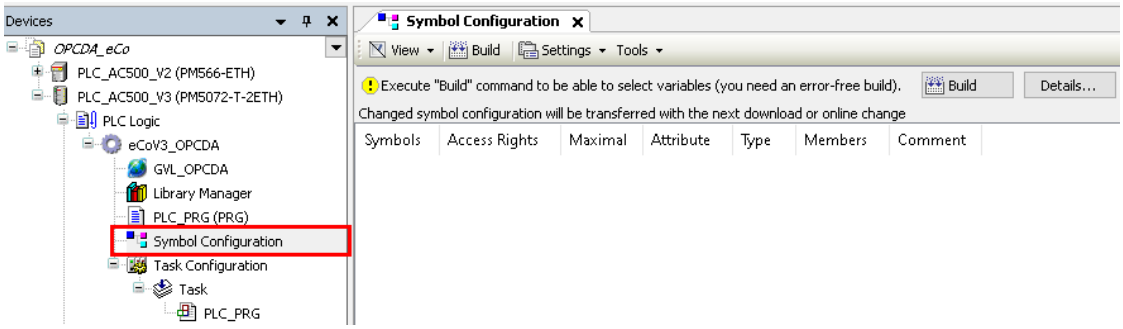
	Sometimes the symbol file looks different than really configured, e.g. more symbols than expected. In such a case please repeat the steps again.

7.1.2 AC500 V3 PLC

Add the object Symbol Configuration to the Application object in the Automation Builder device tree.

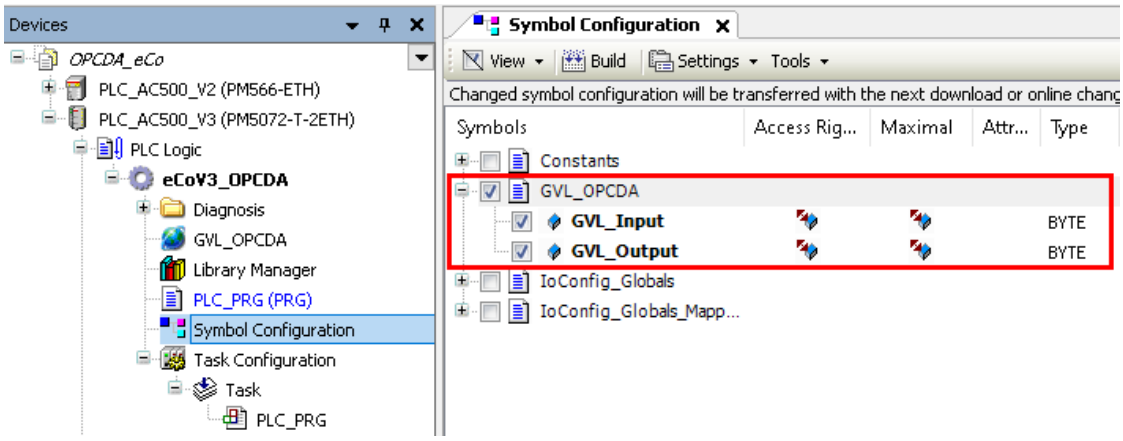


The symbol configuration editor appears. Keep the default settings and click Add button.

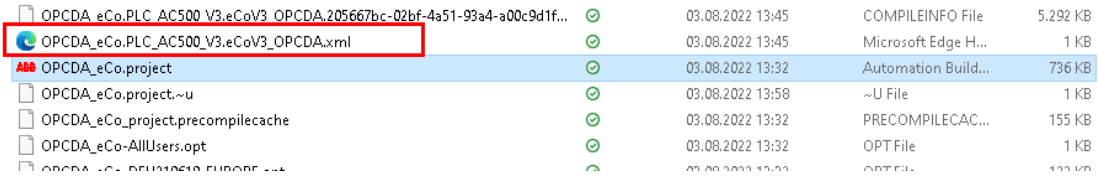


In symbol configuration editor, execute ‘Build’ command to be able to select variables.

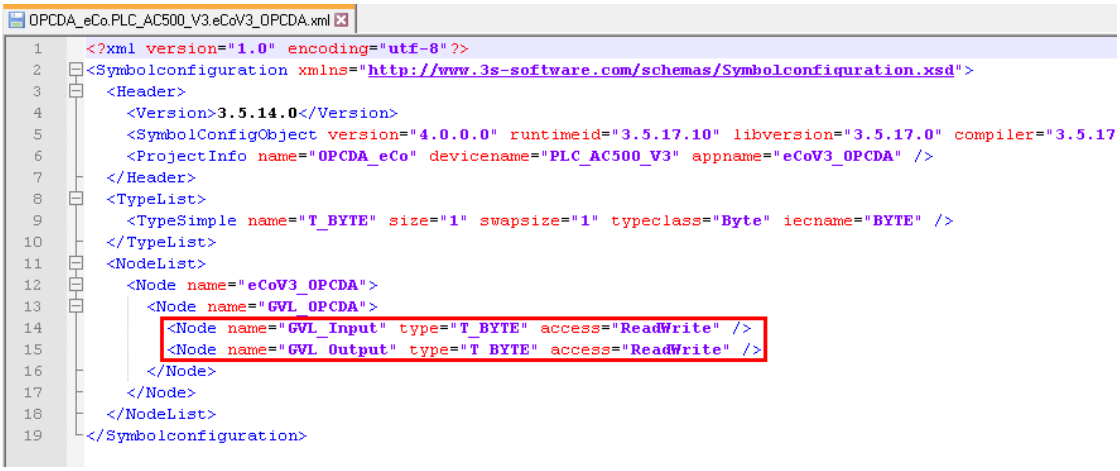
Then all the project items(variables) will be displayed in the window. Select only the desired items. These are needed e.g. for communication with a CP600 Panel, OPC DA or OPC UA client.



Go online and download the project into PLC. The symbol file is automatically generated by a build command of an AC500 V3 project when a symbol configuration exists and is stored with the name extension XML next to the project file.



The symbol file can be opened with any text editor and checked if the entries are correct.



7.2 CP600

When changing the PLC from AC500 V2 to AC500 V3 the protocol and the symbol file or addresses are changed. In this chapter it is explained, how to update the Panel Builder project to establish the communication with the existing project to the AC500 V3 PLC.

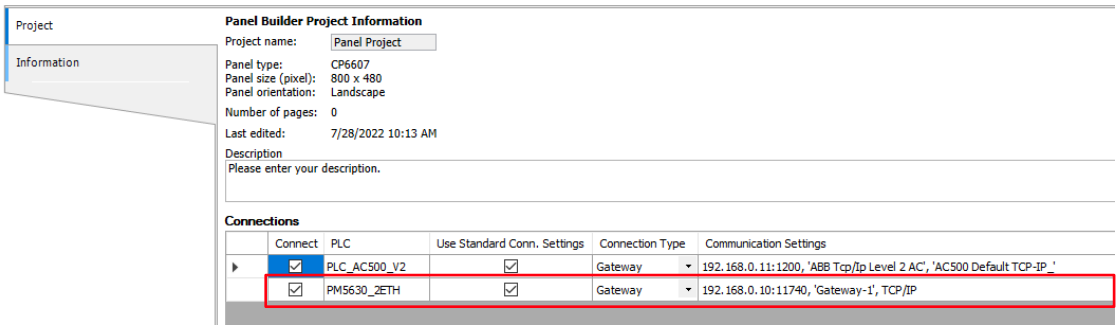
	CAUTION! Before starting the conversion of the tags from AC500 V2 protocol to AC500 V3 protocol a copy of the Panel Builder / Automation Builder project should be saved as backup.

	There are two possibilities how to change from an AC500 V2 protocol to an AC500 V3 protocol. 1. Renaming the used tags from AC500 V2 style to AC500 V3 style 2. Replacing the AC500 V2 tags by AC500 V3 tags in the widgets The Renaming of the tags, like described in chapter 7.2.1, is only working if the V2 tags have a path like in ABB CoDeSys ETH protocol. If there is no path only the replacing is working. The renaming should be used whenever there are many tags used in the project. **In case only a few tags are used in the project the recommendation is to replace the tags like described in chapter 7.2.2**

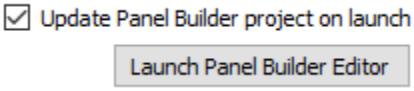
The following two chapters describe the steps to rename or replace the existing variables. This is described exemplarily on the basis of ABB CoDeSys ETH protocol and ABB Modbus TCP protocol. The described steps are also working with all other protocols.

