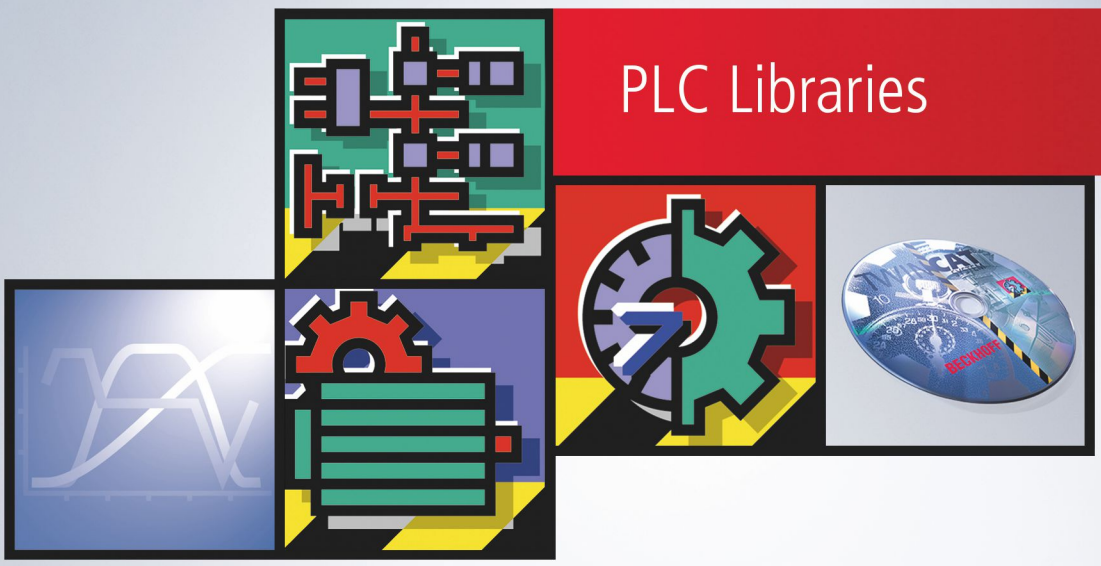
BECKHOFF New Automation Technology

Manual | EN

TX1200

TwinCAT 2 | PLC Library: TcGENIbus



2022-10-27 | Version: 1.1

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 Safety instructions 6
1.3 Notes on information security 7

2 Target groups 8

3 GENIbus 9
3.1 Device addressing 9
3.2 Wiring 9

4 Integration into TwinCAT 13
4.1 Integration in TwinCAT (CX5010) 13
4.2 Integration in TwinCAT (CX8080) 14
4.3 Integration in TwinCAT (CX9020) 16
4.4 Configuration in the TwinCAT System Manager 18
4.5 Settings when using the on-board RS485 interface 20
4.6 Linking the communication variables when using a KL6021 26
4.7 Linking the communication variables when using a KL6041 30
4.8 Configuration in the TwinCAT System Manager 35

5 Programming 45
5.1 General Information 45
5.2 Function blocks 45
5.2.1 FB_GENIbusCommunication 46
5.2.2 FB_GENIbusGet 48
5.2.3 FB_GENIbusSet 49
5.2.4 FB_GENIbusInfo 50
5.2.5 FB_GENIbusGetMValue 51
5.2.6 FB_GENIbusMagnaPump 53
5.3 Data types 56
5.3.1 E_GENIbusACK 56
5.3.2 E_GENIbusActCtrlMode 56
5.3.3 E_GENIbusActOpMode 56
5.3.4 E_GENIbusAddrType 57
5.3.5 E_GENIbusCommandPriority 57
5.3.6 E_GENIbusComMode 57
5.3.7 E_GENIbusCtrlMode 57
5.3.8 E_GENIbusKeyMode 57
5.3.9 E_GENIBusMDataSize 57
5.3.10 E_GENIbusNightReductionMode 58
5.3.11 E_GENIbusOpMode 58
5.3.12 E_GENIbusOS 58
5.3.13 E_GENIbusSD 58
5.3.14 E_GENIbusSIF 58
5.3.15 ST_GENIbusCommandBuffer 59
5.3.16 ST_GENIbusComRegisterData 59

5.3.17 ST_GENIbusEL6AMSAddress ... 59
5.3.18 ST_GENIbusEL6DeviceIn22B ... 59
5.3.19 ST_GENIbusEL6DeviceOut22B ... 60
5.3.20 ST_GENIbusInData ... 60
5.3.21 ST_GENIbusKL6DeviceIn22B ... 60
5.3.22 ST_GENIbusKL6DeviceIn5B ... 60
5.3.23 ST_GENIbusKL6DeviceOut22B ... 60
5.3.24 ST_GENIbusKL6DeviceOut5B ... 61
5.3.25 ST_GENIbusMessageQueue ... 61
5.3.26 ST_GENIbusMessageQueueItem ... 61
5.3.27 ST_GENIbusMValue ... 62
5.3.28 ST_GENIbusOutData ... 62
5.3.29 ST_GENIbusPcComDeviceIn64B ... 63
5.3.30 ST_GENIbusPcComDeviceOut64B ... 63
5.3.31 ST_GENIbusReplyClassEntry ... 63
5.3.32 ST_GENIbusReplyDataEntry ... 63
5.3.33 ST_GENIbusRequestClassEntry ... 64
5.3.34 ST_GENIbusRequestDataEntry ... 64
5.3.35 ST_GENIbusResponseTable ... 64
5.3.36 ST_GENIbusResponseTableItem ... 65
5.3.37 ST_GENIbusSerComBuffer ... 65
5.4 Error codes ... 65

6 Appendix ... 68
6.1 Support and Service ... 68

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Target groups

The user of this library requires basic knowledge of the following:

- TwinCAT PLC-Control
- TwinCAT System Manager
- PC and network knowledge
- Structure and properties of the Beckhoff Embedded PC and its Bus Terminal system
- Serial communication (RS485) and GENIbus protocol
- Relevant safety regulations for building technical equipment

This software library is intended for building automation system partners of Beckhoff Automation GmbH & Co. KG. The system partners operate in the field of building automation and are concerned with the installation, commissioning, expansion, maintenance and service of measurement, control and regulating systems for the technical equipment of buildings.

3 GENIbus

The TwinCAT PLC library **TcGENIbus.Lib** contains communication function blocks for the GENIbus master/slave communication from the TwinCAT PLC. GENIbus (Grundfos Electronic Network Intercommunications bus) is a protocol specially developed by the Grundfos company for the exchange of data with its devices. Several Grundfos devices can be connected via GENIbus to form a network and to be integrated into an automation system.

GENIbus is based on the RS485 hardware interface. Data exchange takes place at 9600 baud. In most cases a GENIbus network consists of a master and up to 200 slaves.

Further documentation

- GENIbus Protocol Specification
- Grundfos: Operating the MAGNA3 and MGE model H/I via the GENIpro interface

3.1 Device addressing

In principle, the GENIbus protocol knows only two types of addressing: individual addressing and collective (broadcast) commands. The addresses are to be assigned as follows:

- 0 - 31 : master addresses, i.e. the TwinCAT controllers
- 32 - 231 : slave addresses, e.g. pumps
- 255 : broadcast addressing to all slaves

At function-block level in the library the address range of the slaves is set to 1 – 200, i.e. 31 fewer than in the serial network. The reason for this is that the Grundfos parameterization devices also operate with an address range of 1 - 200. 31 is internally added to the slave address again for the serial communication.

3.2 Wiring

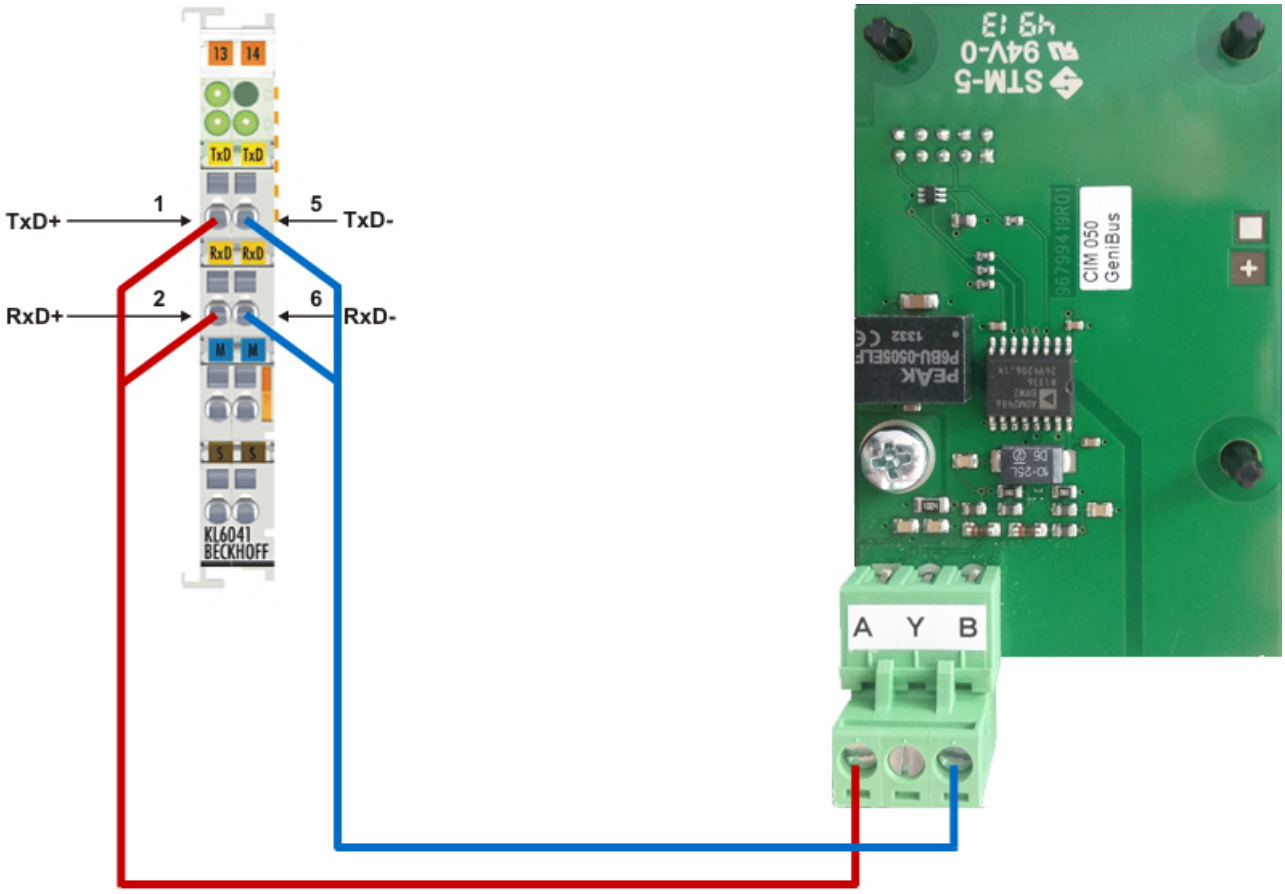
The GENIbus-communication is based on the RS485-standard running in half duplex-mode. Therefore a 2-wire-communication with a plus (A) and a minus (B) cable must be set up.
The serial-communication-terminals KL6021, KL6041 and EL6021 and the serial bus of the CX9020 can handle the half duplex-, but also and the fullduplex-mode. In this mode the plus and the minus channel exists both for the sending (Transmit-Data, TxD) and receiving-channel (Receive-Data, RxD). To use the half duplex-mode correctly, you must connect Rx- to Tx- as well as Rx+ to Tx+.
The serial bus of the CX8080 can only handle the half duplex-mode, so no direct connection between any bus-inputs is neccessary.

NOTE

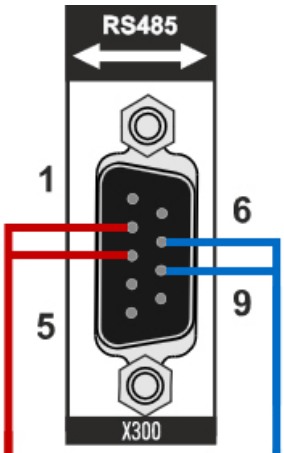
The described plus- and minus-connection to the GENIbus-module (port A for plus and port B for minus) only applies for the CIM 050, which was used for the tests. Before connecting the Beckhoff-terminals to the Grundfos-hardware, please read the documentation of the Grundfos-module, which terminals are the right ones. Furthermore, this chapter only describes the serial-bus-connections. For long wirings a shielding of the cable will be unenviable.

Wiring diagrams:

KL6021, KL6041 and EL6021

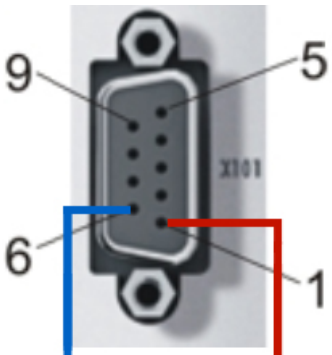


CX9020 (this configuration is not available as a program-example)



PIN	Signal
2	TxD+
3	RxD+
5	GND
6	VCC
7	TxD-
8	RxD-

CX8080



PIN	Meaning	Signal
1	**RS485**	**A (+)**
2	RxD (RS232)	Receive Data
3	TxD (RS232)	Transmit Data
4	+ 5 V	Vcc
5	GND	Ground
6	**RS485**	**B (-)**
7	RTS (RS232)	Request to Send
8	CTS (RS232)	Clear to Send
9	GND	Ground

4 Integration into TwinCAT

4.1 Integration in TwinCAT (CX5010)

This program shows the use of the individual function blocks in 5 examples https://infosys.beckhoff.com/content/1033/tcplclibgenibus/Resources/12167387787/.zip

The communication runs via an EtherCAT Terminal.

Hardware

Setting up the components

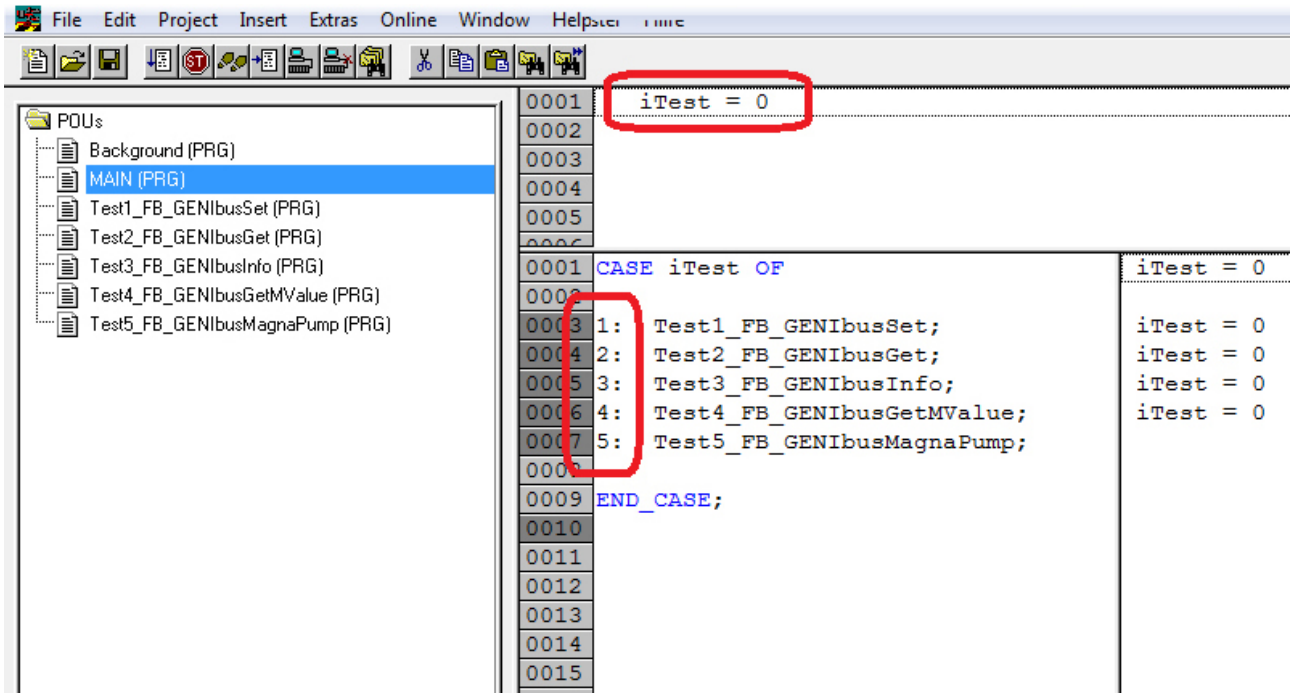
The following hardware is required:

- 1x CX5010 Embedded PC
- 1x EL1408 eight-channel digital input terminal for the execution of the individual tests
- 1x EL6021 serial RS485 terminal

Software

PLC program

The respective test program section can be selected by setting the *iTest* variable in the MAIN program to values from 1 to 5.



In the respective program sections, function blocks are then prepared that you can operate via the test inputs *ib1* to *ib8*:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;
```

```
   stInData    AT %I* : ST_GENIbusInData;
   stOutData   AT %Q* : ST_GENIbusOutData;
   stCommandBuffer    : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Pushbutton switch inputs for the tests.

stInData: Structure with the input variables for various terminal types.

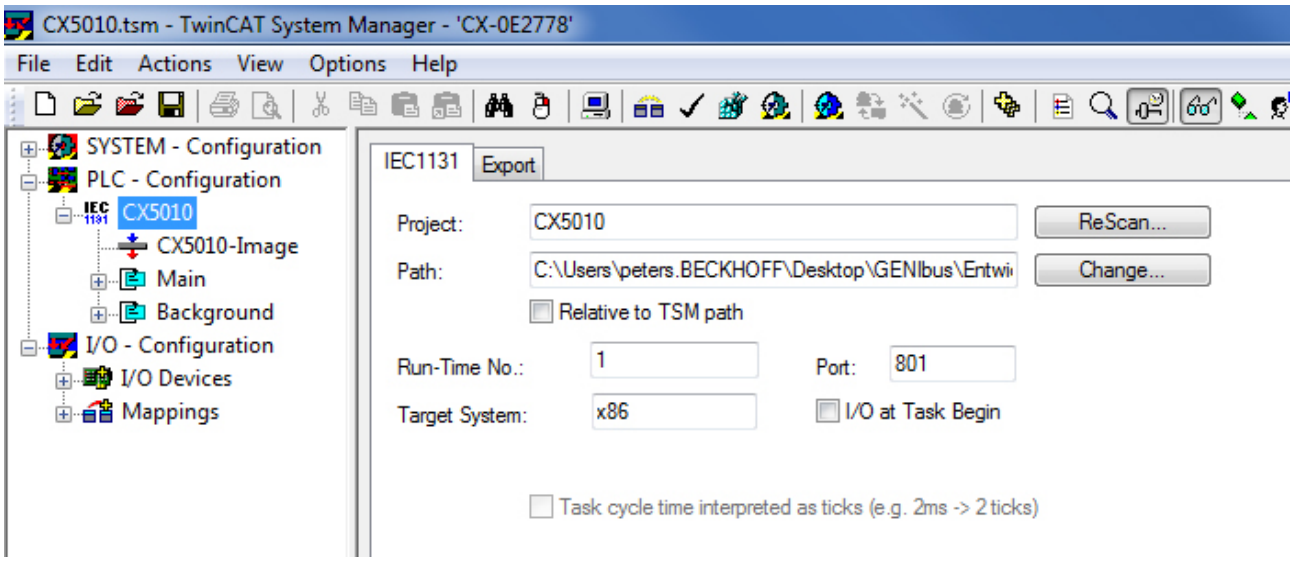
stOutData: Structure with the output variables for various terminal types.

stCommandBuffer: Reference to the structure for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

TwinCAT System Manager

In the TwinCAT System Manager the variables are already linked and assigned to the two tasks (Background: fast for communication, Main: slower for application).