7.2.1 Renaming tags e.g. CoDeSys V2 ETH to CODESYS V3 ETH protocol

1. Update the project from AC500 V2 PLC to AC500 V3 PLC as described in chapter 5
2. If not already done, create a symbol file as described in chapter 7.1.2
3. If the Panel Builder project is inside the Automation Builder project
   a) Open Panel_CP600 from the devices tree
   b) Make sure the AC500 V3 PLC connection is checked



   c) Check Update Panel Builder project on launch and click Launch Panel Builder Editor



4. If the Panel Builder project is separate from the Automation Builder project

   a) Open Panel Builder

   b) Go to Protocols and click 

   c) In the drop down list in column PLC select CODESYS V3 ETH

   d) Set the IP address and confirm with OK

   e) Go to Tags

   f) Select CODESYS V3 ETH protocol

   g) Click Import Dictionary 

   h) Select CODESYS3 xml Hierarchical

   i) Browse to the symbol file saved next to the Automation Builder project

   j) Select the symbol file and click open

5. Go to the Tags in Panel Builder. Here the old ABB CoDeSys ETH is inside as well as the new CODESYS V3 ETH. In both protocols the same tag names are existing. But the path to these tags is different.

   AC500 V2 Tags

.arrStData[9].bStatus	UINT	PLC_AC500_V2//arrStData[9]/bStatus
.arrStData[9].rValue	REAL	PLC_AC500_V2//arrStData[9]/rValue
.arrStData[9].xStart	BOOL	PLC_AC500_V2//arrStData[9]/xStart
.strMessage	STRING	PLC_AC500_V2//strMessage
.xReadData	BOOL	PLC_AC500_V2//xReadData
HMI_PRO.iRetainedVar	INT	PLC_AC500_V2/HMI_PRO/iRetainedVar
HMI_PRO.sRetainedString	STRING	PLC_AC500_V2/HMI_PRO/sRetainedString
PLC_PRG.lrVar	LREAL	PLC_AC500_V2/PLC_PRG/lrVar
PLC_PRG.udiVar	UDINT	PLC_AC500_V2/PLC_PRG/udiVar

   AC500 V3 Tags

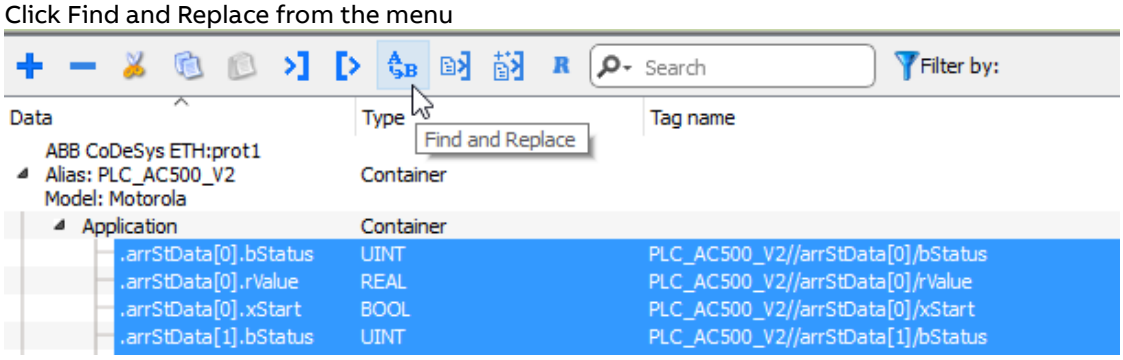
[9]	Container	
bStatus	UINT	PM5630_2ETH/Application/Global_Variables/arrStData[9]/bStatus
rValue	REAL	PM5630_2ETH/Application/Global_Variables/arrStData[9]/rValue
xStart	BOOL	PM5630_2ETH/Application/Global_Variables/arrStData[9]/xStart
strMessage	STRING	PM5630_2ETH/Application/Global_Variables/strMessage
xReadData	BOOL	PM5630_2ETH/Application/Global_Variables/xReadData
HMI_PRO	Container	
iRetainedVar	INT	PM5630_2ETH/Application/HMI_PRO/iRetainedVar
sRetainedString	STRING	PM5630_2ETH/Application/HMI_PRO/sRetainedString
PLC_PRG	Container	
lrVar	LREAL	PM5630_2ETH/Application/PLC_PRG/lrVar
udiVar	UDINT	PM5630_2ETH/Application/PLC_PRG/udiVar

> The following description is focusing on the update where the AC500 V2 PLC, the AC500 V3 PLC and the Panel Builder project are all in the same Automation Builder project.
>
> If they are not in the same project and no alias names are set the PLC names might be missing. The following steps can be done 1 to 1 just the names are different

6. Rename global variables
   V2 Global variables start with PLC_AC500_V2//
   V3 Global variables start with PM5630_2ETH/Application/Global_Variables/

   a) Select all tags in the AC500 V2 protocol which are global

b) Click Find and Replace from the menu



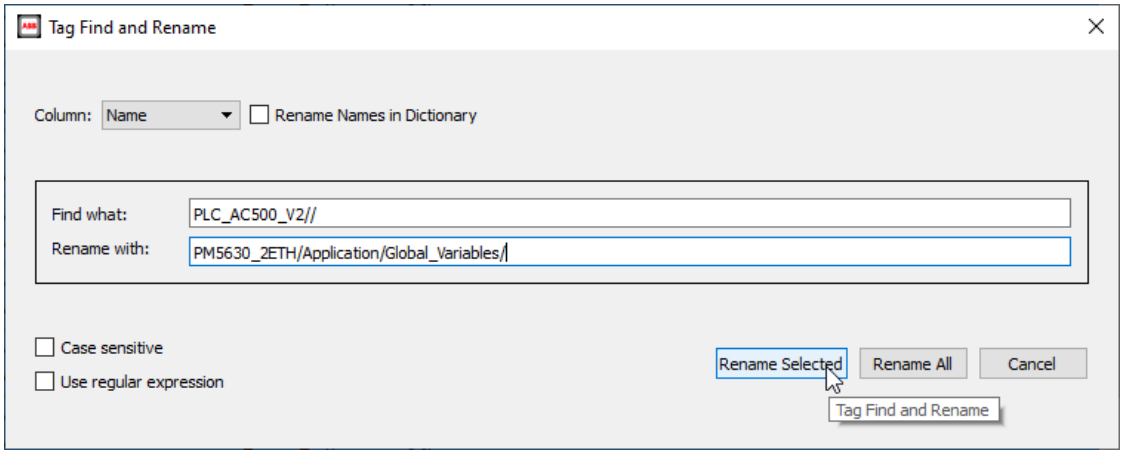
c) Confirm with Yes

d) Find what: PLC_AC500_V2//

e) Rename with: PM5630_2ETH/Application/Global_Variables/

	The expressions above have to be adapted to the naming used in the project

f) Click Rename Selected



7. Repeat with the local variables

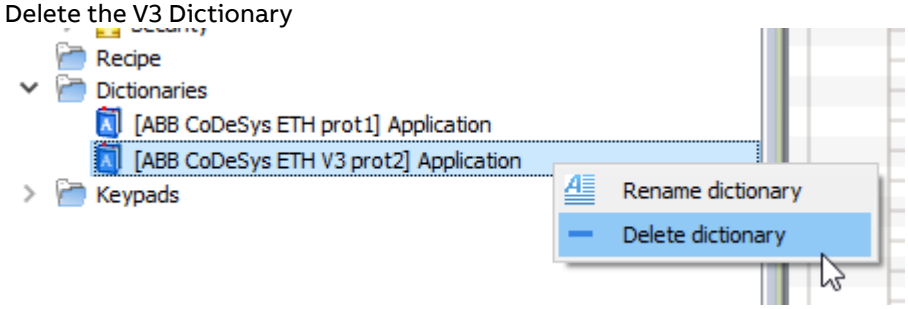
   Find what: PLC_AC500_V2/

   Rename with: PM5630_2ETH/Application/

8. Move AC500 V2 variables to AC500 V3

   a) Please recheck that all variables below ABB CoDeSys ETH have the same names as in CODESYS V3 ETH

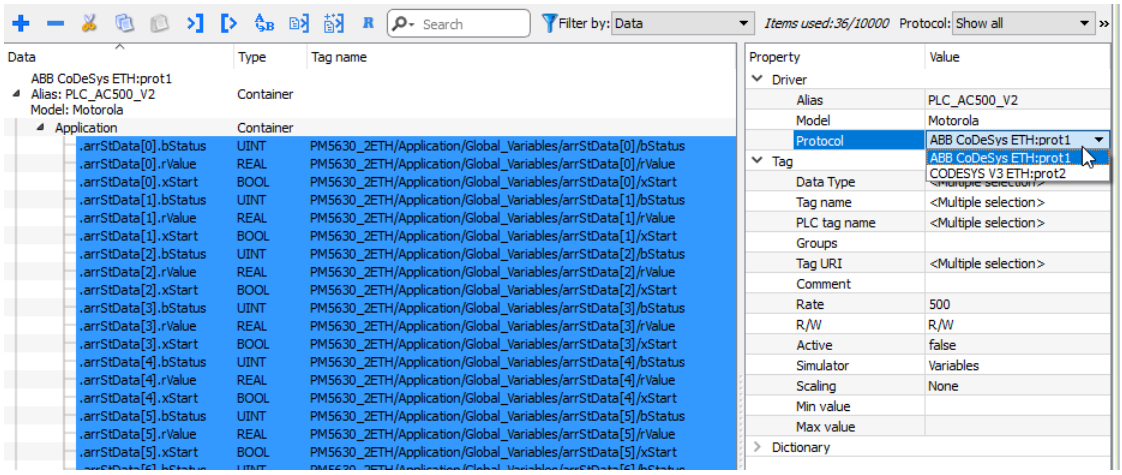
   b) Delete the V3 Dictionary



   All tabs below CODESYS V3 ETH are gone now

   c) Select all AC500 V2 tags in ABB CoDeSys ETH

d) In the properties window on the right side. Change the protocol from ABB CoDeSys ETH to CODESYS V3



e) Acknowledge the pop up window with OK
All tags below ABB CoDeSys ETH are now grey
The tags can be found in CODESYS V3 ETH now

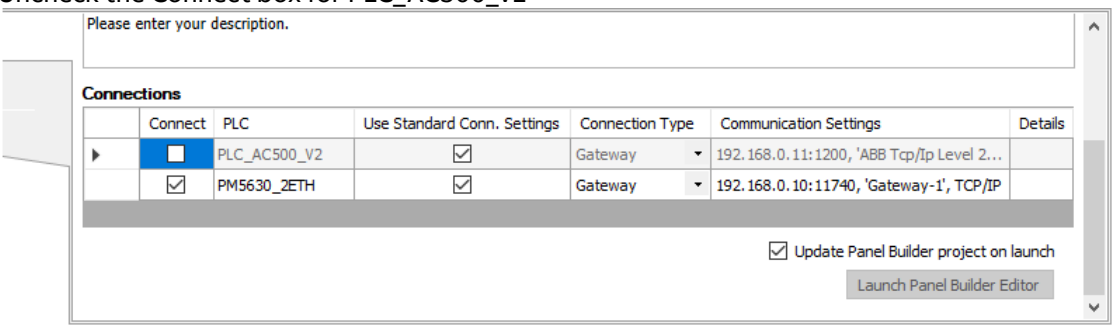
f) Recheck in any Page, that the PLC variables are still mapped correct

9. Reload symbol file using Panel Builder in an Automation Builder project

a) Save the Panel Builder project

b) Close the Panel Builder project

c) Uncheck the Connect box for PLC_AC500_V2



d) Check the Box Update Panel Builder project on launch and click Launch Panel Builder Editor

10. Reload Symbol File using Panel Builder as standalone

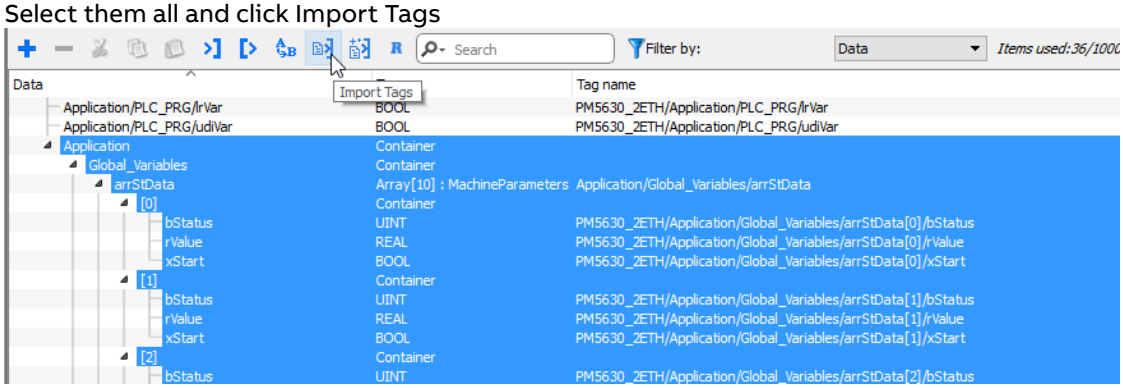
a) Click import dictionary 

b) Open the symbol file from the file system

11. Update tags

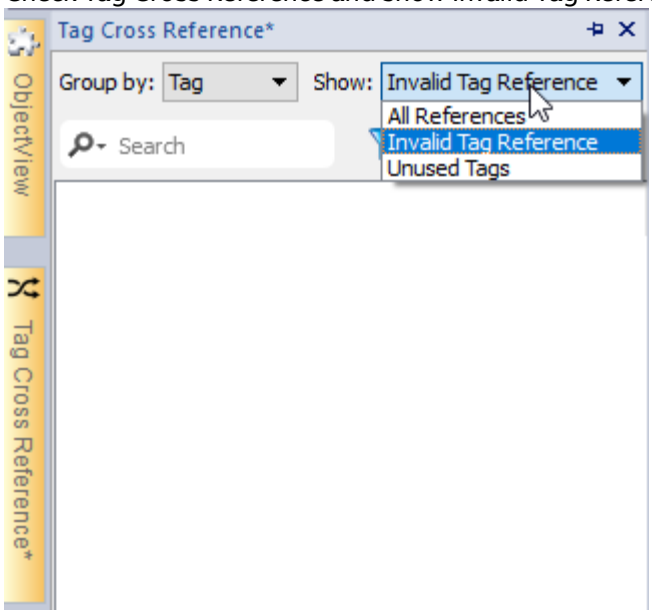
a) In the Tags view now several grey entries can be found

b) Select them all and click Import Tags



c) Confirm with Yes to All

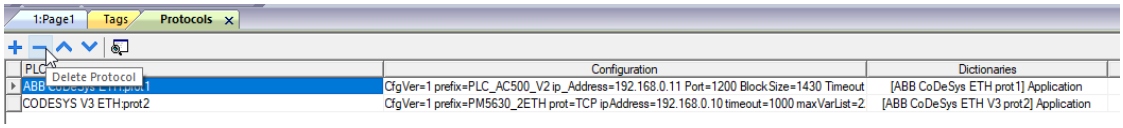
d) Check Tag Cross Reference and show Invalid Tag Reference



No entry should be in the list

12. Download the updated panel builder project to the CP600 and make sure it is communicating with the AC500 V3 PLC

13. Delete the ABB CoDeSys ETH protocol from the Protocols and download it again to the CP600

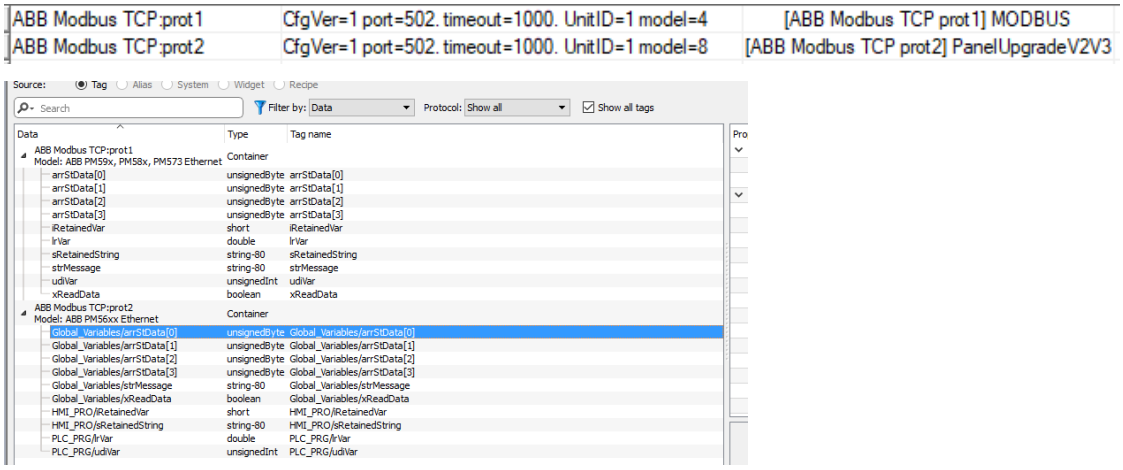


7.2.2 Replacing tags e.g. ABB Modbus TCP V2 to ABB Modbus TCP V3

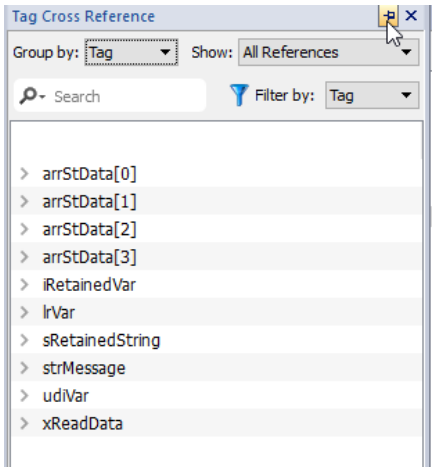
The workaround described in chapter 7.2.1 is only working with limitations for ABB Modbus. The steps can be done 1 to 1 as described for CODESYS ETH protocol. But the renaming of the tags is not possible, as there is no prefix like in the CODESYS ETH protocol. If following the steps 6 and 7, described in the last chapter, the adding of the path must be done for each tag by hand.

The recommended possibility is to not modify the tags by renaming but replace all used variables from AC500 V2 protocol to AC500 V3 protocol. In the description below the required steps for replacing the variables are described.