Compile the PLC program and read it into the TwinCAT System Manager:



In accordance with these instructions [▶ 18], check whether the variable assignment is correct and that the variables have been linked.

After that you can start the TwinCAT System Manager and load and start the PLC program.

Also see about this

- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 59]

4.2 Integration in TwinCAT (CX8080)

This program shows the use of the individual function blocks in 5 examples https://infosys.beckhoff.com/content/1033/tcplclibgenibus/Resources/12167389195/.zip

The communication runs via the on-board PC interface of the CX.

Hardware

Setting up the components

The following hardware is required:

- 1x CX8080 Embedded PC
- 1x KL1408 eight-channel digital input terminal for the execution of the individual tests.
- 1x end terminal KL9010

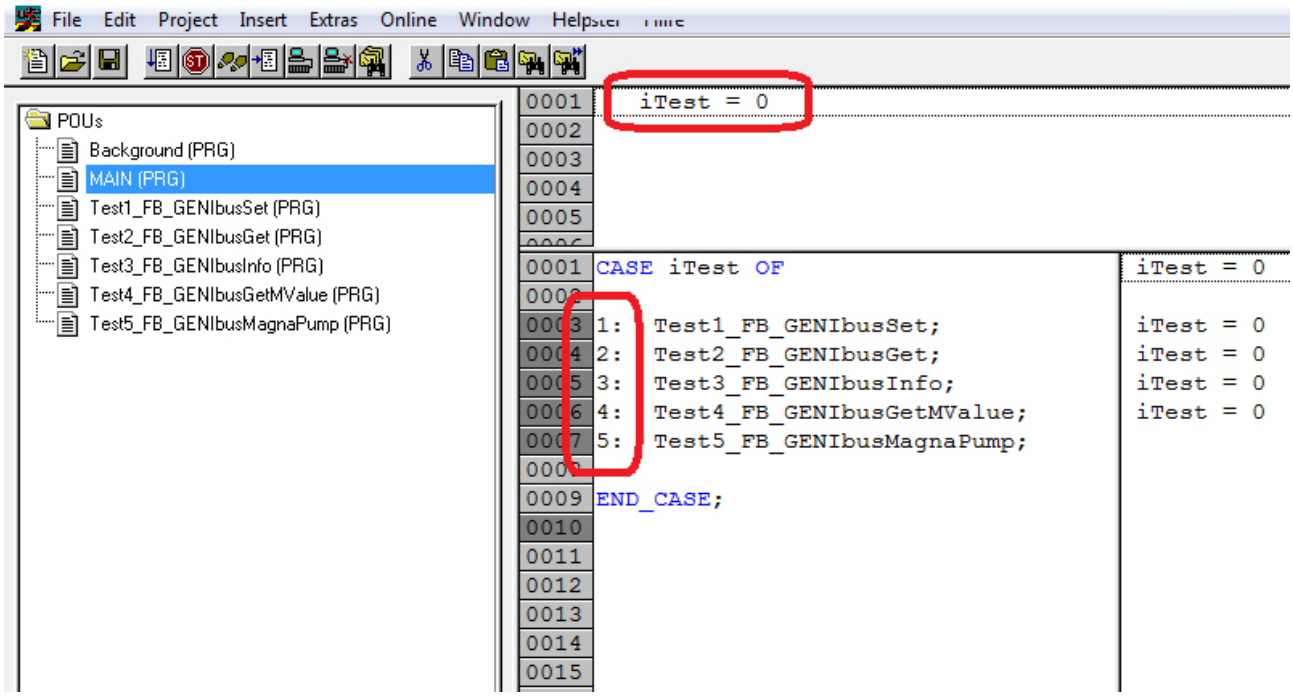
The RS485 communication pins of the Sub-D interface are:

- A (+) = Pin1
- B (-) = Pin6

Software

PLC program

The respective test program section can be selected by setting the *iTest* variable in the MAIN program to values from 1 to 5.



In the respective program sections, function blocks are then prepared that you can operate via the test inputs *ib1* to *ib8*:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;

   stInData     AT %I* : ST_GENIbusInData;
   stOutData    AT %Q* : ST_GENIbusOutData;
   stCommandBuffer     : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Pushbutton switch inputs for the tests.

stInData: Structure with the input variables for various terminal types.

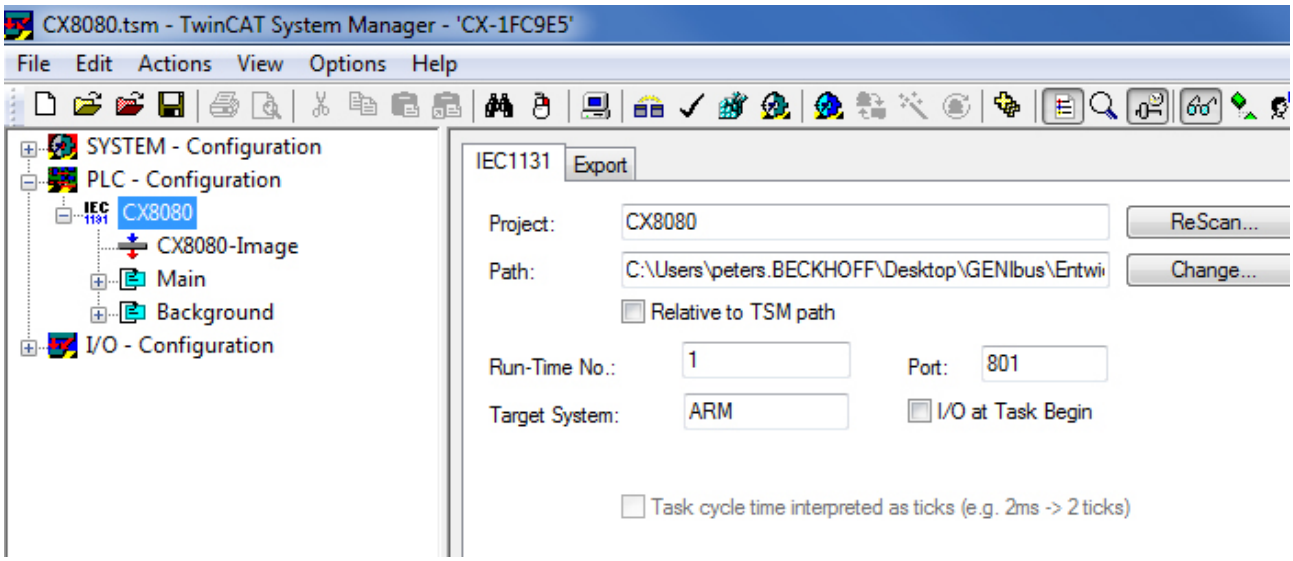
stOutData: Structure with the output variables for various terminal types.

stCommandBuffer: Reference to the structure for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

TwinCAT System Manager

In the TwinCAT System Manager the variables are already linked and assigned to the two tasks (Background: fast for communication, Main: slower for application).

Compile the PLC program and read it into the TwinCAT System Manager:



In accordance with these instructions [▶ 18], check whether the variable assignment is correct and that the variables have been linked.

After that you can start the TwinCAT System Manager and load and start the PLC program.

Also see about this

- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 59]

4.3 Integration in TwinCAT (CX9020)

This program shows the use of the individual function blocks in 5 examples https://infosys.beckhoff.com/content/1033/tcplclibgenibus/Resources/12167390603/.zip

The communication runs via a K-bus terminal.

Hardware

Setting up the components

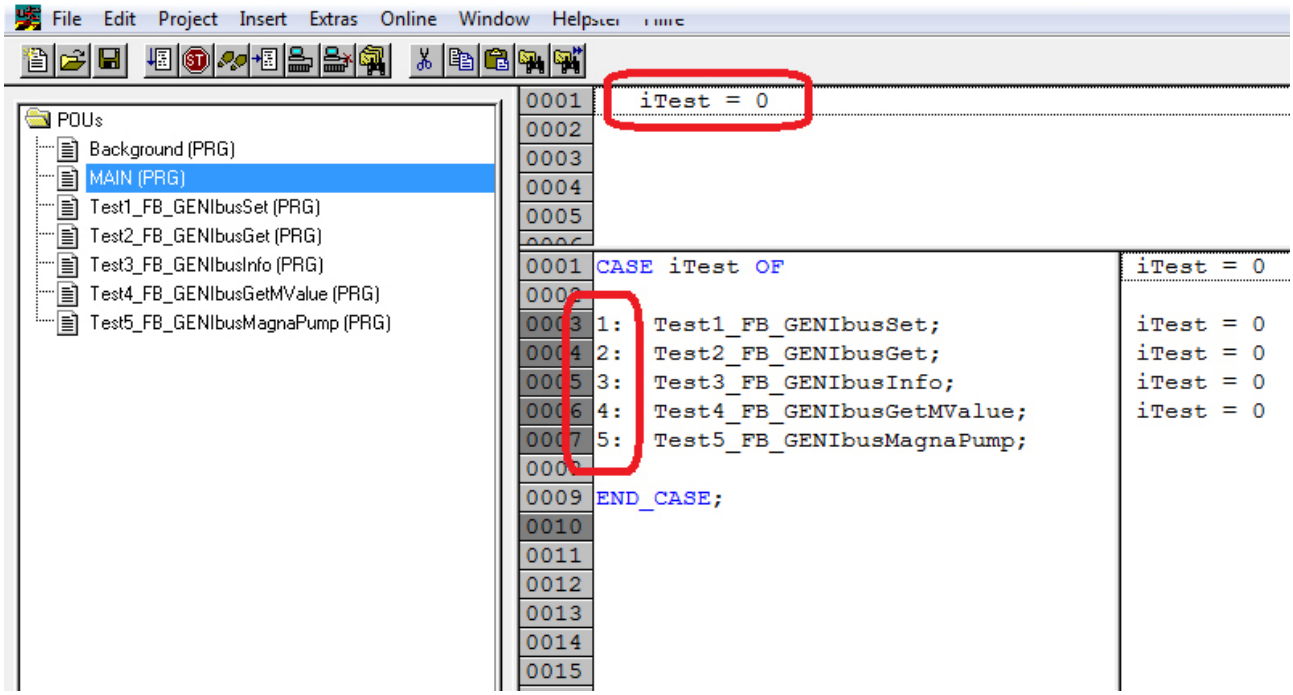
The following hardware is required:

- 1x CX9020 Embedded-PC
- 1x KL1408 eight-channel digital input terminal for the execution of the individual tests.
- 1x KL6041 serial RS485 terminal
- 1x KL9010 end terminal

Software

PLC program

The respective test program section can be selected by setting the *iTest* variable in the MAIN program to values from 1 to 5.



In the respective program sections, function blocks are then prepared that you can operate via the test inputs *ib1* to *ib8*:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;

   stInData     AT %I* : ST_GENIbusInData;
   stOutData    AT %Q* : ST_GENIbusOutData;
   stCommandBuffer     : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Pushbutton switch inputs for the tests.

stInData: Structure with the input variables for various terminal types.

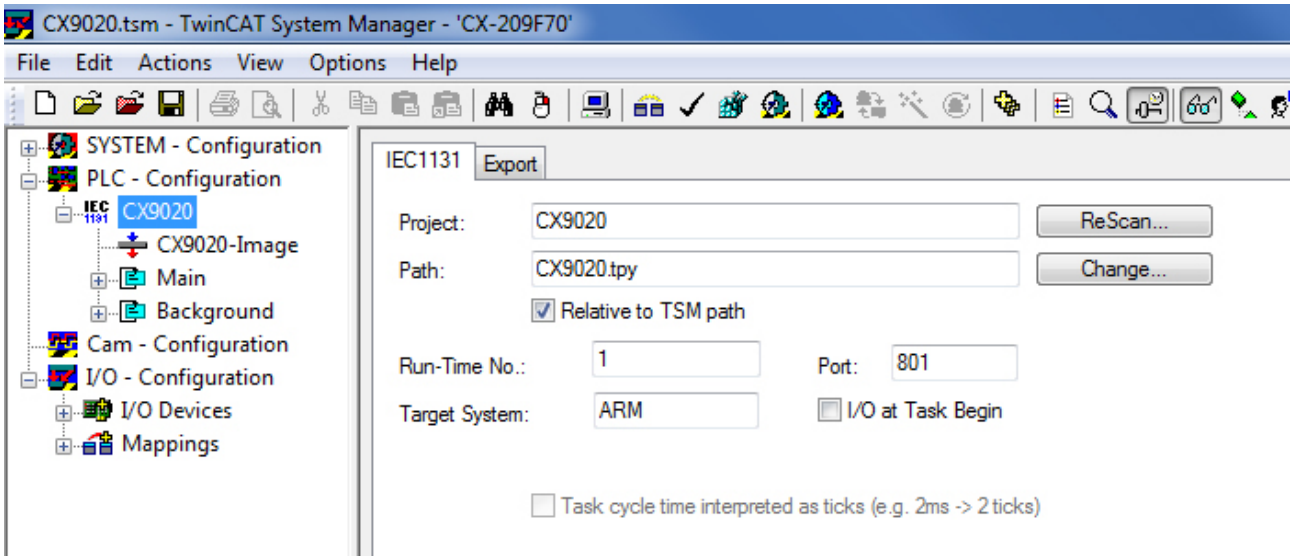
stOutData: Structure with the output variables for various terminal types.

stCommandBuffer: Reference to the structure for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

TwinCAT System Manager

In the TwinCAT System Manager the variables are already linked and assigned to the two tasks (Background: fast for communication, Main: slower for application).

Compile the PLC program and read it into the TwinCAT System Manager:



In accordance with these instructions [▶ 18], check whether the variable assignment is correct and that the variables have been linked.

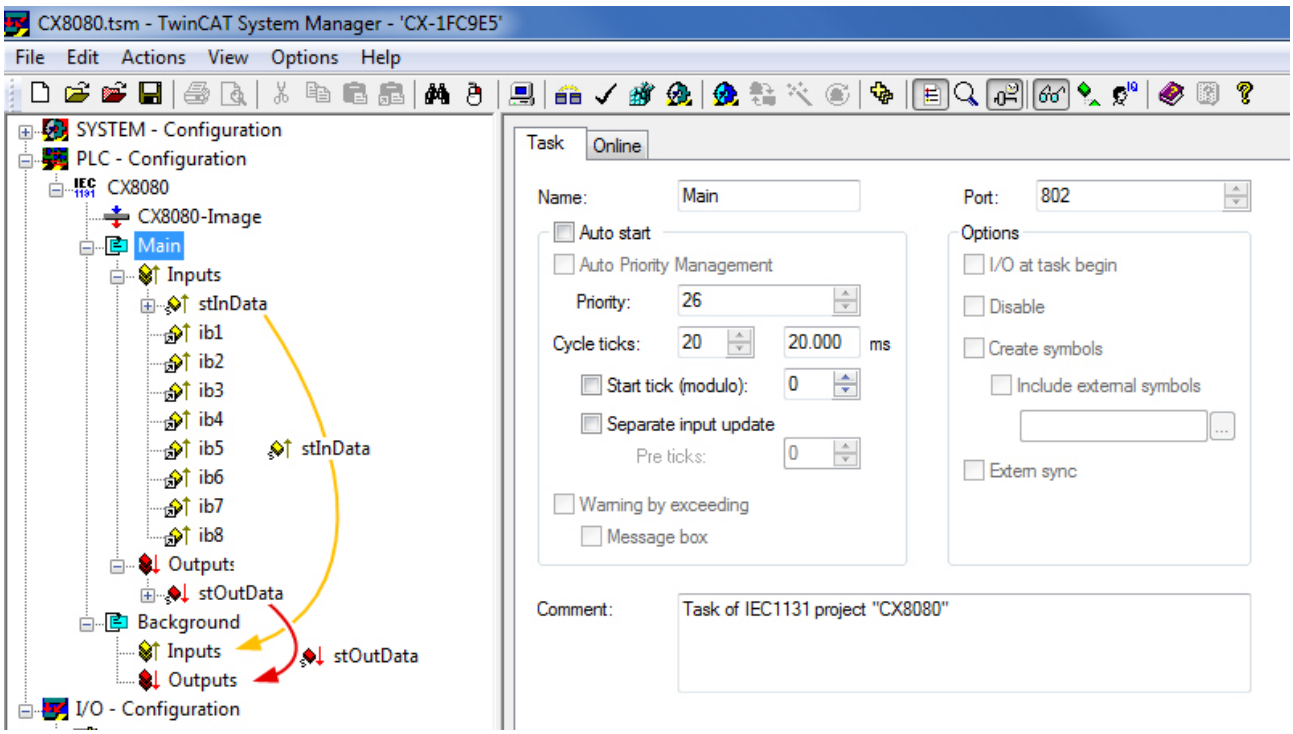
After that you can start the TwinCAT System Manager and load and start the PLC program.