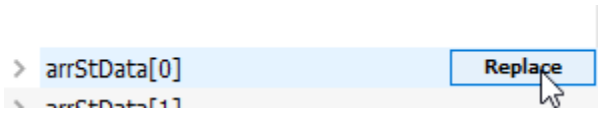
1. Add the Modbus TCP protocol and the tags via the export file to the project



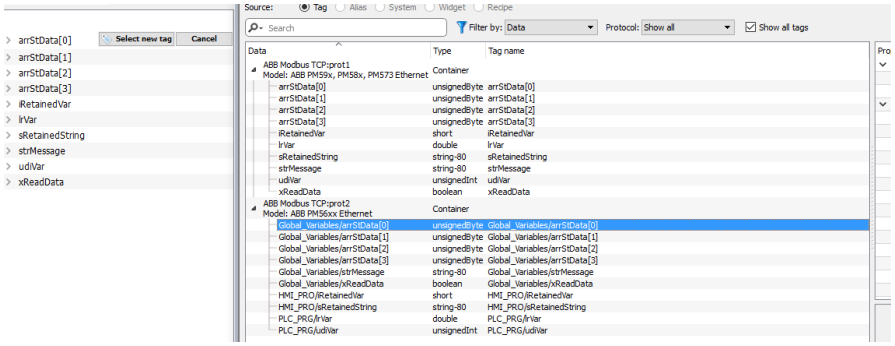
2. Open Tag Cross Reference View, pin it and click refresh view at the bottom 



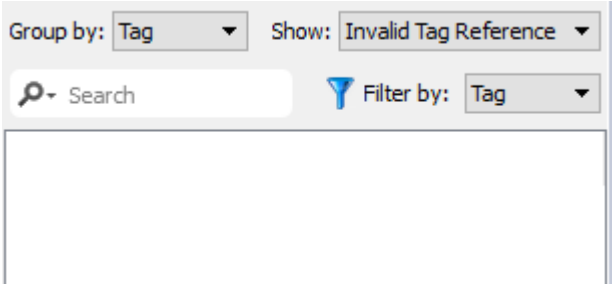
3. Select the first tag in the cross reference list and click Replace



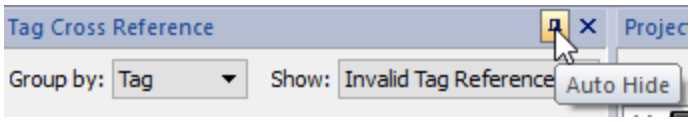
4. Click Select new tag and choose the corresponding V3 Tag from the dialog and confirm with OK



5. Repeat the last two steps above to replace all AC500 V2 tags by AC500 V3 tags
6. Make sure that all variables in the Tag Cross Reference (Show = All References) are AC500 V3 variables
7. Delete the AC500 V2 protocol from Protocols
8. Change the Cross Reference View from “All References” to “Invalid Tag Reference” and make sure, that there is no entry in the list



9. Hide the cross reference view again, download the project to the HMI and ensure that the communication with the AC500 V3 PLC is running as expected.



7.3 OPC DA

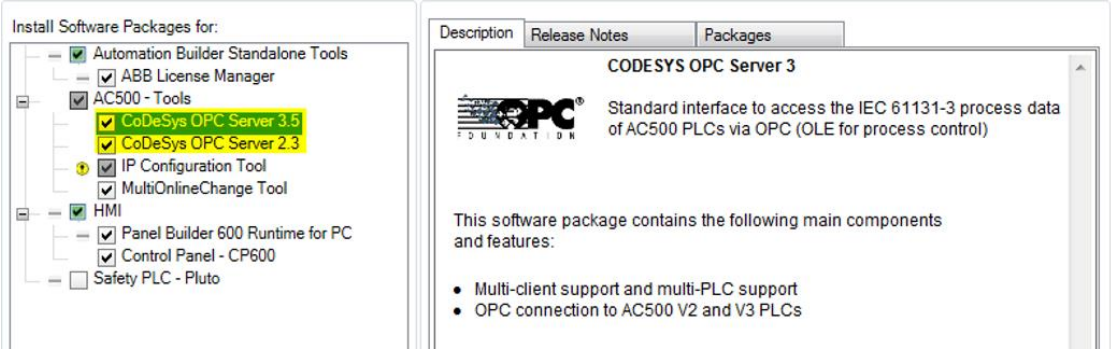
This chapter gives an introduction of the steps for the use of OPC DA with AC500 V2 and AC500 V3:

- OPC DA for AC500 V2, e.g. PM566-TP-ETH
- OPC DA for AC500 V3, e.g. PM5072-T-2ETH
- Update the project from AC500 V2 to AC500 V3

⚠	CAUTION! Before starting the conversion of the OPC DA from V2 to V3 a copy of the Panel Builder / Automation Builder project should be saved as backup.

7.3.1 OPC DA for AC500 V2

Install the OPC server AC500 V3 or AC500 V2 as additional tools with the Automation Builder installer.

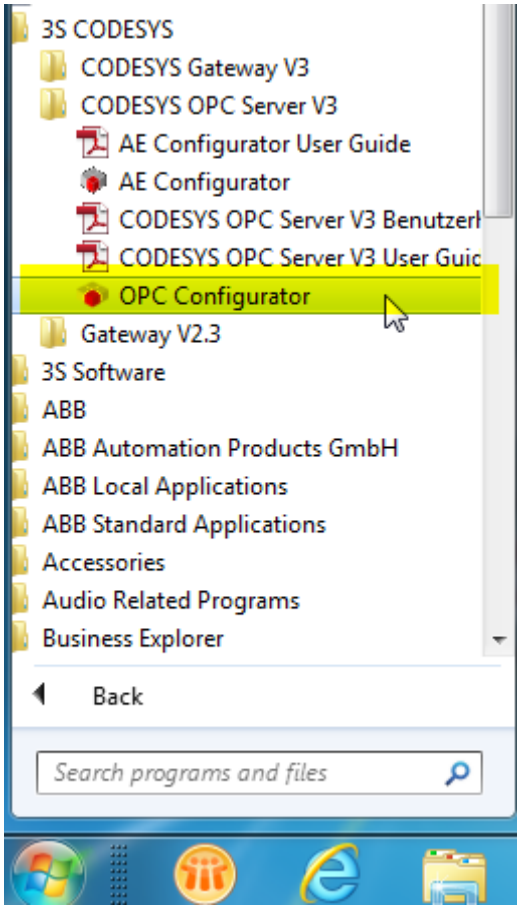


	Note: Prerequisite for the OPC Server configuration is that a symbol file is configured and downloaded to the AC500 V2 PLC as described in chapter AC500 V2 PLC7.1.1.

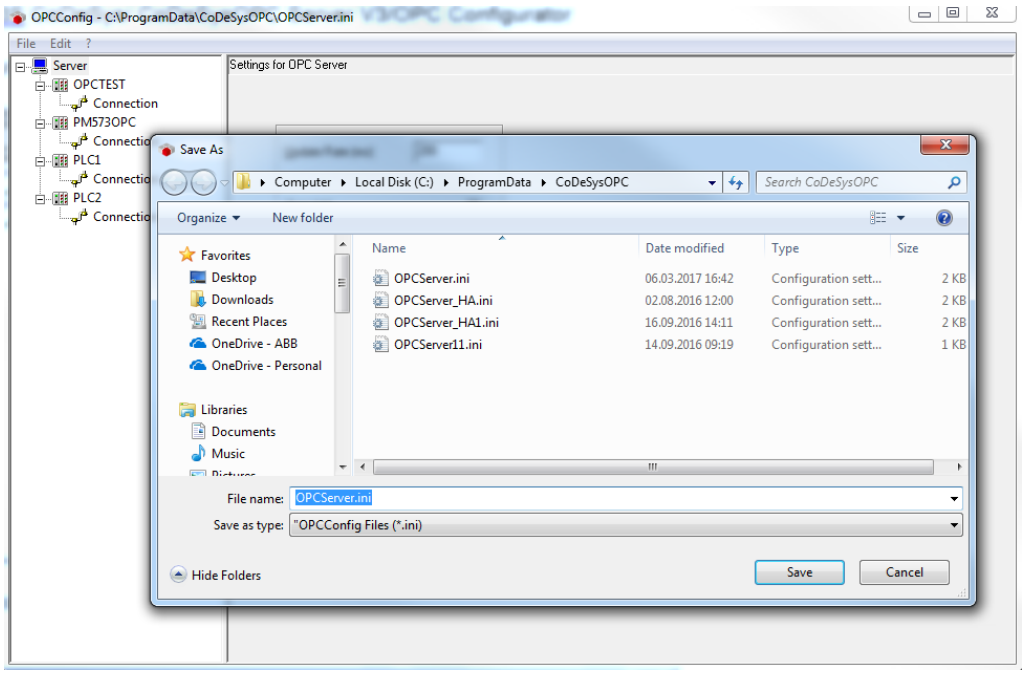
7.3.1.1 Configure OPC Server

Since OPC Server AC500 V2 and AC500 V3 can be installed in parallel on the same PC, we will use OPC server V3 in this example which can support OPC connection to AC500 V2 and AC500 V3 PLCs.