Also see about this

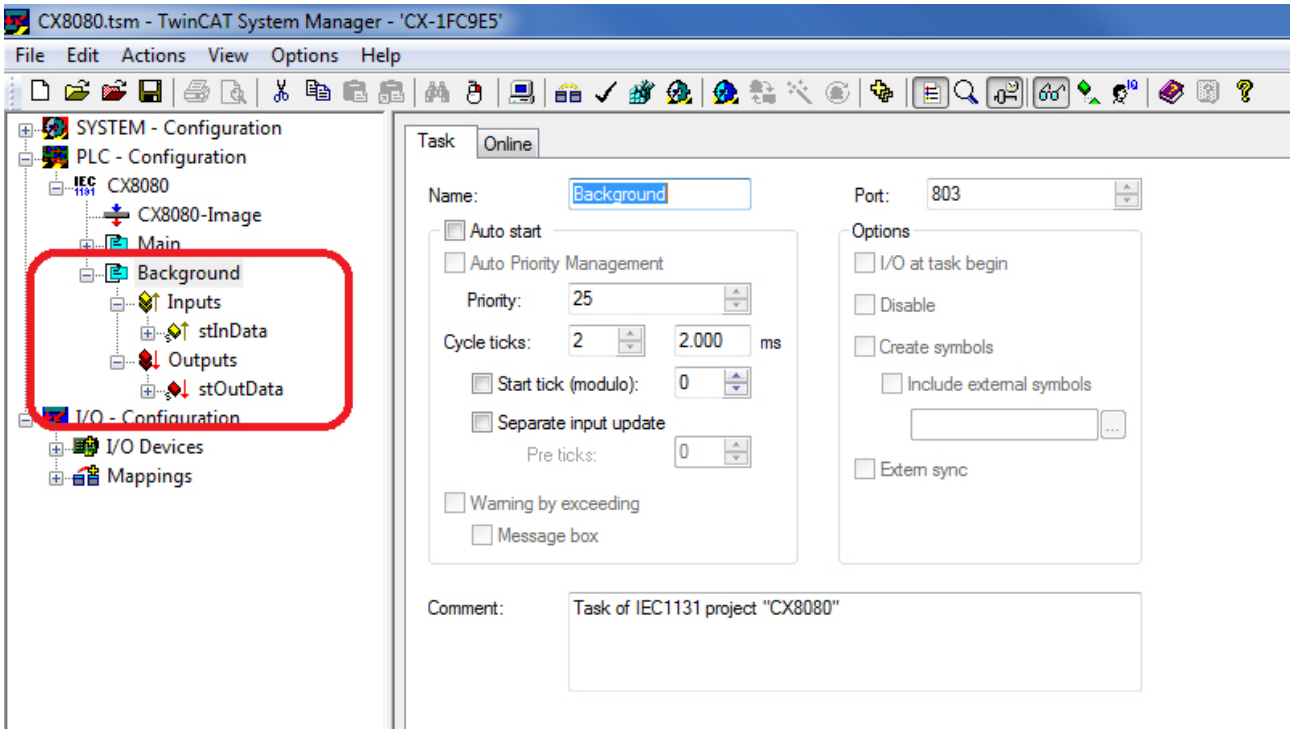
- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 59]

4.4 Configuration in the TwinCAT System Manager

If the PLC program has been compiled without error, it is important that the communication variables in the TwinCAT System Manager are assigned to the correct task. Otherwise they would not be queried or written with the desired cycle time.
After reading in the PLC program, it is usually the case that the communication variables *stInData* and *stOutData* are assigned to the slower MAIN task. These have to be pulled into the fast task – in this case “Background” – by drag & drop.



This should result in the following picture:

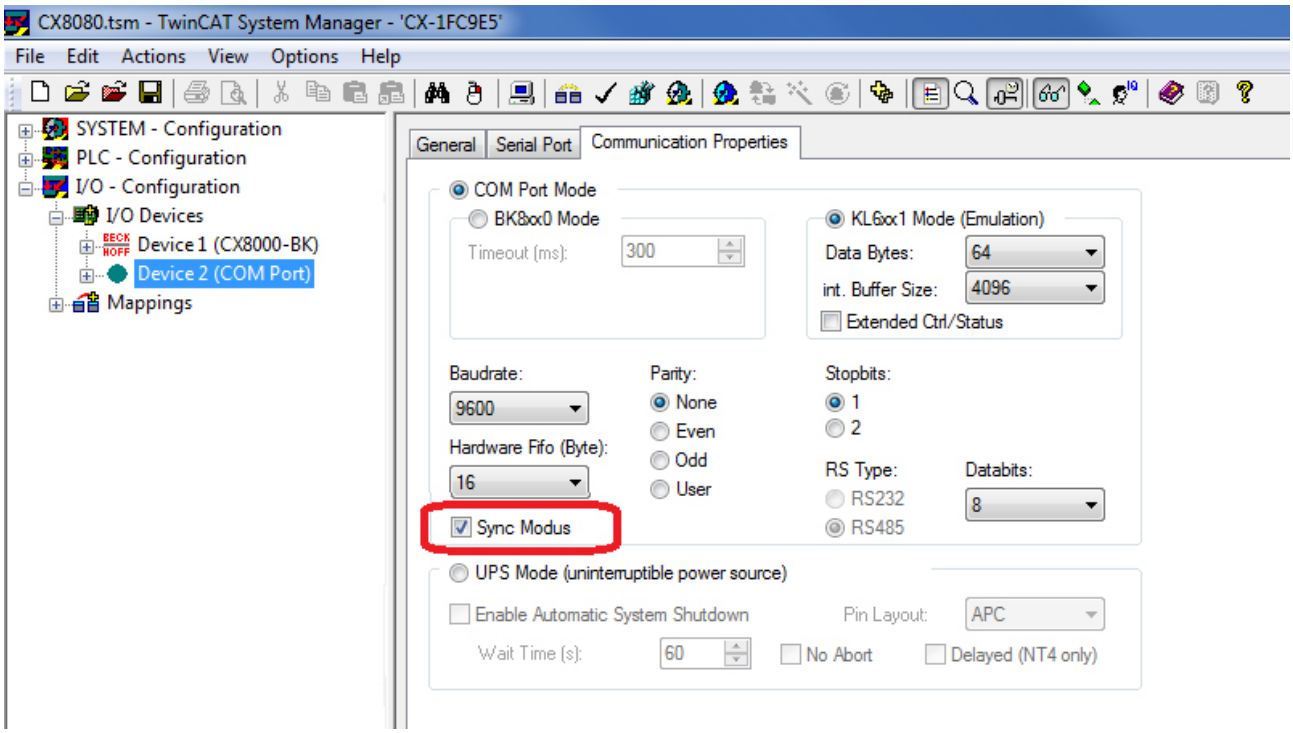


The communication variables must now be linked to the hardware in use. The following options are available for this:

- OnBoard RS485 interface (PcComm) [▶ 20]
- KL6021 (5Byte) [▶ 26]
- KL6041 (22Byte) [▶ 30]
- EL6021 (22Byte) [▶ 35]

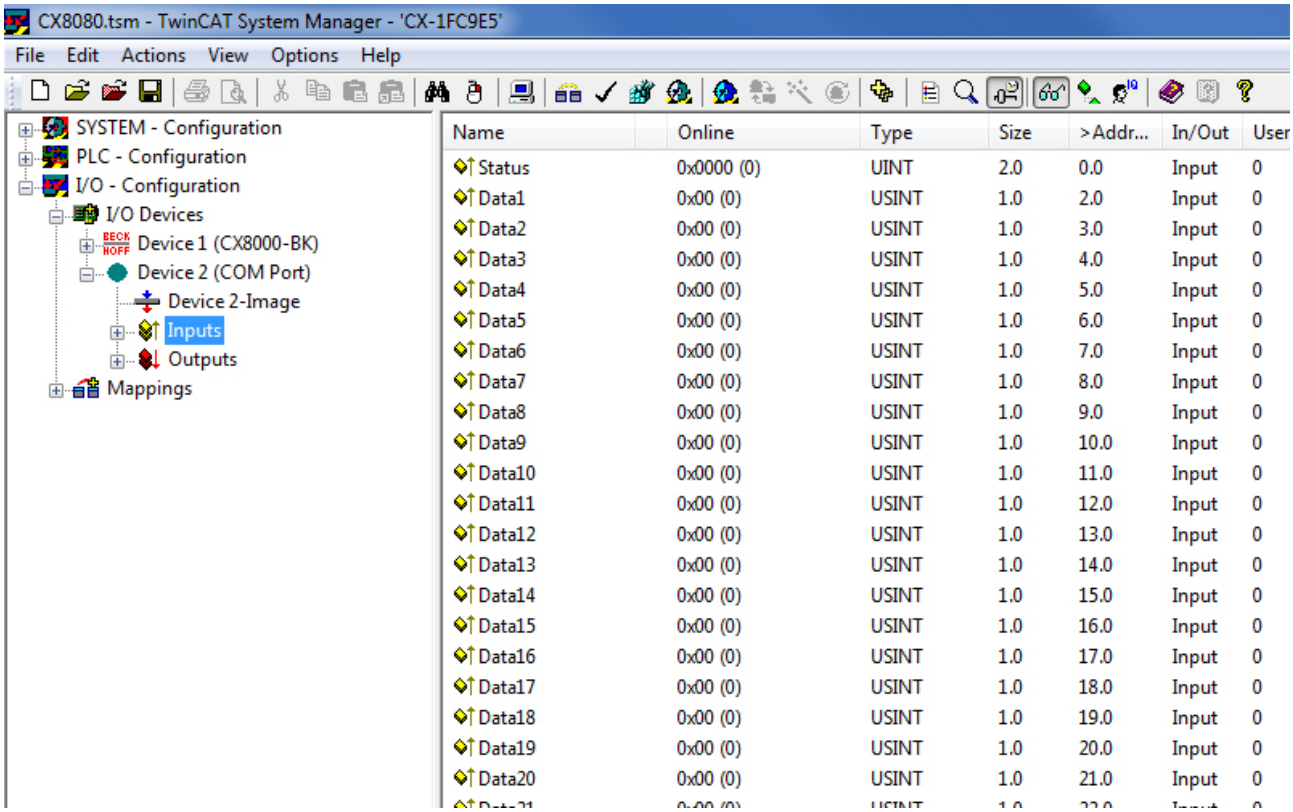
4.5 Settings when using the on-board RS485 interface

If an on-board interface is selected, it must first be correctly set in the TwinCAT System Manager, not least in order to obtain the correct process image. Unlike the terminal-based serial interfaces, the communication parameters are also set in the TwinCAT System Manager. Precisely the following settings are to be made on the “Communication Properties” tab of the COM port:

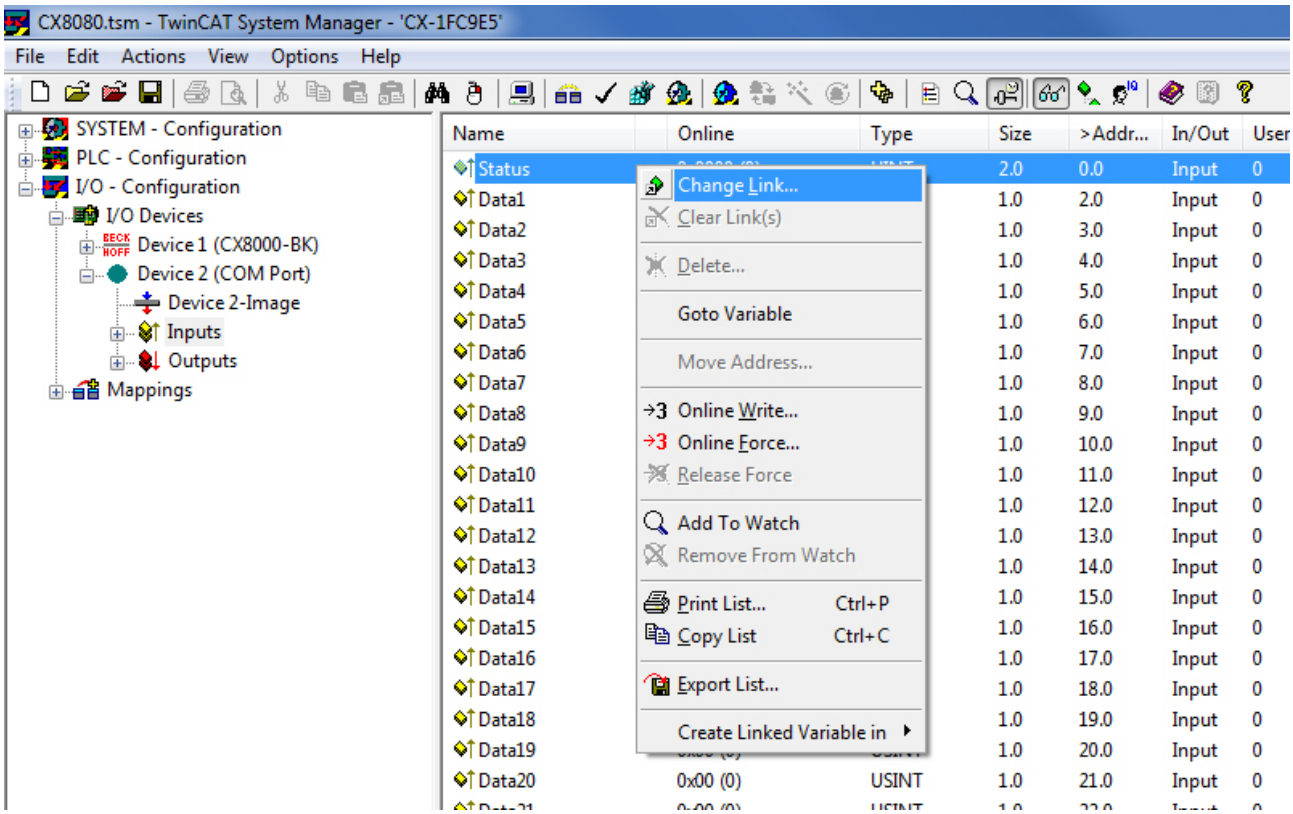


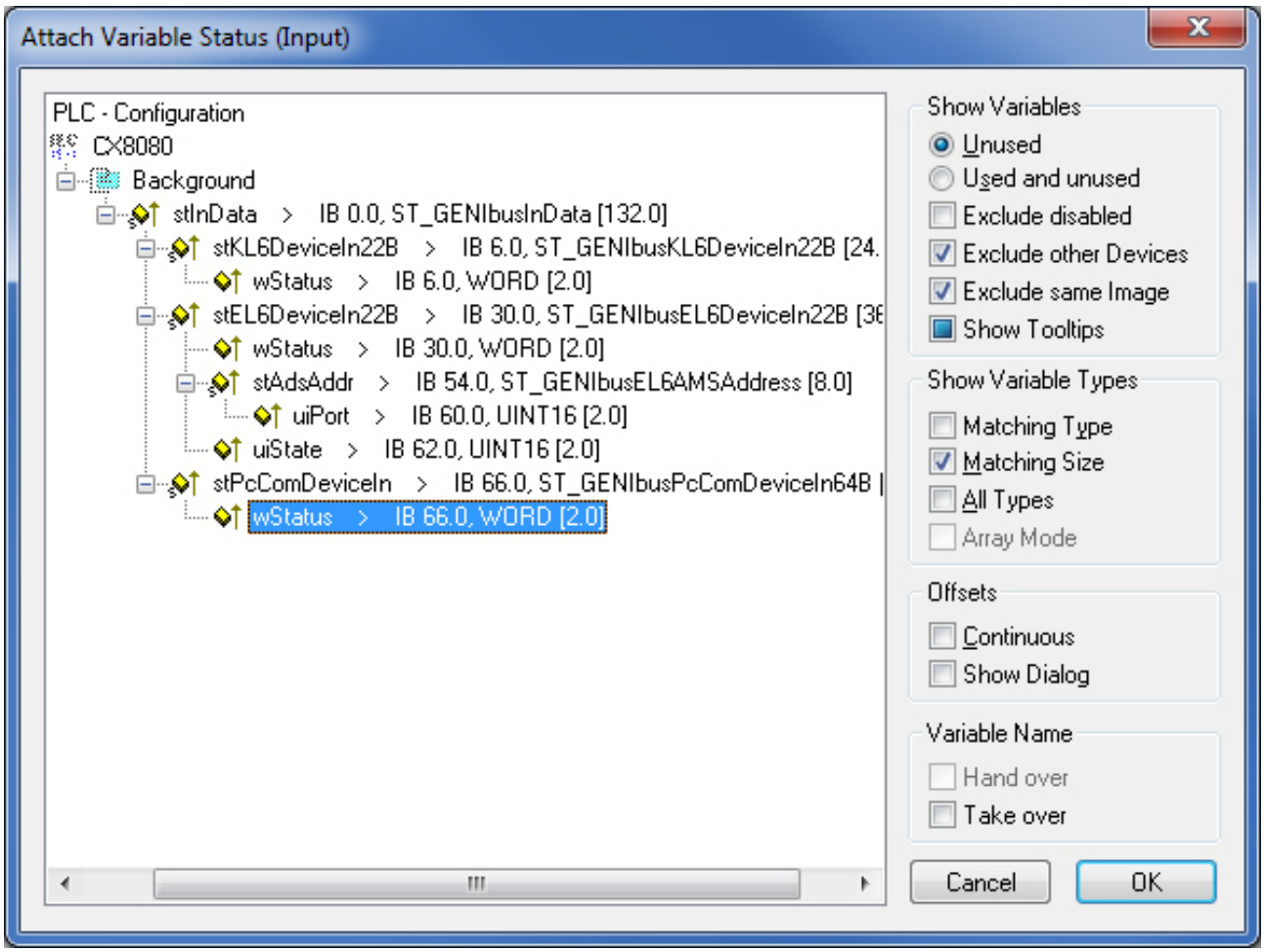
The activation of the “Sync Mode” is particularly important. This ensures that the interface, which a KL6xx1 only emulates, also works correctly under higher PLC loads.

The linking of the process image to the PLC input and output variables can be accomplished most simply from the hardware side, since multi-links are possible from there. The variables must be visible in the right-hand side of the TwinCAT System Manager for this.

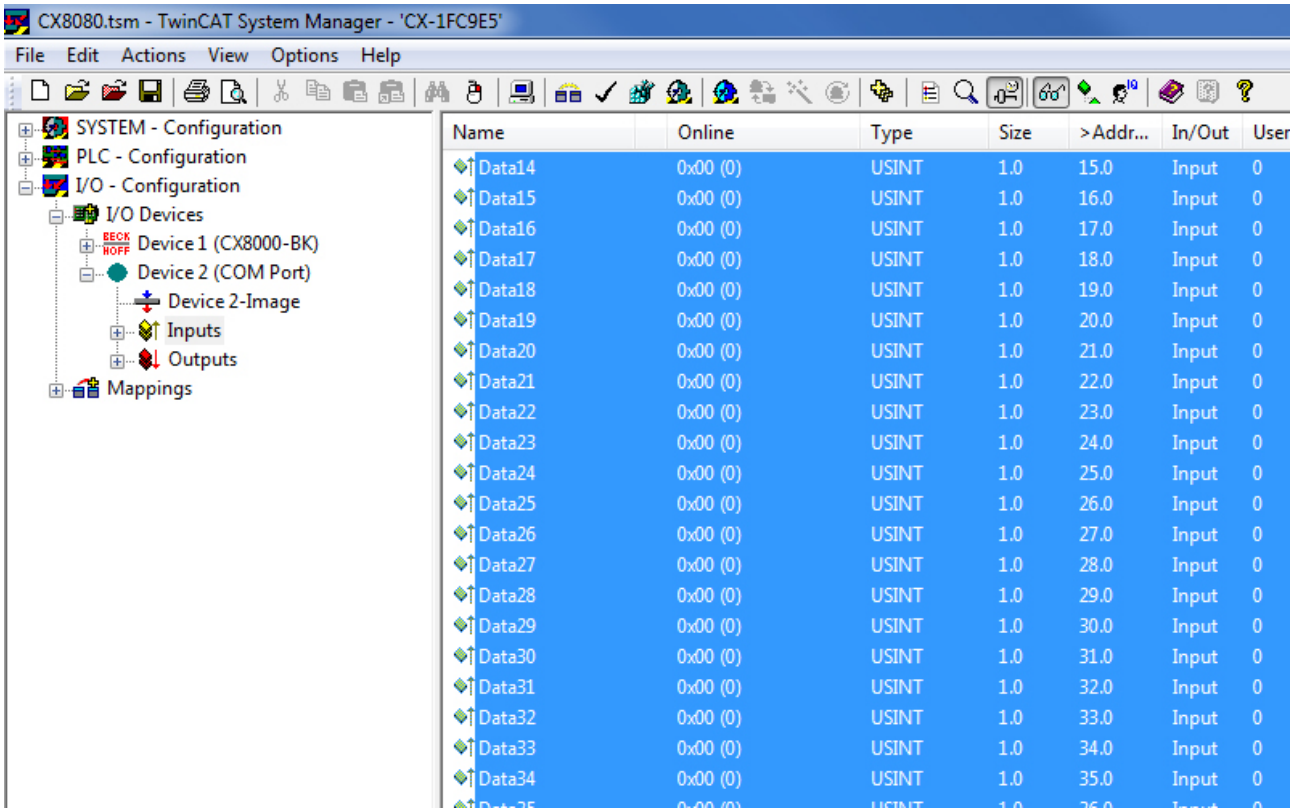


First of all, just the status is linked with the status variable of the communication input. Note that when doing so the structure for PC communication must be selected.