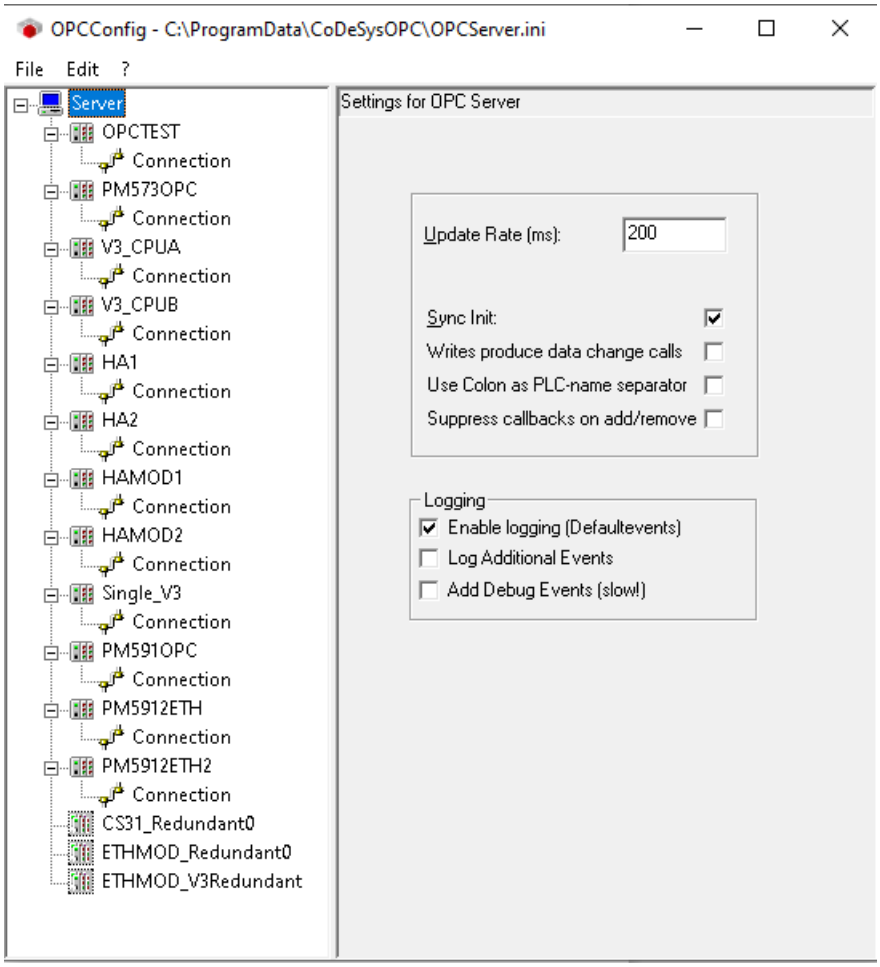
Start via 3S CODESYS/CoDeSysOPC Server V3/OPC Configurator, or start the OPC configurator by fold-er C:\Program Files (x86)\3S CODESYS\CODESYS OPC Server 3\ OPCConfig.exe directly.



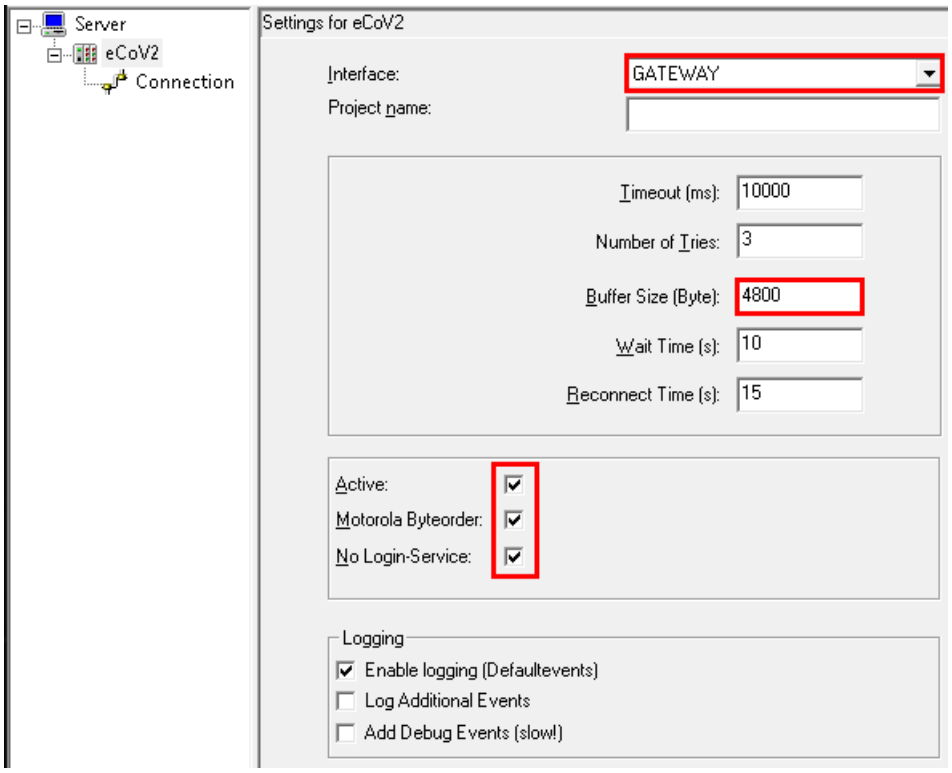
When "OPCConfig.exe" is called, the current configuration of the "OPCServer.ini" is displayed.



If the configuration is required for other projects, save it under a new name.

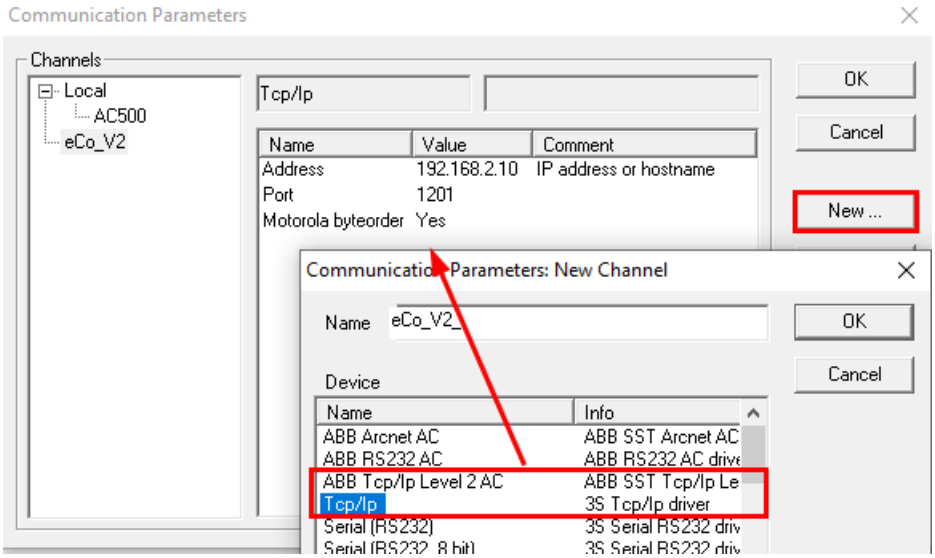


Append a new PLC under *Server* tree, and make the settings as shown below:



The project name is not required because the symbol information should be uploaded from AC500-eCo V2.

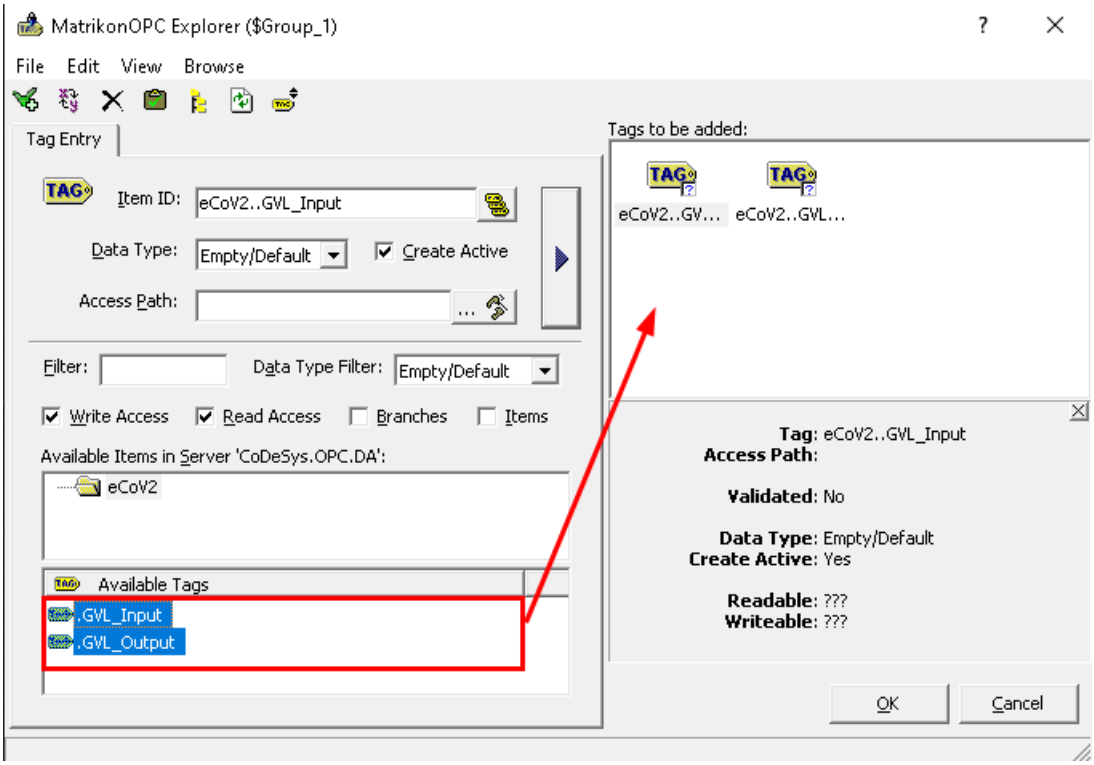
Open the connection settings and choose a channel of the channel list (normally the channel which is used for programming), or click button 'New…' to create a new channel.



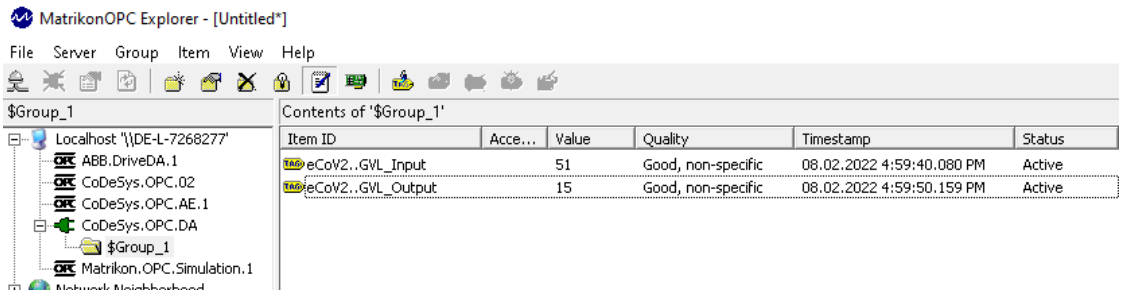
If more than one PLC, then repeat for the other PLCs. After that save as 'OPCServer.ini' and close the OPCConfig tool.

7.3.1.2 Check OPC Server with MatrikonOPC Explorer.

- Start MatrikonOPC Explorer
- Connect CoDesys.OPC.DA
- Add Group
- Add Items → Select available Items in 'Server CoDeSys.OPC.DA'
- Add to Tag List and close the Item browser

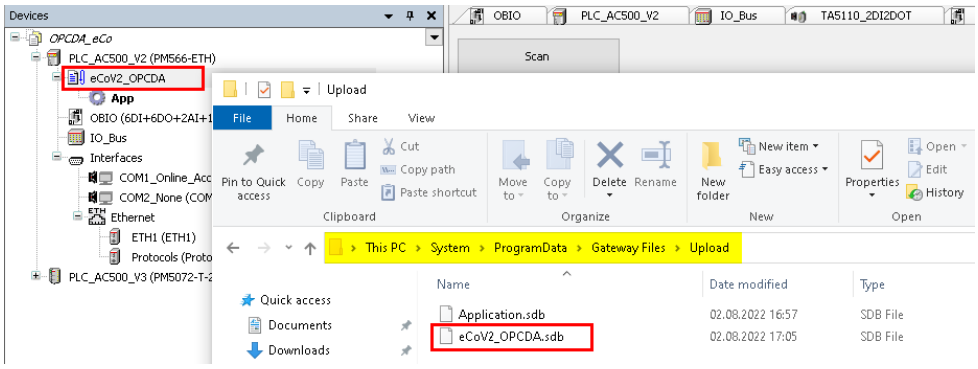


Now the variables will be shown and automatically updated.



If the configuration is correct, then CoDeSys.OPC.DA is connected. The OPC Client is running and the quality of the items is good. The values of the items can be read and written by an OPC client.

When OPC server is started, the xxx.sdb file will be copied from PLC to PC path ‘C:\ProgramData\Gateway Files\Upload’ for gateway communication.



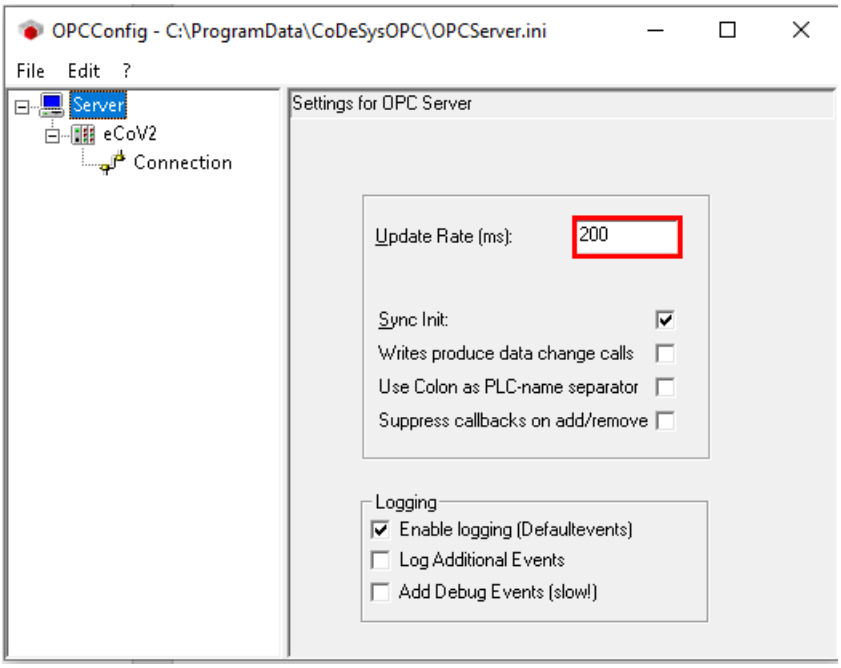
The xxx.sdb is a binary file and is needed by OPC server indeed.

7.3.2 OPC DA for AC500 V3

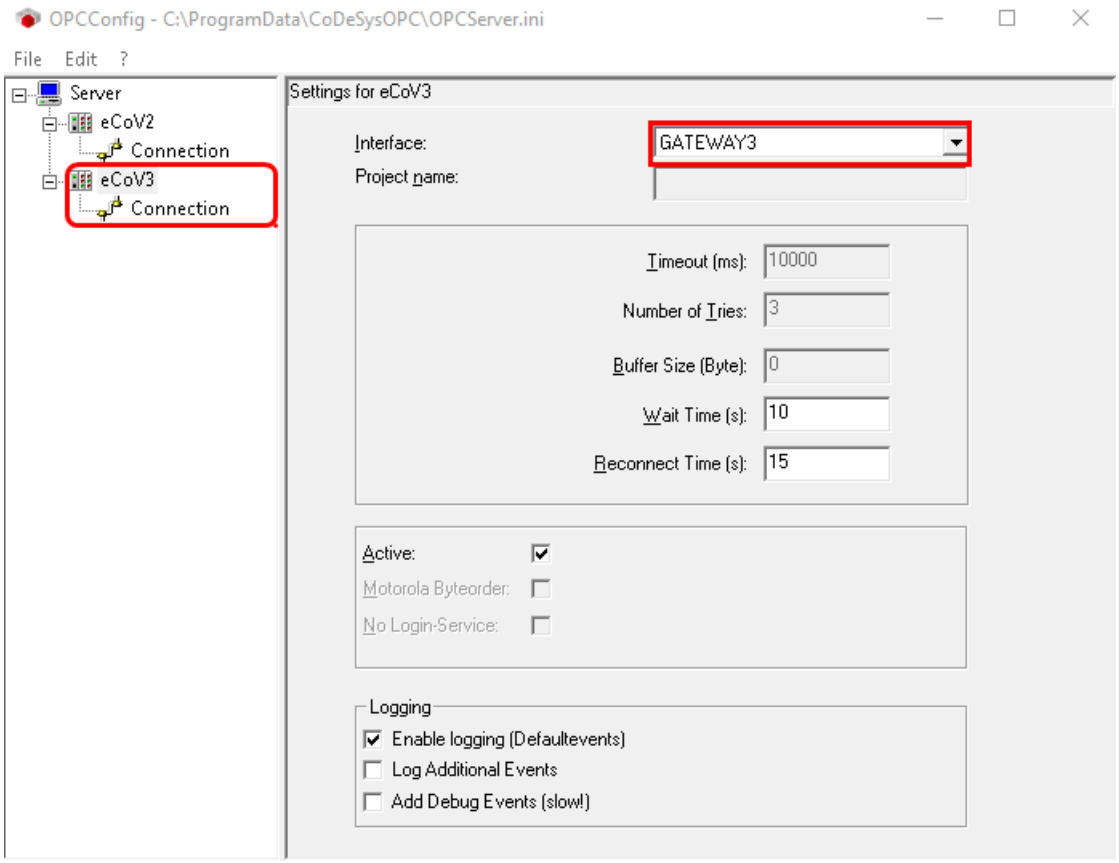
Note: Prerequisite for the OPC Server configuration is that a symbol file is configured and downloaded to the AC500 V3 PLC as described in chapter 7.1.2.