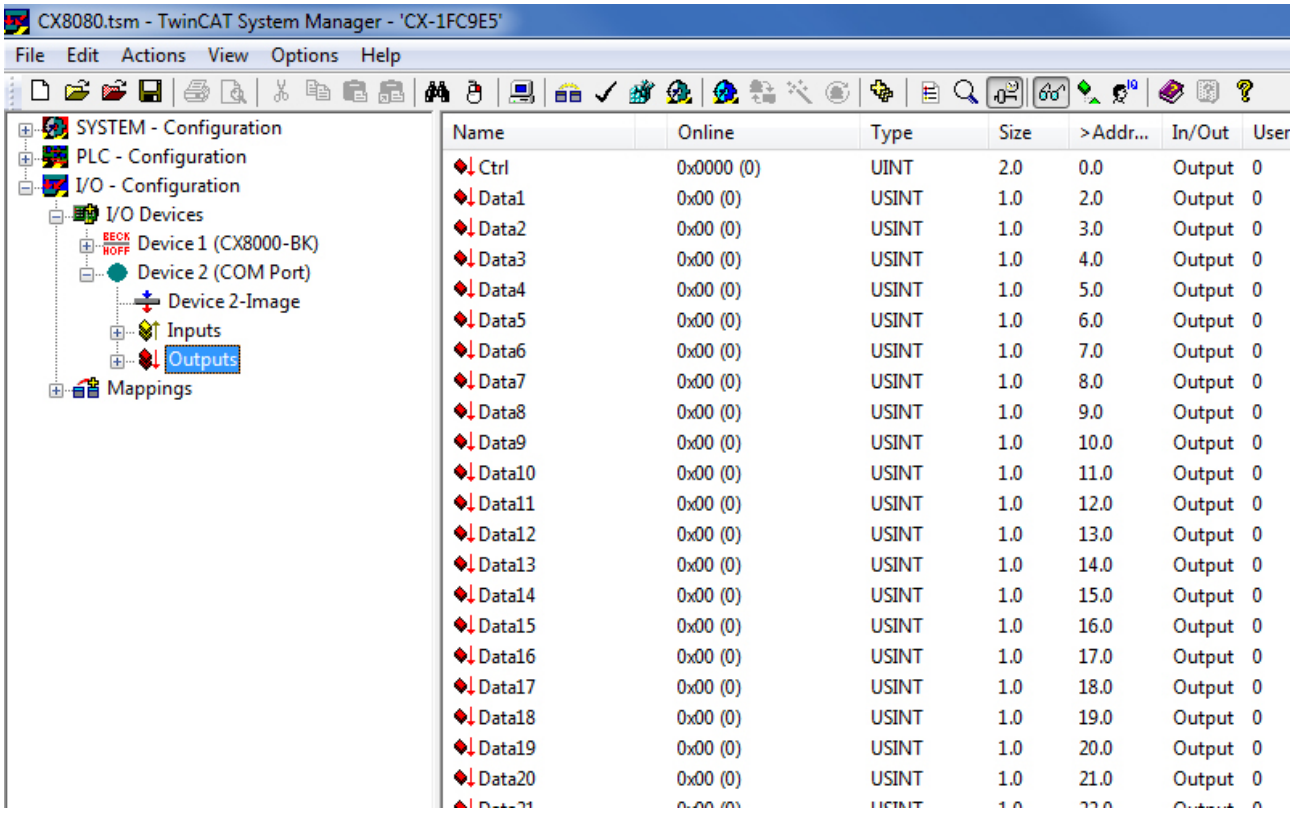


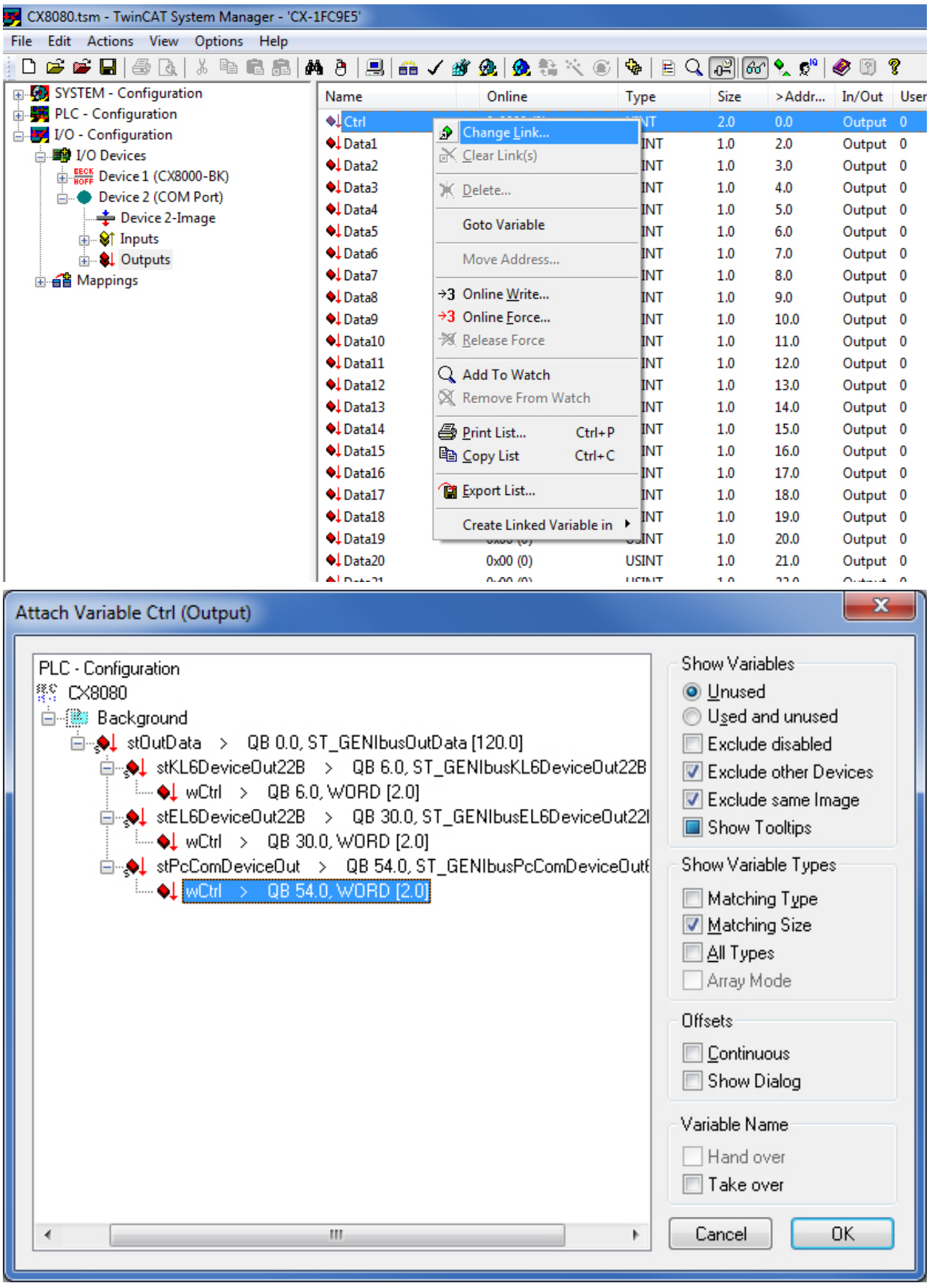


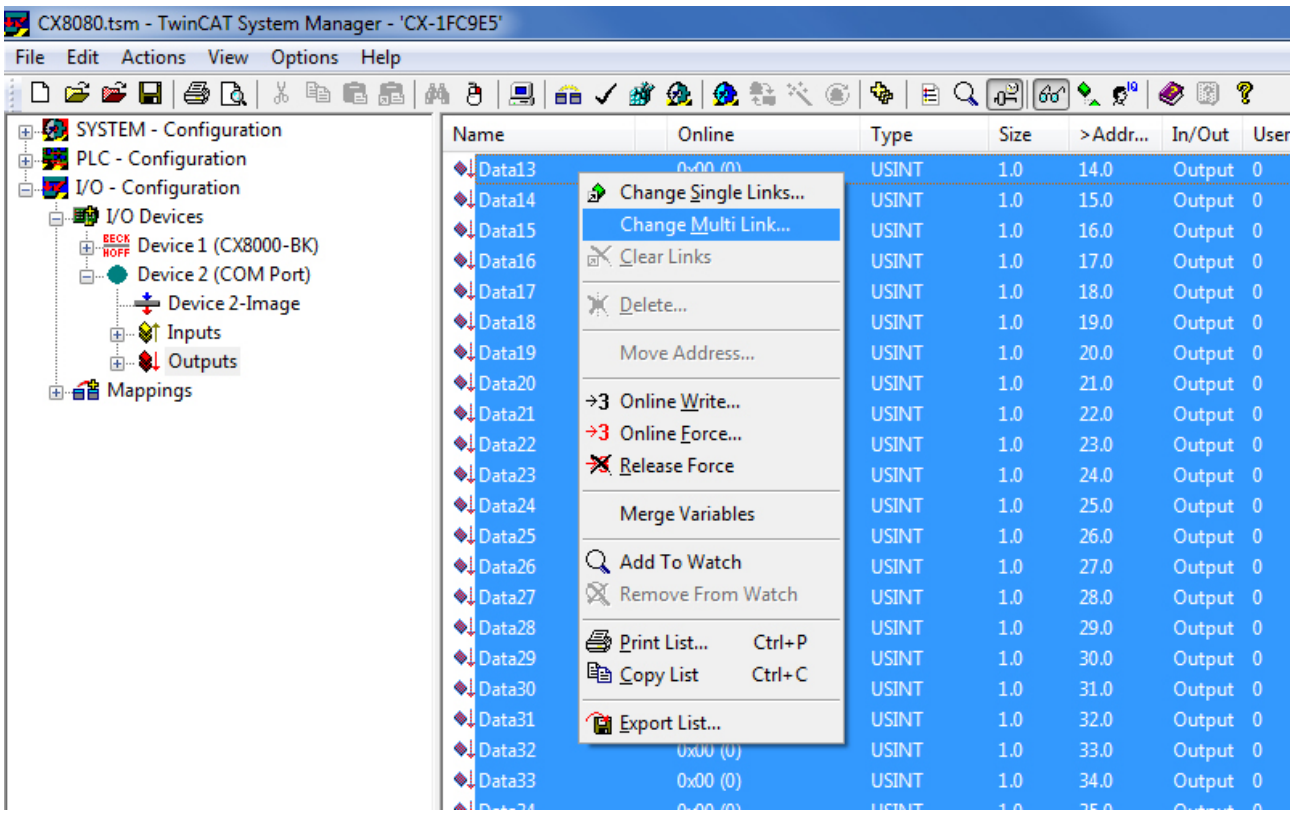
After that the data bytes can be conveniently linked to the corresponding variables by multi-link. The selection of several variables can be achieved by clicking on “Data1” and pressing the ↓ key with the Shift key pressed.

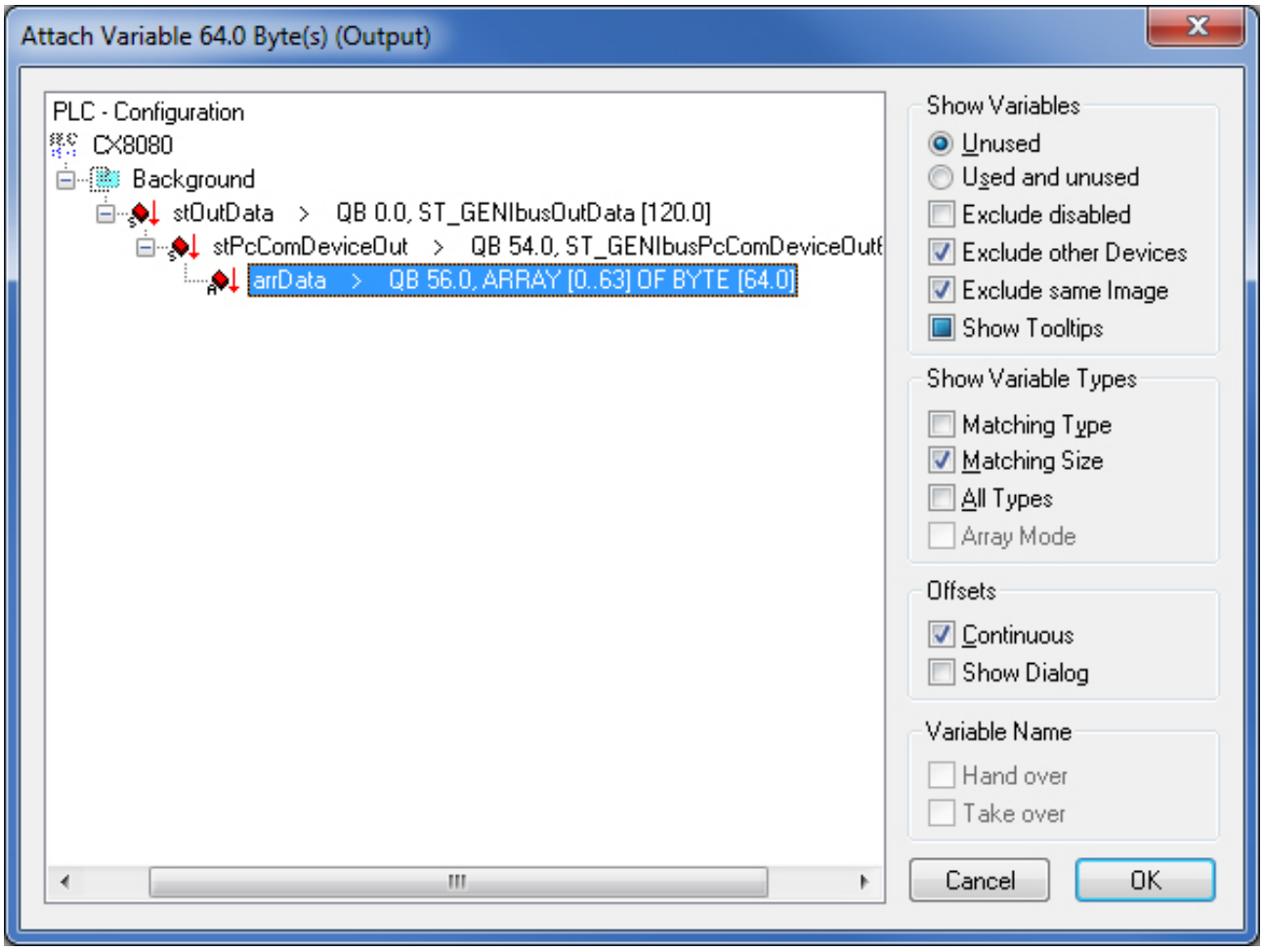


The output variables must be linked in the same way:




CX8080.tsm - TwinCAT System Manager - 'CX-1FC9E5'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (CX8000-BK)
Device 2 (COM Port)
Device 2-Image
Inputs
Outputs
Mappings
Name Online Type Size >Addr... In/Out User
Ctrl 2.0 0.0 Output 0
Data1 1.0 2.0 Output 0
Data2 1.0 3.0 Output 0
Data3 1.0 4.0 Output 0
Data4 1.0 5.0 Output 0
Data5 1.0 6.0 Output 0
Data6 1.0 7.0 Output 0
Data7 1.0 8.0 Output 0
Data8 1.0 9.0 Output 0
Data9 1.0 10.0 Output 0
Data10 1.0 11.0 Output 0
Data11 1.0 12.0 Output 0
Data12 1.0 13.0 Output 0
Data13 1.0 14.0 Output 0
Data14 1.0 15.0 Output 0
Data15 1.0 16.0 Output 0
Data16 1.0 17.0 Output 0
Data17 1.0 18.0 Output 0
Data18 1.0 19.0 Output 0
Data19 1.0 20.0 Output 0
Data20 0x00 (0) USINT 1.0 21.0 Output 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in
Attach Variable Ctrl (Output)
PLC - Configuration
CX8080
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
wCtrl > QB 6.0, WORD [2.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
wCtrl > QB 30.0, WORD [2.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutI
wCtrl > QB 54.0, WORD [2.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK


CX8080.tsm - TwinCAT System Manager - 'CX-1FC9E5'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (CX8000-BK)
Device 2 (COM Port)
Device 2-Image
Inputs
Outputs
Mappings
Name Online Type Size >Addr... In/Out User
Data13 0x00 (0) USINT 1.0 14.0 Output 0
Data14 USINT 1.0 15.0 Output 0
Data15 USINT 1.0 16.0 Output 0
Data16 USINT 1.0 17.0 Output 0
Data17 USINT 1.0 18.0 Output 0
Data18 USINT 1.0 19.0 Output 0
Data19 USINT 1.0 20.0 Output 0
Data20 USINT 1.0 21.0 Output 0
Data21 USINT 1.0 22.0 Output 0
Data22 USINT 1.0 23.0 Output 0
Data23 USINT 1.0 24.0 Output 0
Data24 USINT 1.0 25.0 Output 0
Data25 USINT 1.0 26.0 Output 0
Data26 USINT 1.0 27.0 Output 0
Data27 USINT 1.0 28.0 Output 0
Data28 USINT 1.0 29.0 Output 0
Data29 USINT 1.0 30.0 Output 0
Data30 USINT 1.0 31.0 Output 0
Data31 USINT 1.0 32.0 Output 0
Data32 0x00 (0) USINT 1.0 33.0 Output 0
Data33 0x00 (0) USINT 1.0 34.0 Output 0
Change Single Links...
Change Multi Link...
Clear Links
Delete...
Move Address...
Online Write...
Online Force...
Release Force
Merge Variables
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...