7.3.2.1 Configure OPC Server

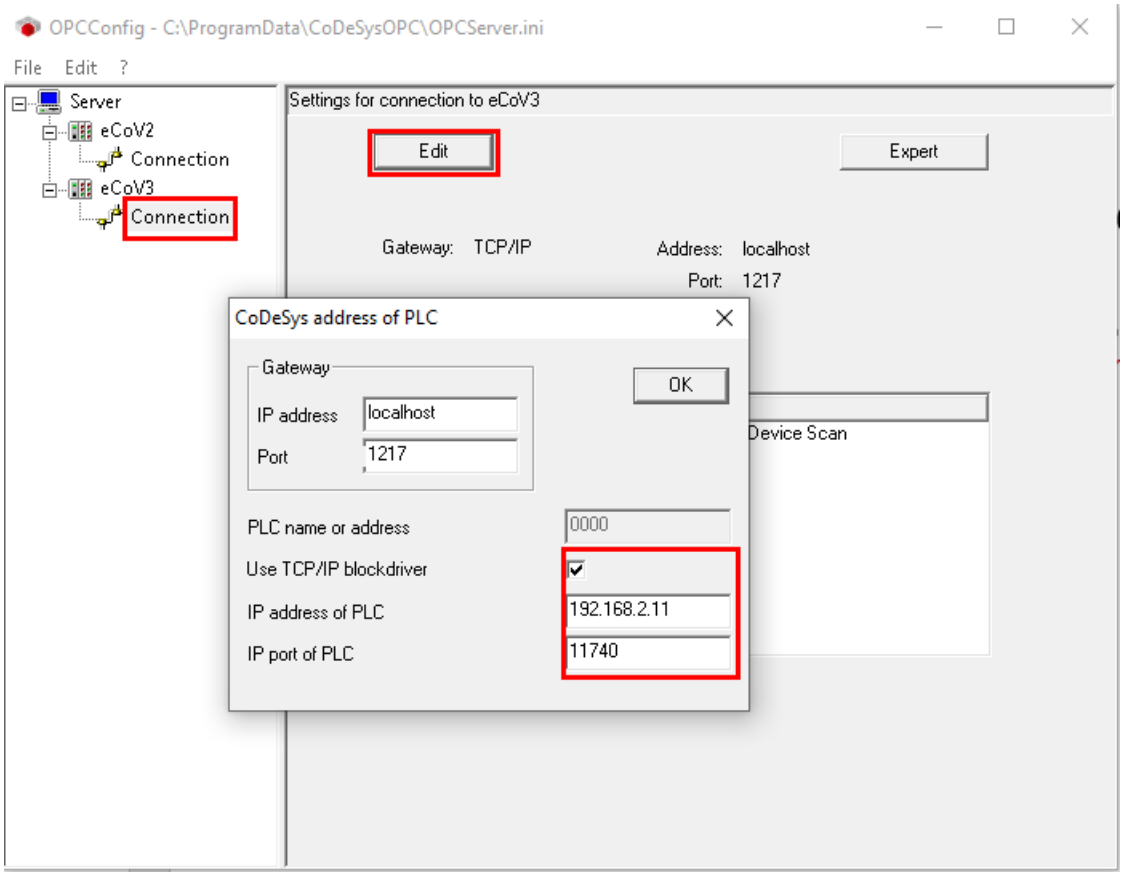
Start via 3S CODESYS/CoDeSysOPC Server V3/OPC Configurator, or start the OPC configurator by fold-er C:\Program Files (x86)\3S CODESYS\CODESYS OPC Server 3\ OPCConfig.exe directly.



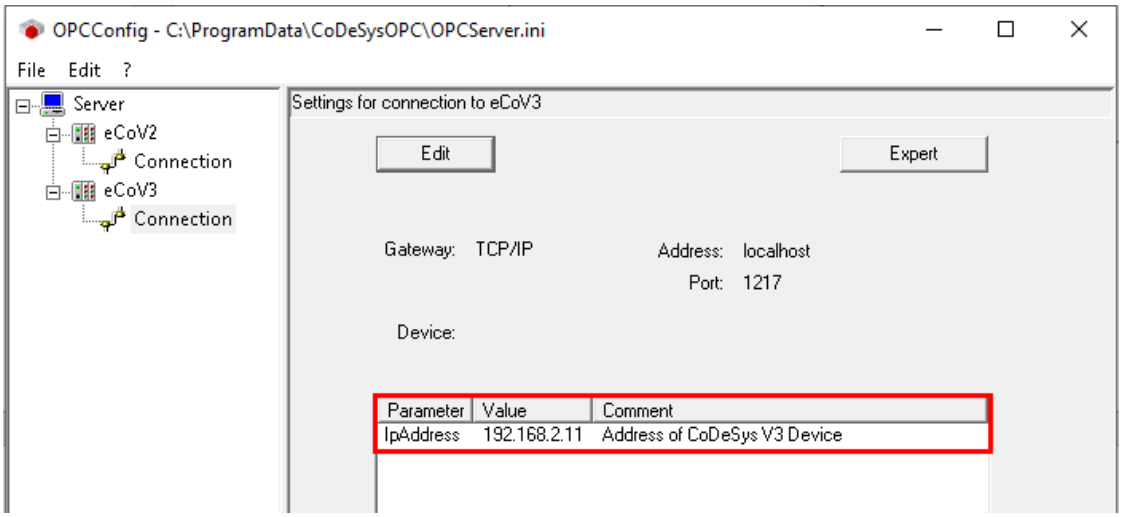
Append a new PLC under Server and make the settings as shown below:



Open the Codesys address of PLC dialog by pushing Edit button, then enter the PLC IP address and Port.

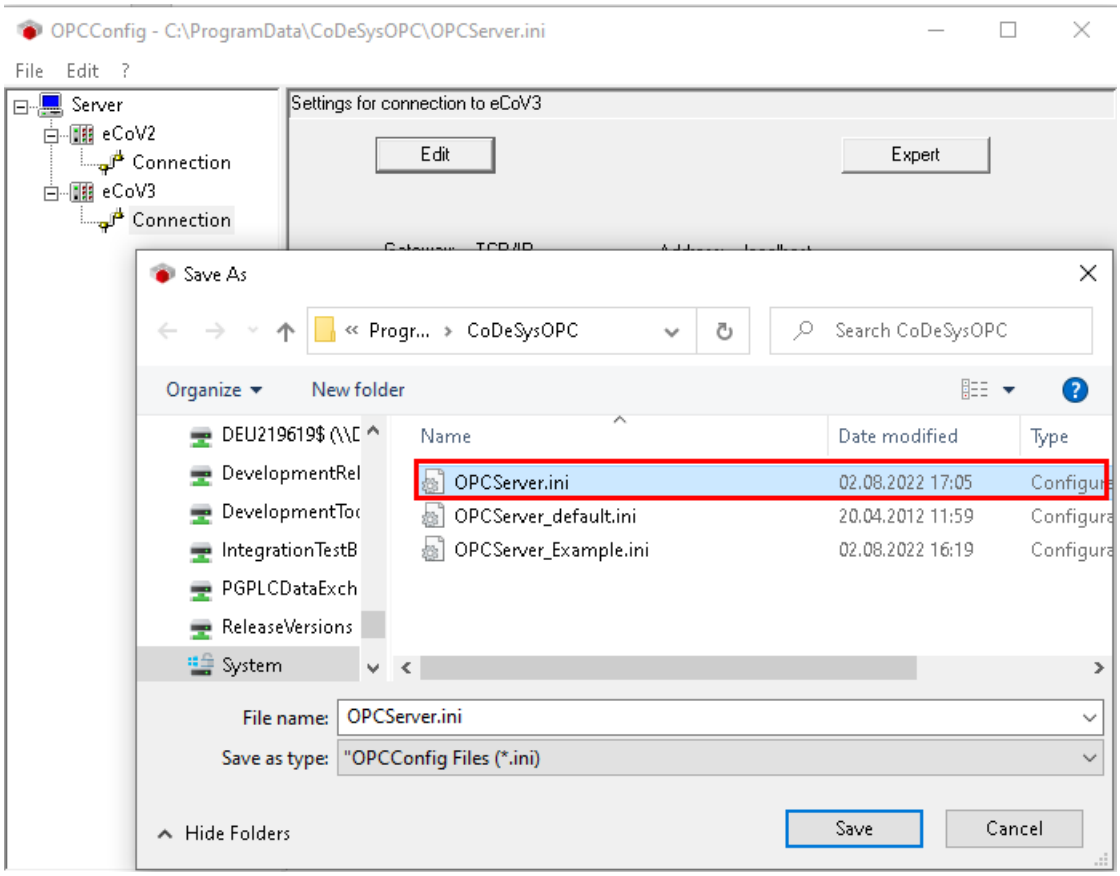


After closing the dialog with OK the settings will be displayed in the main dialog.