Attach Variable 64.0 Byte(s) (Output)
PLC - Configuration
CX8080
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutt
arrData > QB 56.0, ARRAY [0..63] OF BYTE [64.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK

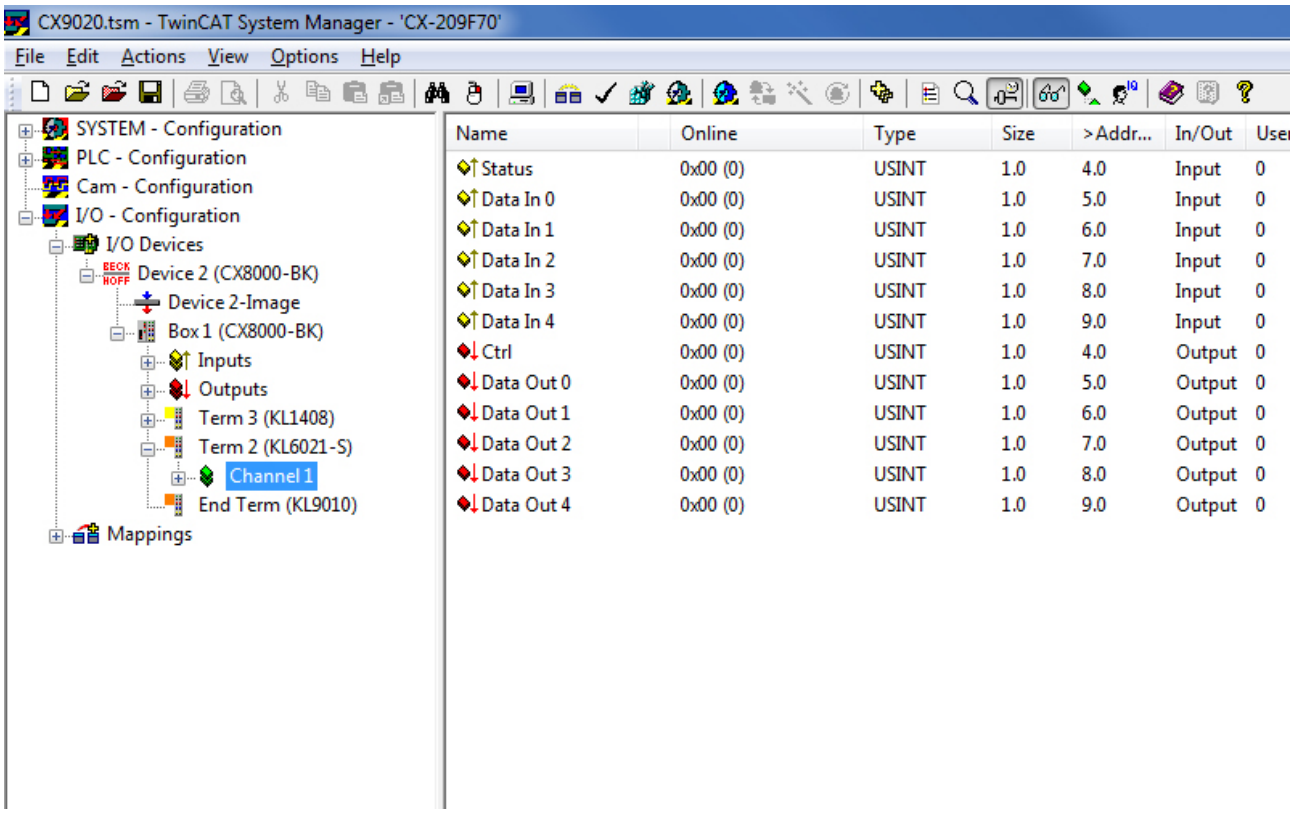
4.6 Linking the communication variables when using a KL6021

The requirement for the linking of the process image is that the terminal is preset to 5 bytes in advance. Unlike the PC interface, this cannot be configured in the TwinCAT System Manager, but only via the KS2000 software. The communication parameters

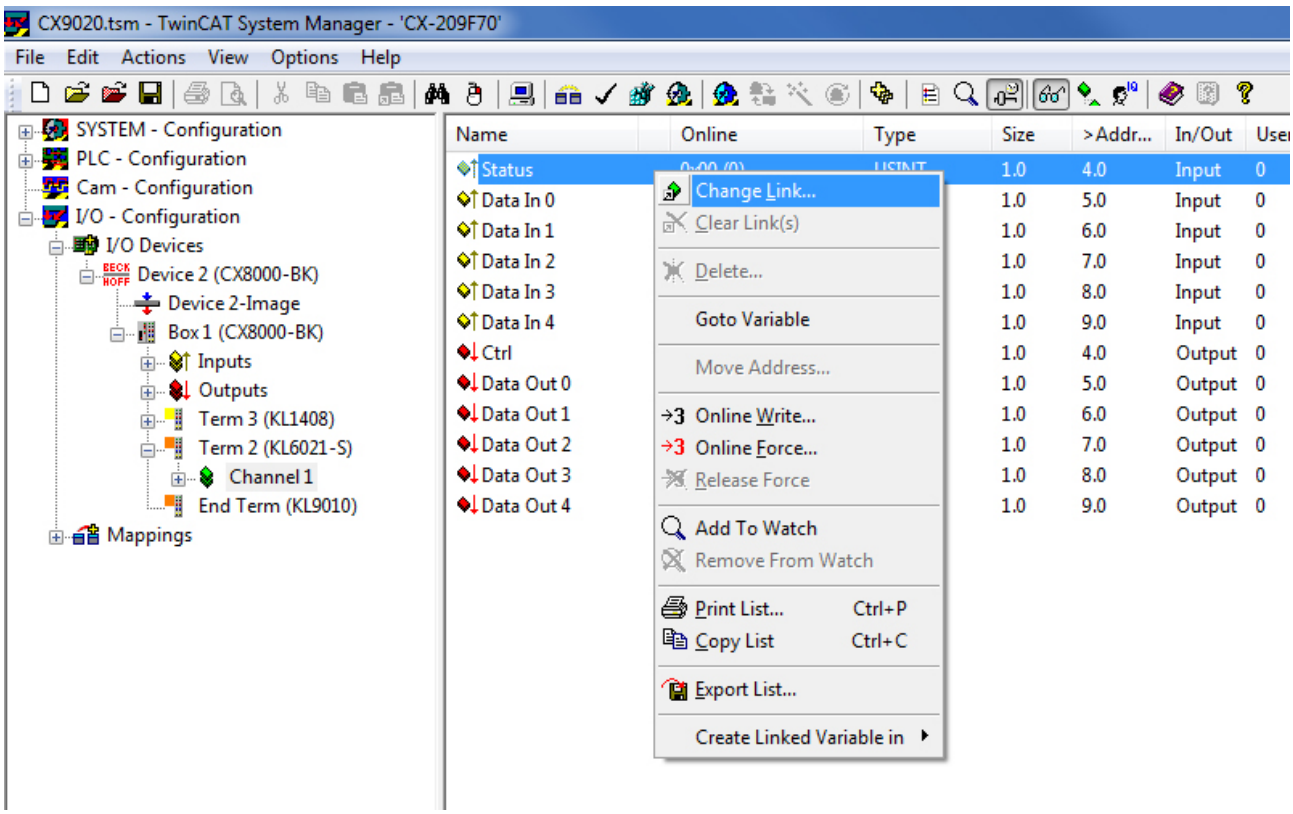
- Baud Rate: 9600 bits/s
- Data bits: 8
- Parity: None
- Stop bits: 1

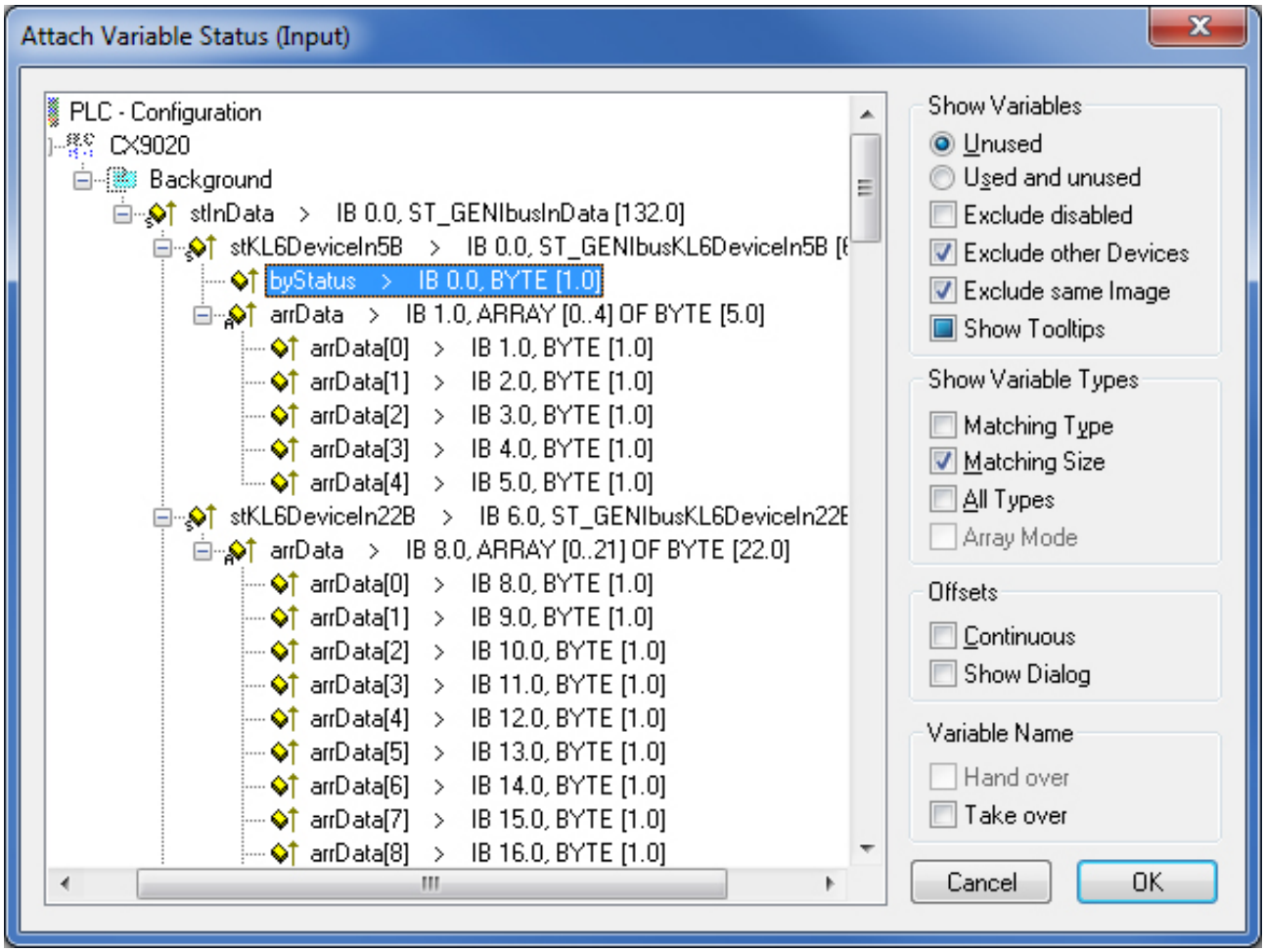
are automatically set by the PLC application, so that following the exchange of a terminal and a subsequent restart, this terminal is correctly set.

The linking of the process image to the PLC input and output variables can be accomplished most simply from the hardware side since multi-links are possible from there. The variables must be visible in the right-hand side of the TwinCAT System Manager for this.

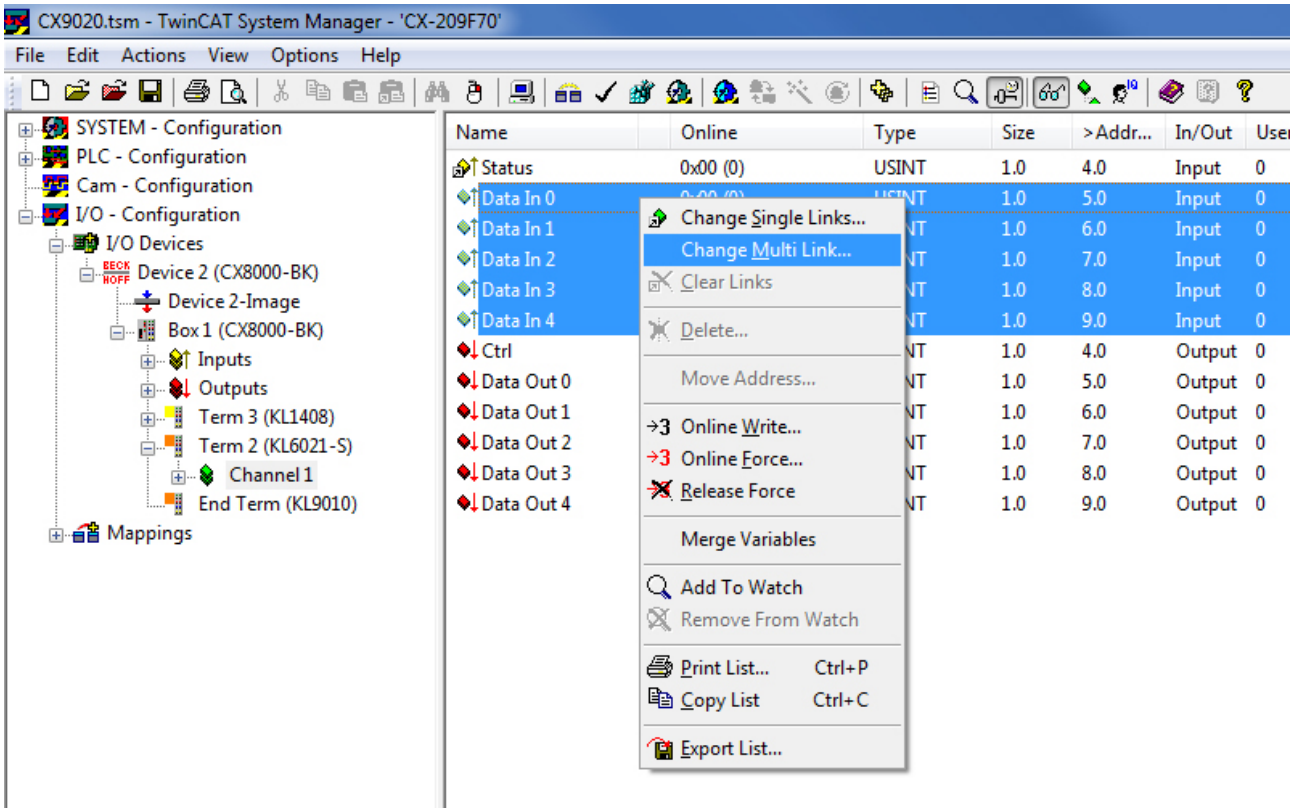


First, just the status is linked with the status variable of the communication input. Note that when doing so the structure for PC communication must be selected.