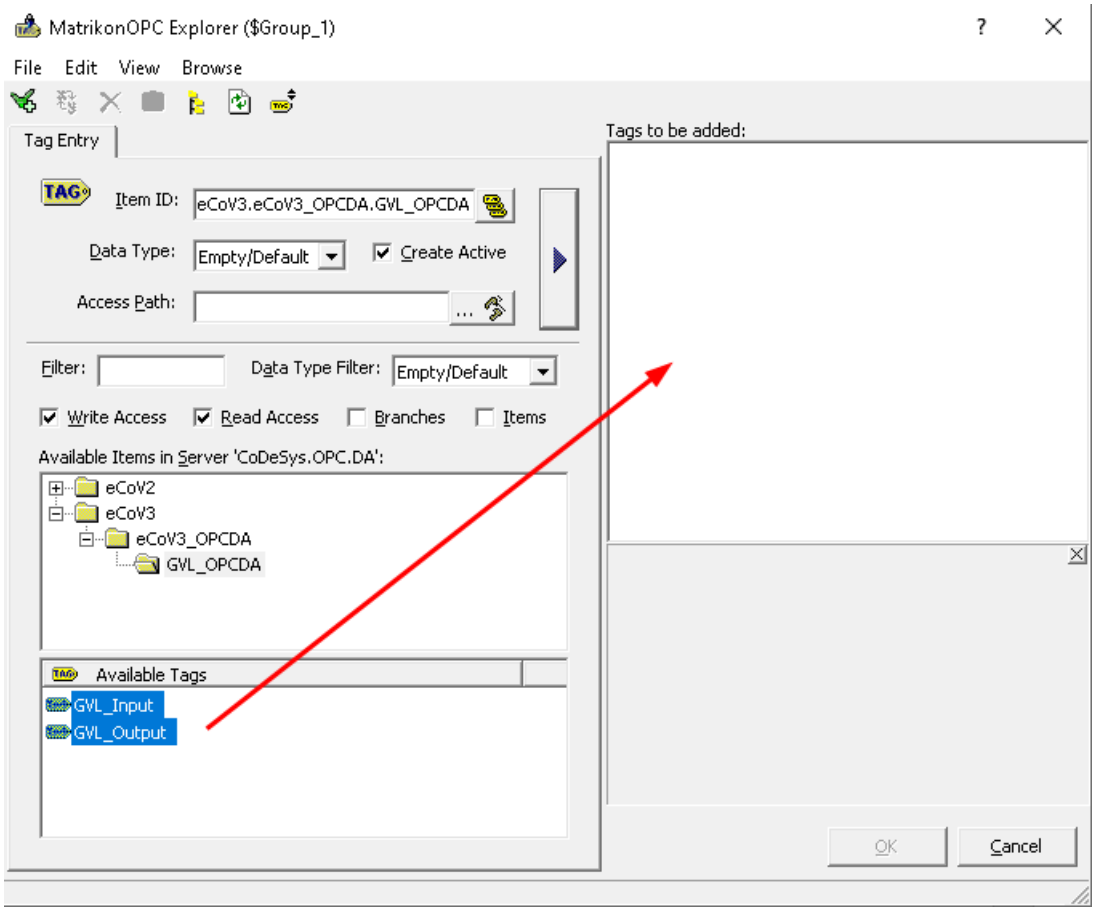
If more than one PLC, then repeat for the other PLCs.

Save as by default “OPCServer.ini” in the installation directory and close the OPCConfig tool..

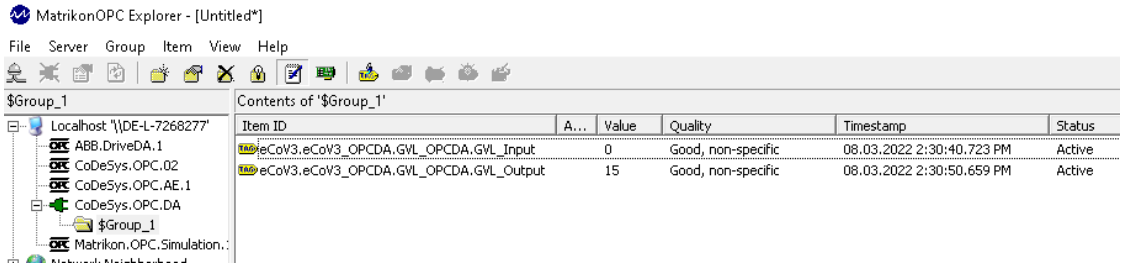


7.3.2.2 Check OPC Server with MatrikonOPC Explorer.

- Start MatrikonOPC Explorer
- Connect CoDesys.OPC.DA
- Add Group
- Add Items → Select available Items in 'Server CoDeSys.OPC.DA'
- Add to Tag List and Close the Item browser



Now the variables will be shown and automatically updated.



If the configuration is correct, then CoDeSys.OPC.DA is connected. The OPC Client is running and the quality of the items is good. The values of the items can be read and written by an OPC client.

7.3.3 Update the project from AC500 V2 to AC500 V3

If the user has an old project of AC500 V2, and now wants to upgrade to an AC500 V3 PLC, its OPCDA communication settings must be reconfigured. For details, please refer to the above sub chapter 7.3.2. From the user's point of view, its configuration operation will become simpler compared to the AC500 V2 OPC DA.

7.4 UDP

For AC500 V3 PLC, UDP is no longer configurable in the device tree. A UDP communication can be established with the library **Net Base Services** from Codesys.

An application example is currently in preparation. If a pre version is required, please contact plc.support@de.abb.com.

7.5 MQTT & JSON

An MQTT application using also the JSON library can be converted to V3 quite simple. As both MQTT and JSON library in V2 are already in the PLC open style no adaptions for the function block calls are necessary. If constants are used, they need to be accessed via the namespace in V3.

E.g. MQTT_MAX_TOPIC_LEN needs to be accessed via the namespace AC500_MQTT.Constants.MQTT_MAX_TOPIC_LEN

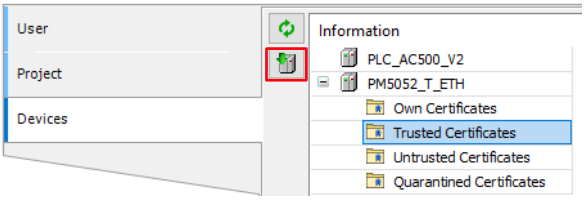
```
ReceivedTopic        :          STRING (AC500_MQTT.Constants.MQTT_MAX_TOPIC_LEN);
```

For saving the error numbers it is recommended to not use WORD variables but use variable type AC500_MQTT.ERROR_ID.

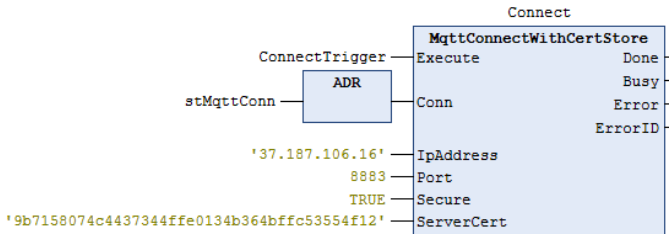
```
ConnectLastErrorID  :          AC500_MQTT.ERROR_ID;
```

The AC500 V3 PLCs offer the possibility to use a certification store.

In AC500 V2 client and server certificates could only be stored as files in the filesystem or used as string buffer. These possibilities are also available for AC500 V3 but the recommendation is to use the inbuild Certificate Manager accessible via the security screen. There the CA certificate can be downloaded to the Trusted Certificates.



For more information regarding Security please check chapter 2.6. The MQTT library in AC500 V3 offers the possibility to use the function block MqttConnectWithCertStore for using a certificate from the cert store. The Thumbprint of the certificate needs to be used as input for this function block.



Similar like described in the steps above also the code for the JSON library can be updated.

- JSON_MAX_KEY_LEN must be replace by AC500_JSON.Constants.JSON_MAX_KEY_LEN

- JSON_ERROR_ID must be replaced by AC500_JSON.ERROR_ID

After performing the steps above, you should be able to compile the MQTT/JSON part of the project again. The same functionality as in AC500 V2 is then available in AC500 V3.

ABB AG
Eppelheimer Straße 82
69123 Heidelberg, Germany
Phone: +49 62 21 701 1444
Fax: +49 62 21 701 1382
E-Mail: plc.support@de.abb.com
www.abb.com/plc