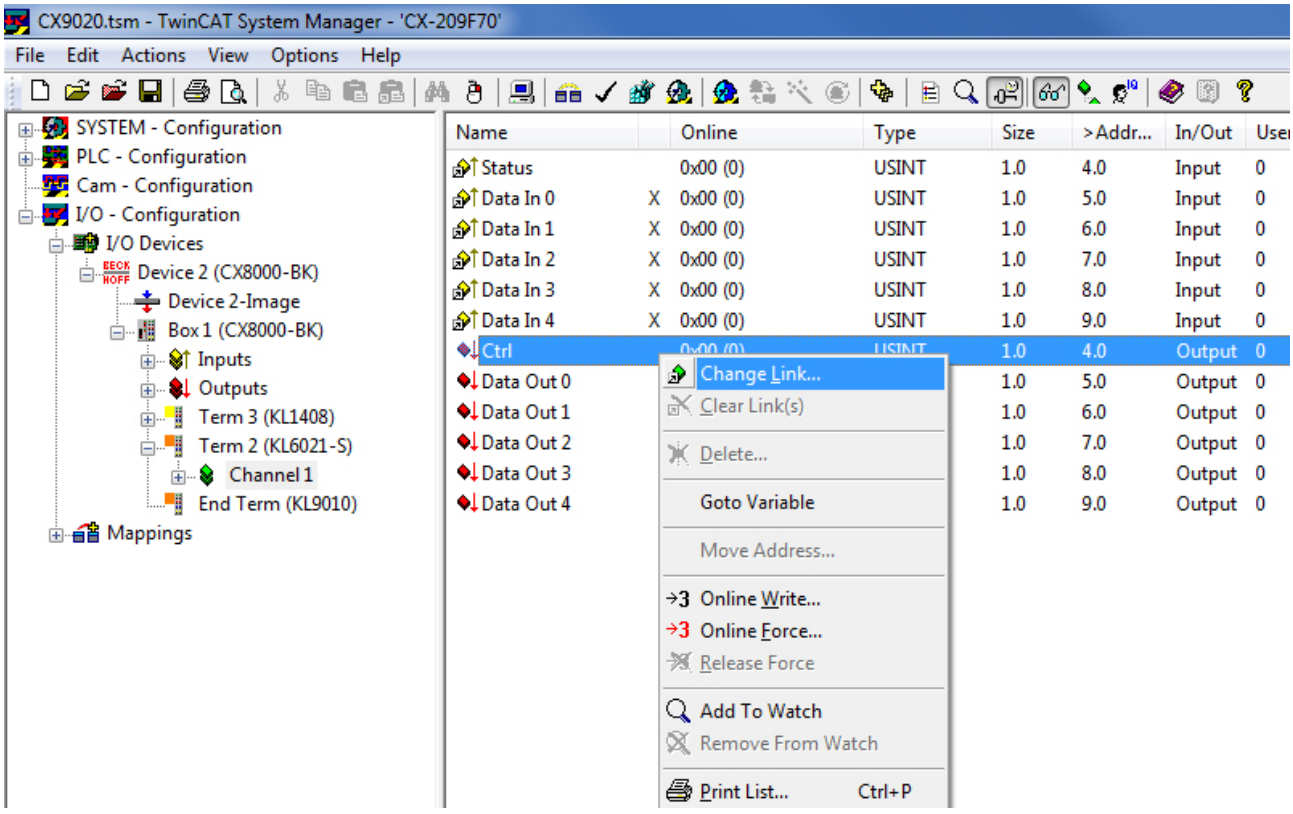


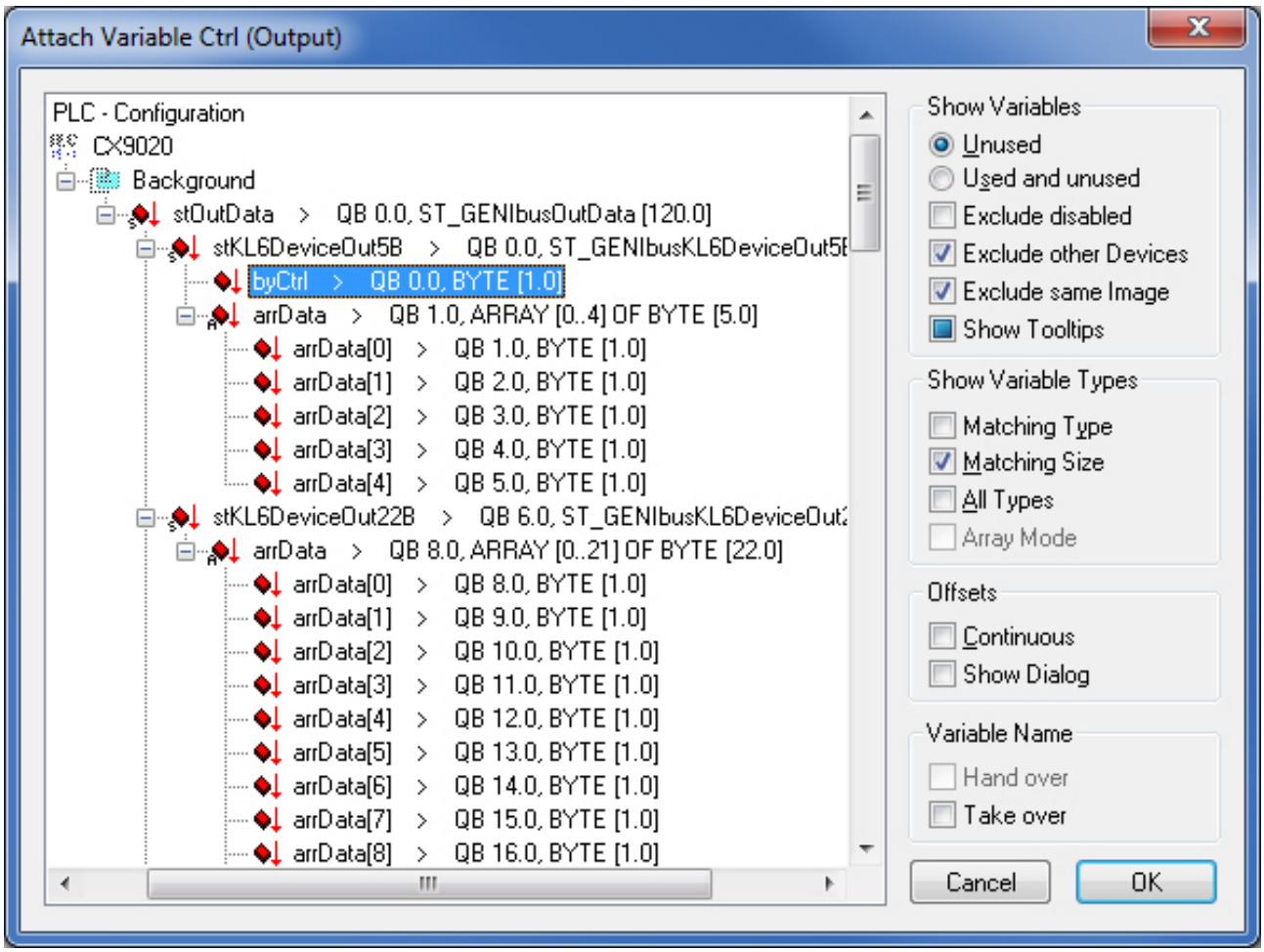


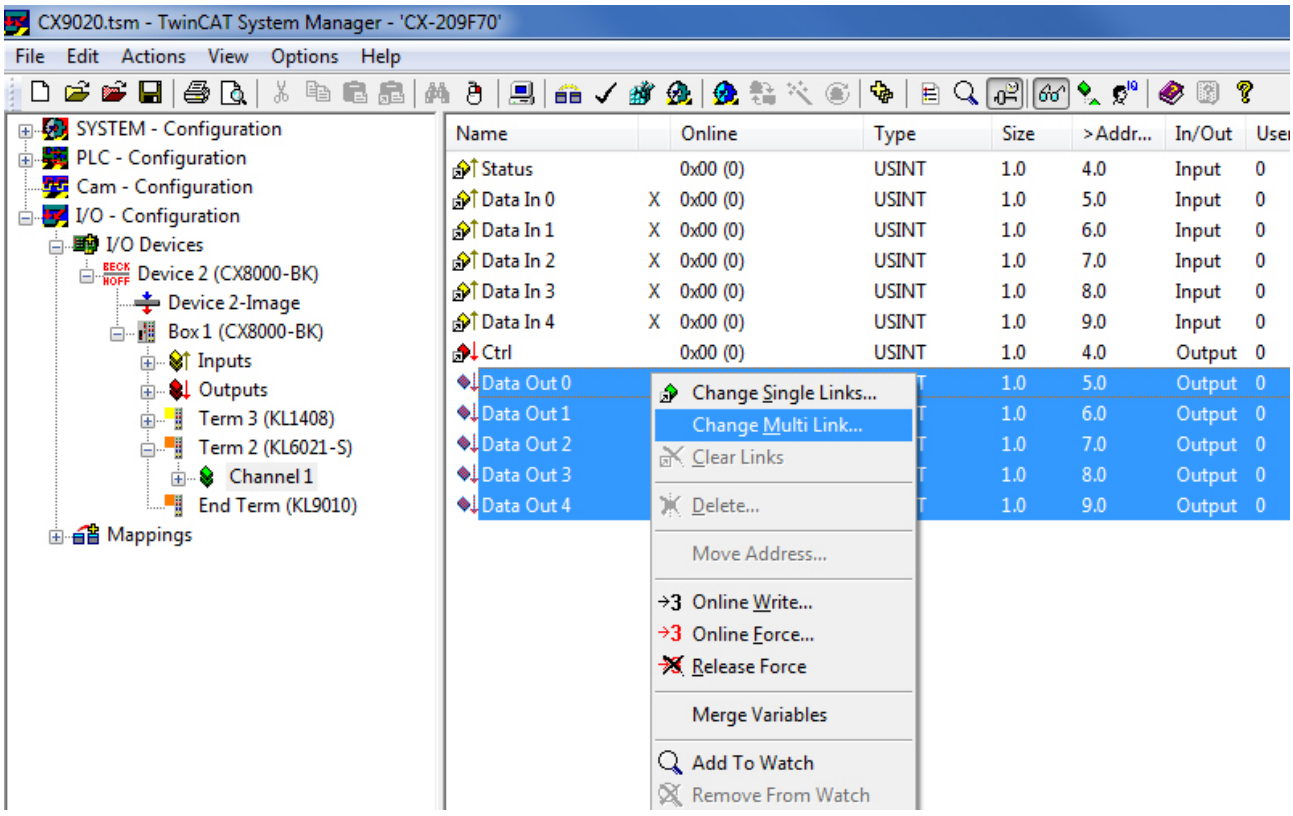
After that the data bytes can be conveniently linked to the corresponding variables by multi-link. The selection of several variables can be achieved by clicking on “Data1” and pressing the ↓ key with the Shift key pressed.

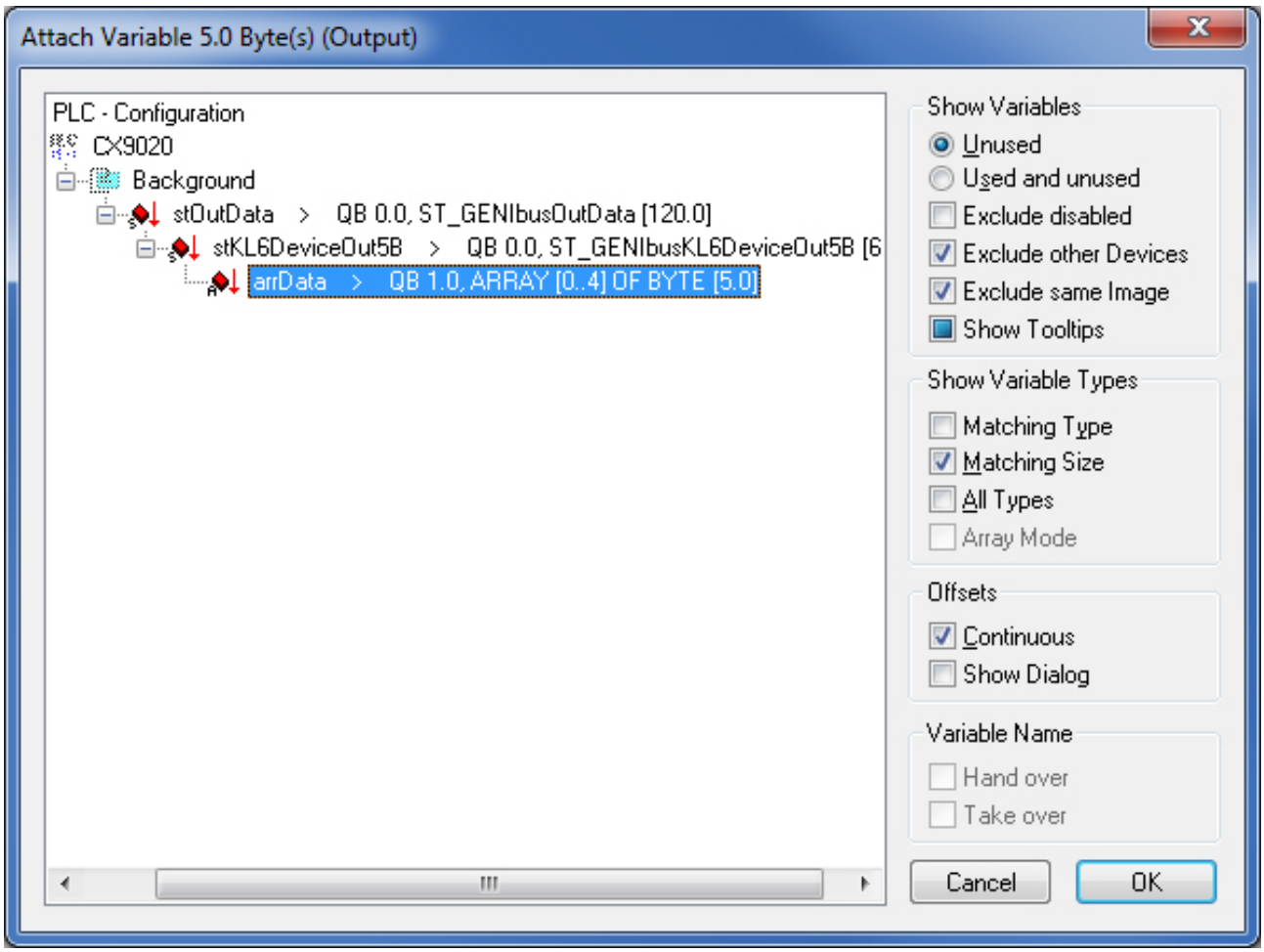


The output variables must be linked in the same way:




Attach Variable Ctrl (Output)
PLC - Configuration
CX9020
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut5B > QB 0.0, ST_GENIbusKL6DeviceOut5B
byCtrl > QB 0.0, BYTE [1.0]
arrData > QB 1.0, ARRAY [0..4] OF BYTE [5.0]
arrData[0] > QB 1.0, BYTE [1.0]
arrData[1] > QB 2.0, BYTE [1.0]
arrData[2] > QB 3.0, BYTE [1.0]
arrData[3] > QB 4.0, BYTE [1.0]
arrData[4] > QB 5.0, BYTE [1.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut2
arrData > QB 8.0, ARRAY [0..21] OF BYTE [22.0]
arrData[0] > QB 8.0, BYTE [1.0]
arrData[1] > QB 9.0, BYTE [1.0]
arrData[2] > QB 10.0, BYTE [1.0]
arrData[3] > QB 11.0, BYTE [1.0]
arrData[4] > QB 12.0, BYTE [1.0]
arrData[5] > QB 13.0, BYTE [1.0]
arrData[6] > QB 14.0, BYTE [1.0]
arrData[7] > QB 15.0, BYTE [1.0]
arrData[8] > QB 16.0, BYTE [1.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK


CX9020.tsm - TwinCAT System Manager - 'CX-209F70'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
Cam - Configuration
I/O - Configuration
I/O Devices
Device 2 (CX8000-BK)
Device 2-Image
Box 1 (CX8000-BK)
Inputs
Outputs
Term 3 (KL1408)
Term 2 (KL6021-S)
Channel 1
End Term (KL9010)
Mappings
Name Online Type Size >Addr... In/Out
Status 0x00 (0) USINT 1.0 4.0 Input 0
Data In 0 0x00 (0) USINT 1.0 5.0 Input 0
Data In 1 0x00 (0) USINT 1.0 6.0 Input 0
Data In 2 0x00 (0) USINT 1.0 7.0 Input 0
Data In 3 0x00 (0) USINT 1.0 8.0 Input 0
Data In 4 0x00 (0) USINT 1.0 9.0 Input 0
Ctrl 0x00 (0) USINT 1.0 4.0 Output 0
Data Out 0 1.0 5.0 Output 0
Data Out 1 1.0 6.0 Output 0
Data Out 2 1.0 7.0 Output 0
Data Out 3 1.0 8.0 Output 0
Data Out 4 1.0 9.0 Output 0
Change Single Links...
Change Multi Link...
Clear Links
Delete...
Move Address...
Online Write...
Online Force...
Release Force
Merge Variables
Add To Watch
Remove From Watch



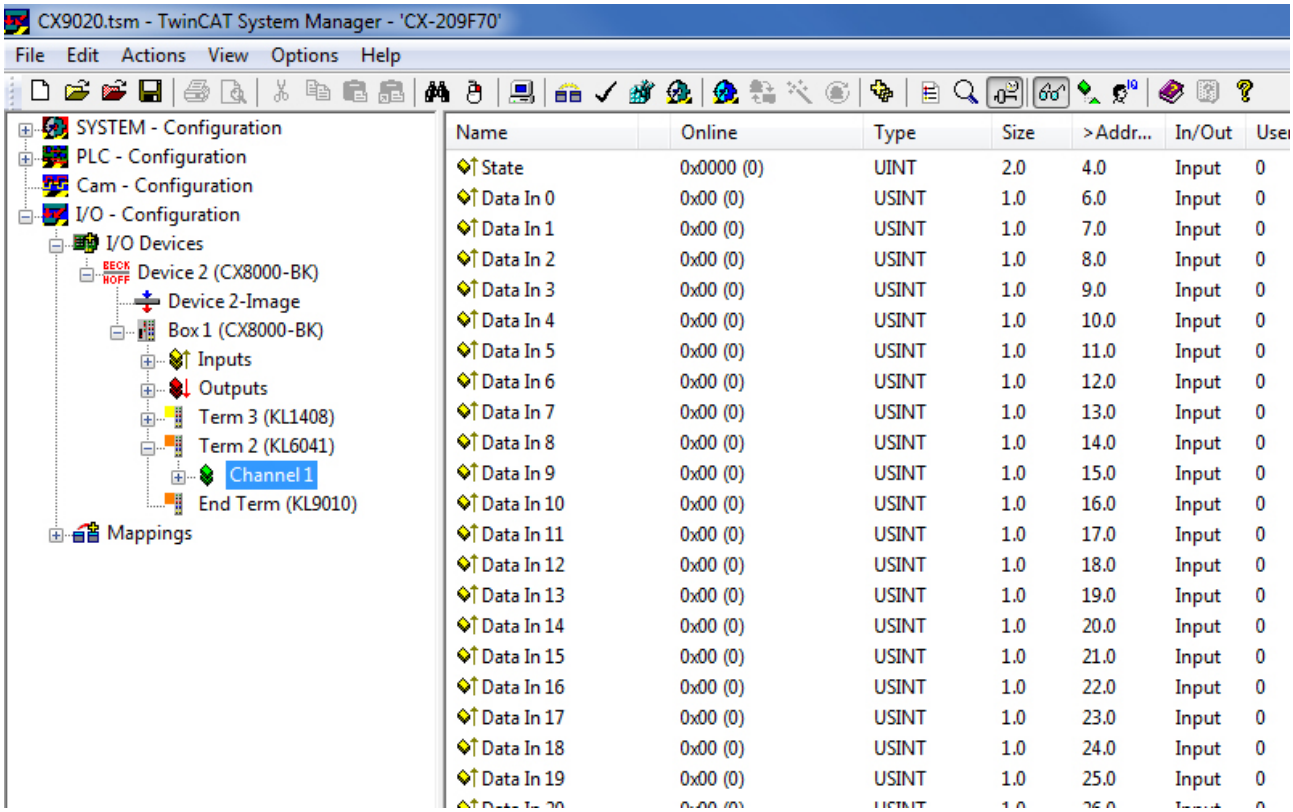
4.7 Linking the communication variables when using a KL6041

The requirement for the linking of the process image is that the terminal is preset to 22 bytes in advance. Unlike the PC interface, this cannot be configured in the TwinCAT System Manager, but only via the KS2000 software. The communication parameters

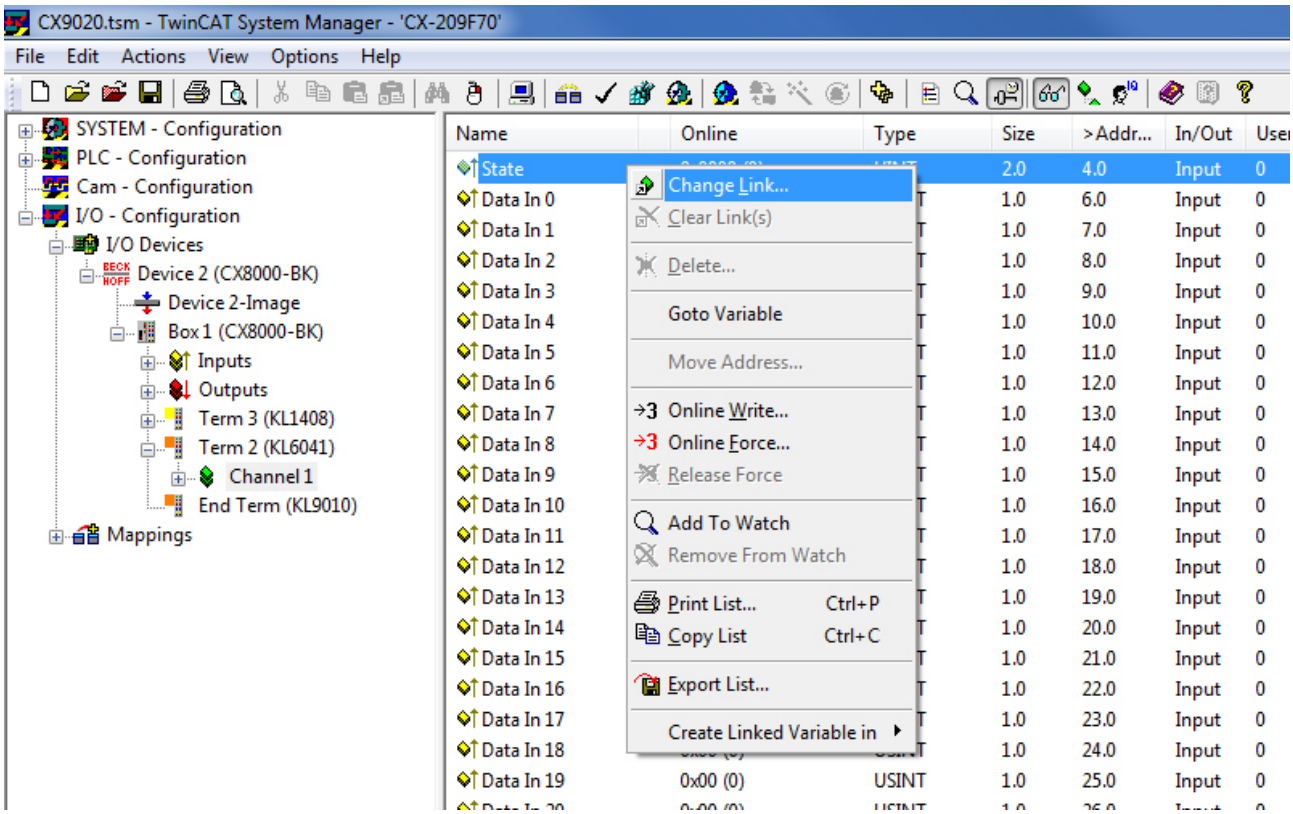
- Baud Rate: 9600 bits/s
- Data bits: 8
- Parity: None
- Stop bits: 1

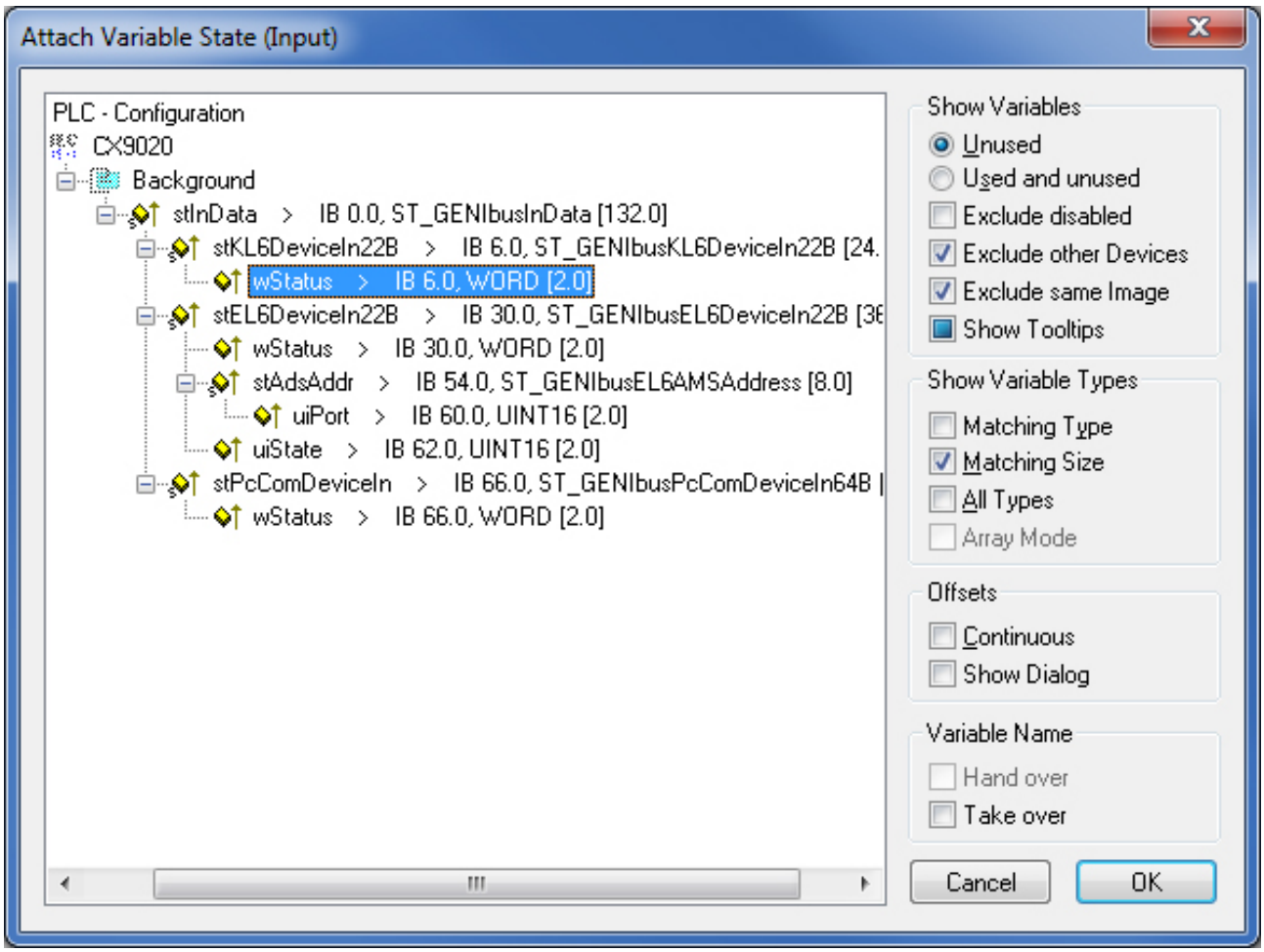
are automatically set by the PLC application, so that following the exchange of a terminal and a subsequent restart, this terminal is correctly set.

The linking of the process image to the PLC input and output variables can be accomplished most simply from the hardware side, since multi-links are possible from there. The variables must be visible in the right-hand side of the TwinCAT System Manager for this.

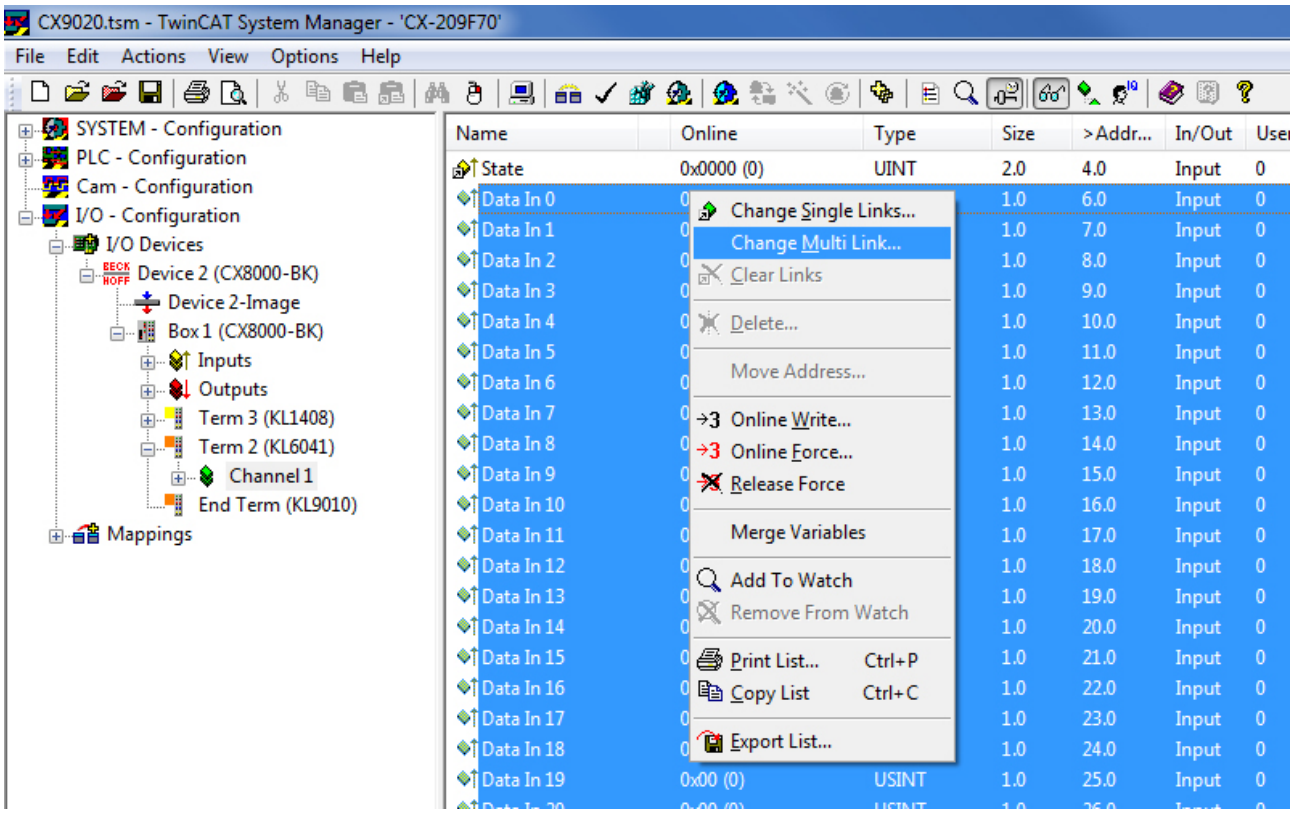


First, just the status is linked with the status variable of the communication input. Note that when doing so the structure for PC communication must be selected.

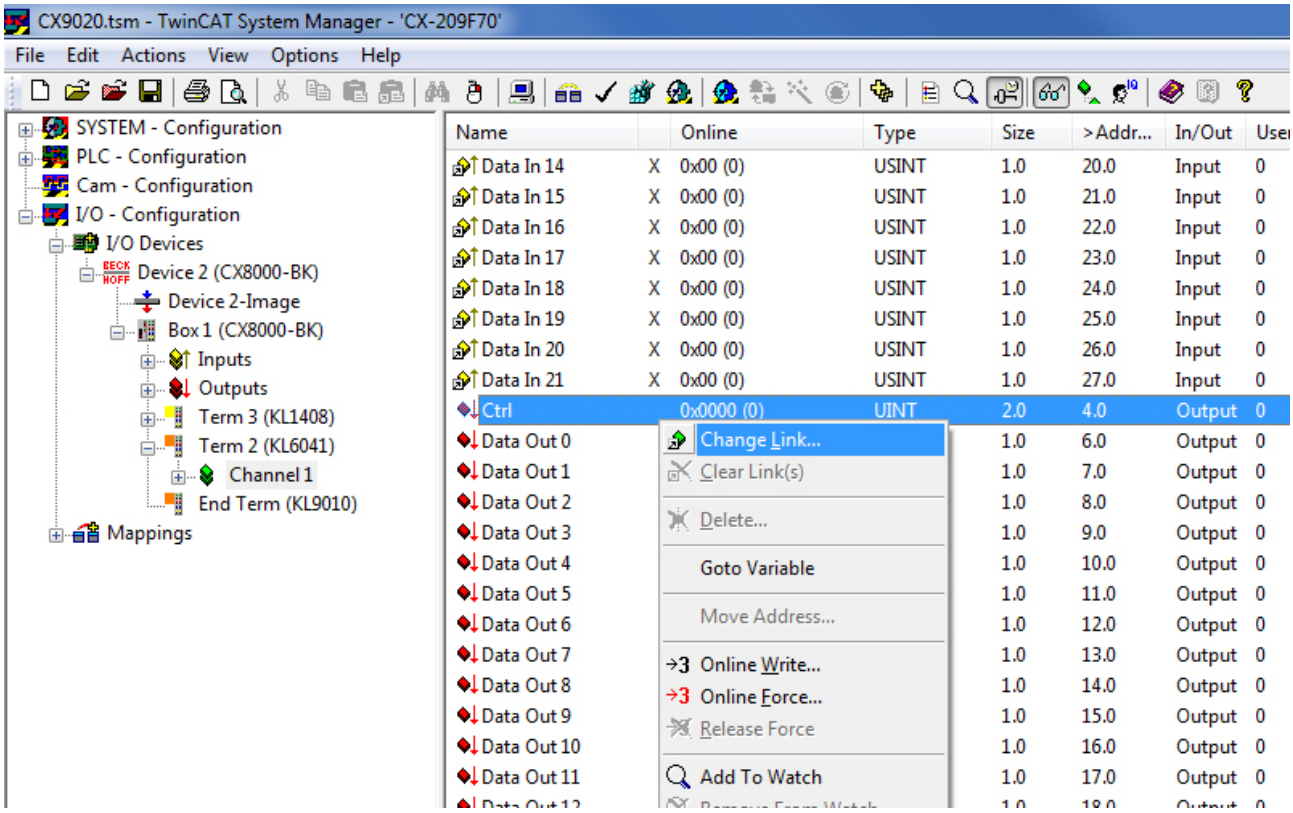


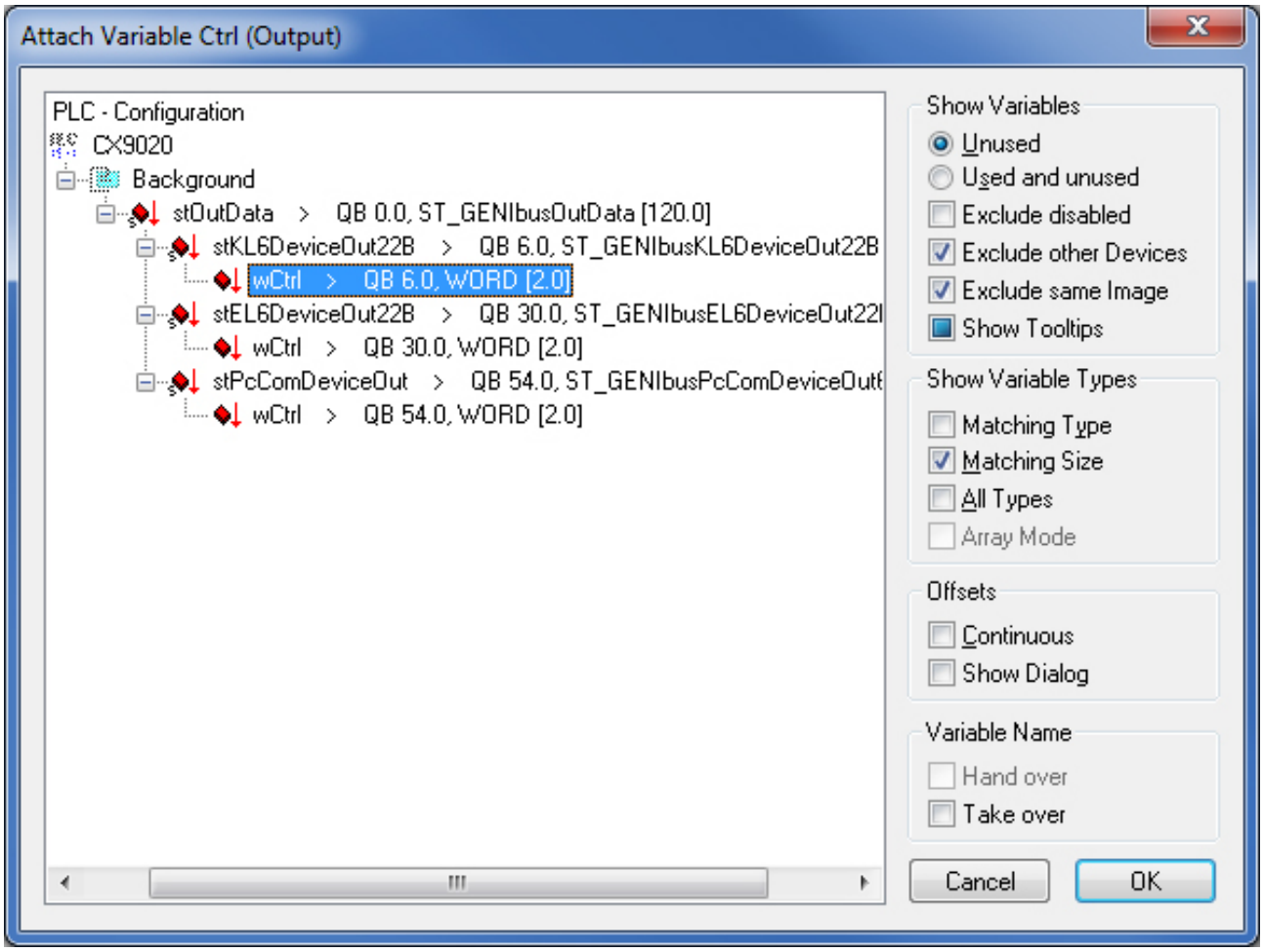


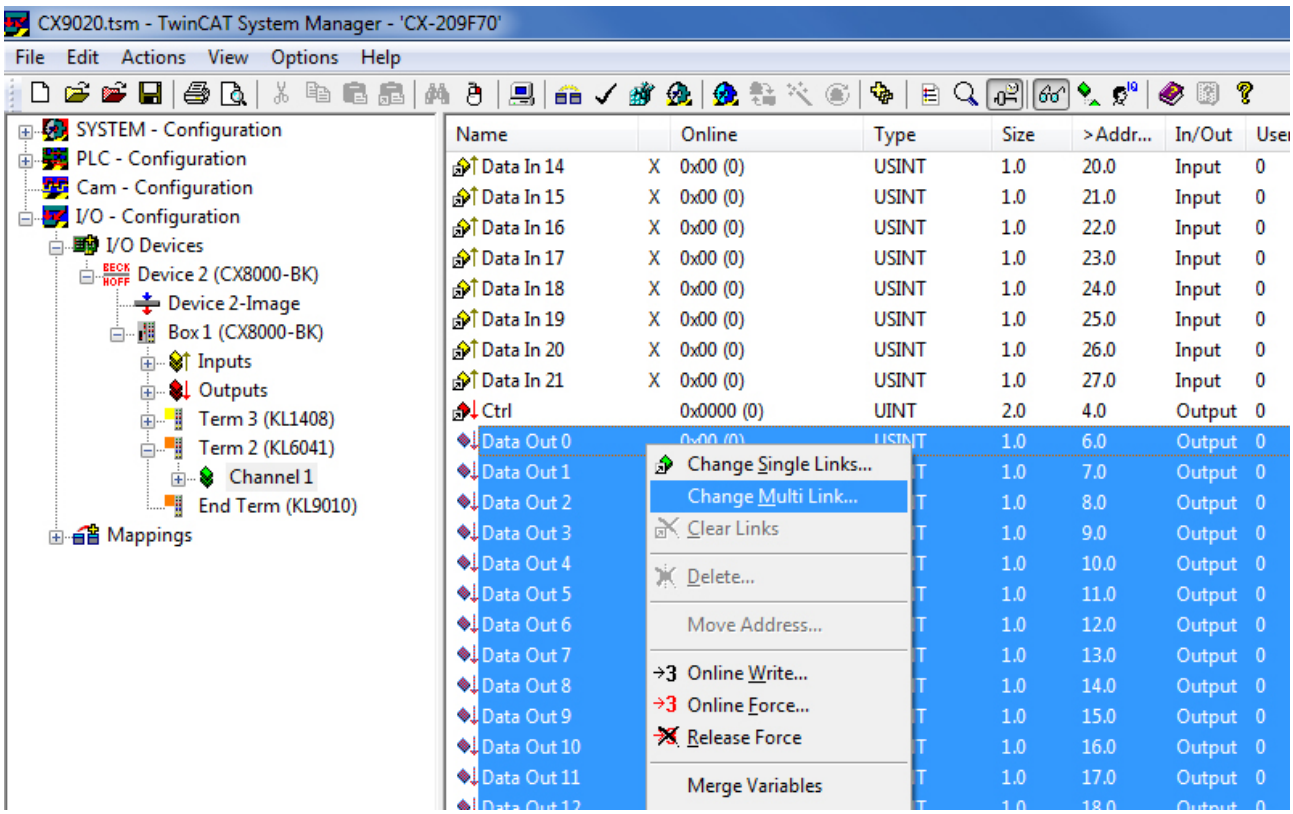
After that the data bytes can be conveniently linked to the corresponding variables by multi-link. The selection of several variables can be achieved by clicking on “Data1” and pressing the ↓ key with the Shift key pressed.

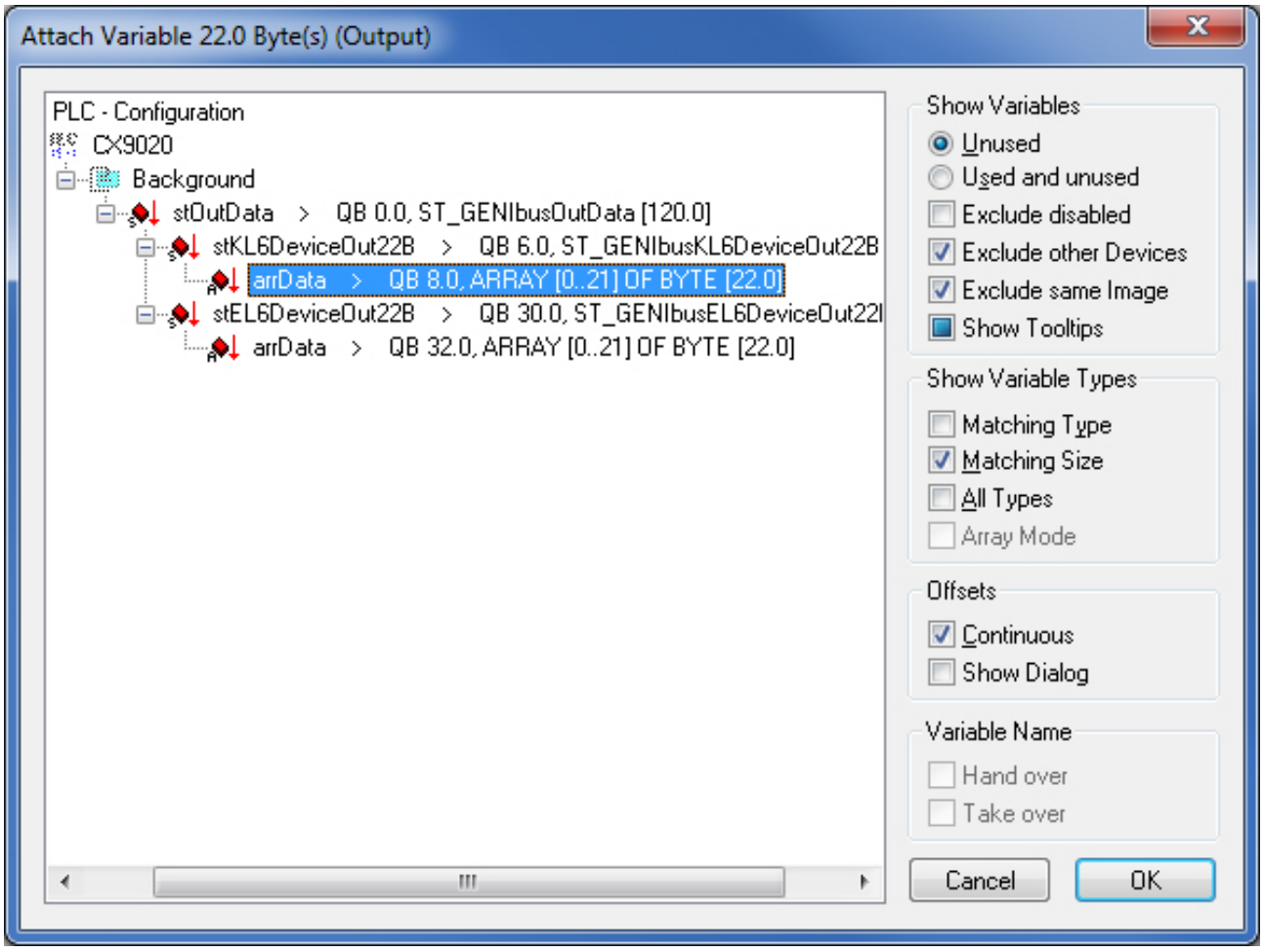


The output variables must be linked in the same way:






CX9020.tsm - TwinCAT System Manager - 'CX-209F70'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
Cam - Configuration
I/O - Configuration
I/O Devices
Device 2 (CX8000-BK)
Device 2-Image
Box 1 (CX8000-BK)
Inputs
Outputs
Term 3 (KL1408)
Term 2 (KL6041)
Channel 1
End Term (KL9010)
Mappings
Name Online Type Size >Addr... In/Out
Data In 14 X 0x00 (0) USINT 1.0 20.0 Input 0
Data In 15 X 0x00 (0) USINT 1.0 21.0 Input 0
Data In 16 X 0x00 (0) USINT 1.0 22.0 Input 0
Data In 17 X 0x00 (0) USINT 1.0 23.0 Input 0
Data In 18 X 0x00 (0) USINT 1.0 24.0 Input 0
Data In 19 X 0x00 (0) USINT 1.0 25.0 Input 0
Data In 20 X 0x00 (0) USINT 1.0 26.0 Input 0
Data In 21 X 0x00 (0) USINT 1.0 27.0 Input 0
Ctrl 0x0000 (0) UINT 2.0 4.0 Output 0
Data Out 0 1.0 6.0 Output 0
Data Out 1 1.0 7.0 Output 0
Data Out 2 1.0 8.0 Output 0
Data Out 3 1.0 9.0 Output 0
Data Out 4 1.0 10.0 Output 0
Data Out 5 1.0 11.0 Output 0
Data Out 6 1.0 12.0 Output 0
Data Out 7 1.0 13.0 Output 0
Data Out 8 1.0 14.0 Output 0
Data Out 9 1.0 15.0 Output 0
Data Out 10 1.0 16.0 Output 0
Data Out 11 1.0 17.0 Output 0
Change Single Links...
Change Multi Link...
Clear Links
Delete...
Move Address...
Online Write...
Online Force...
Release Force
Merge Variables


Attach Variable 22.0 Byte(s) (Output)
PLC - Configuration
CX9020
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
arrData > QB 8.0, ARRAY [0..21] OF BYTE [22.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
arrData > QB 32.0, ARRAY [0..21] OF BYTE [22.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK

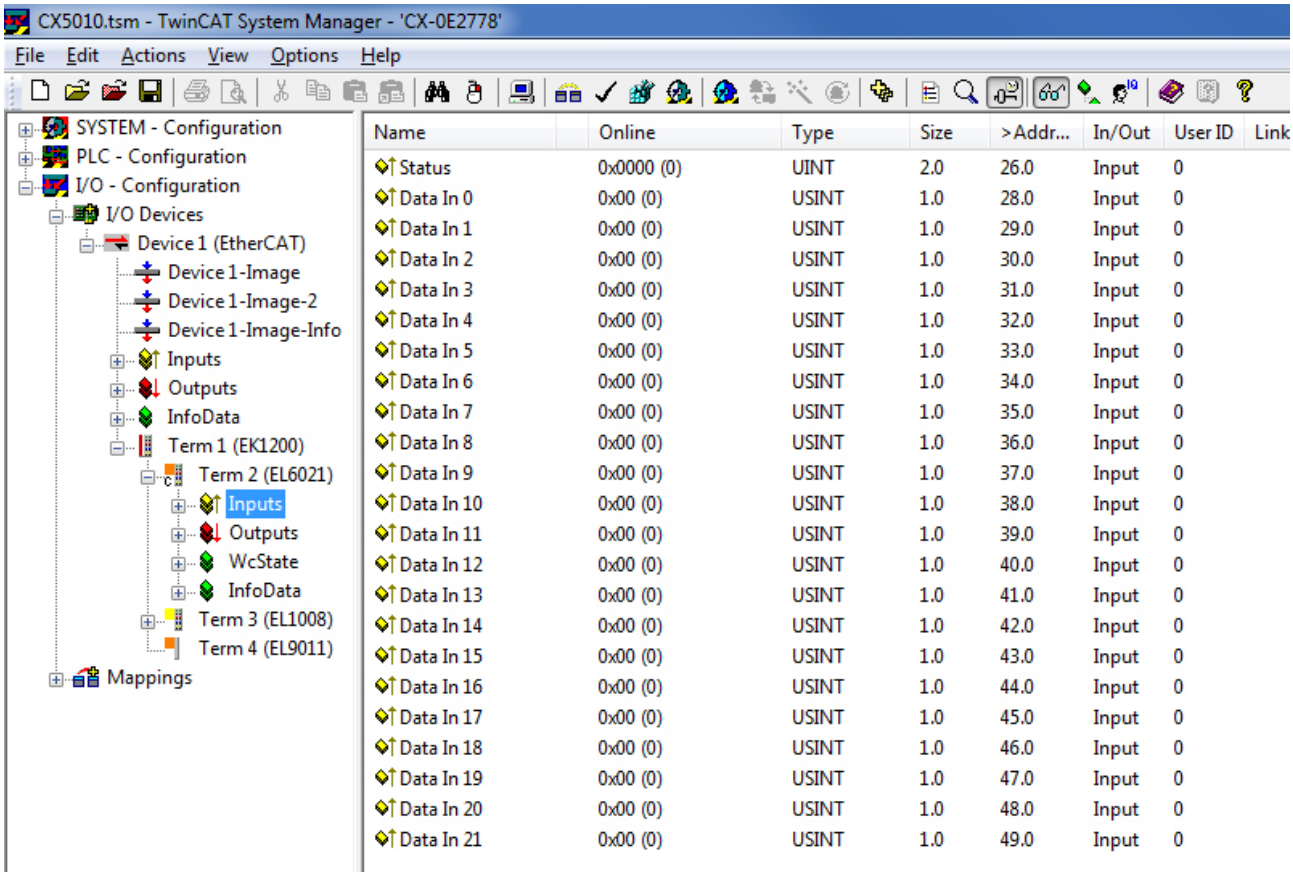
4.8 Configuration in the TwinCAT System Manager

Unlike the PC interface, the communication parameters

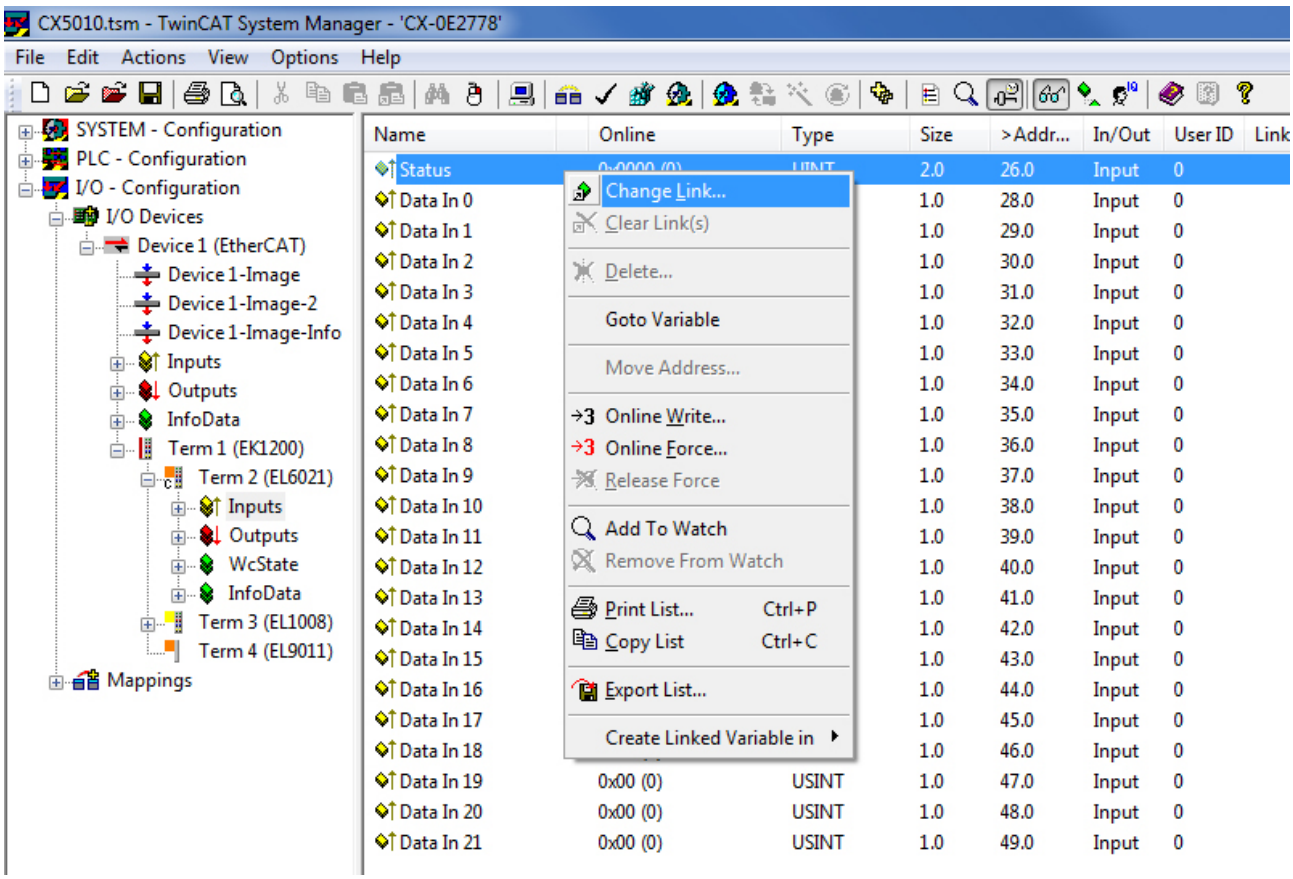
- Baud Rate: 9600 bits/s
- Data bits: 8
- Parity: None
- Stop bits: 1

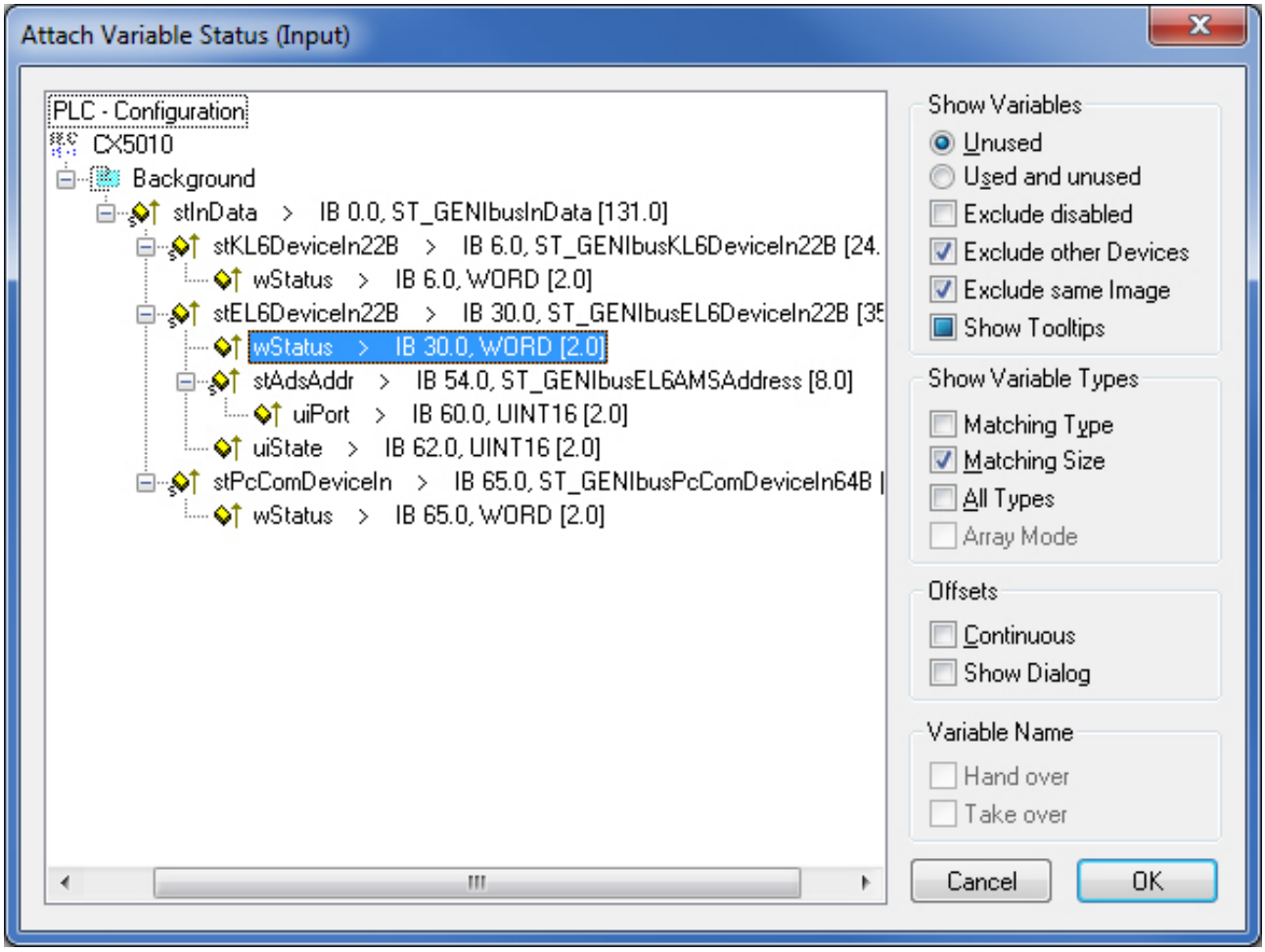
are automatically set by the PLC application, so that following the exchange of a terminal and a subsequent restart, this terminal is correctly set.

The linking of the process image to the PLC input and output variables can be accomplished most simply from the hardware side since multi-links are possible from there. The variables must be visible in the right-hand side of the TwinCAT System Manager for this.

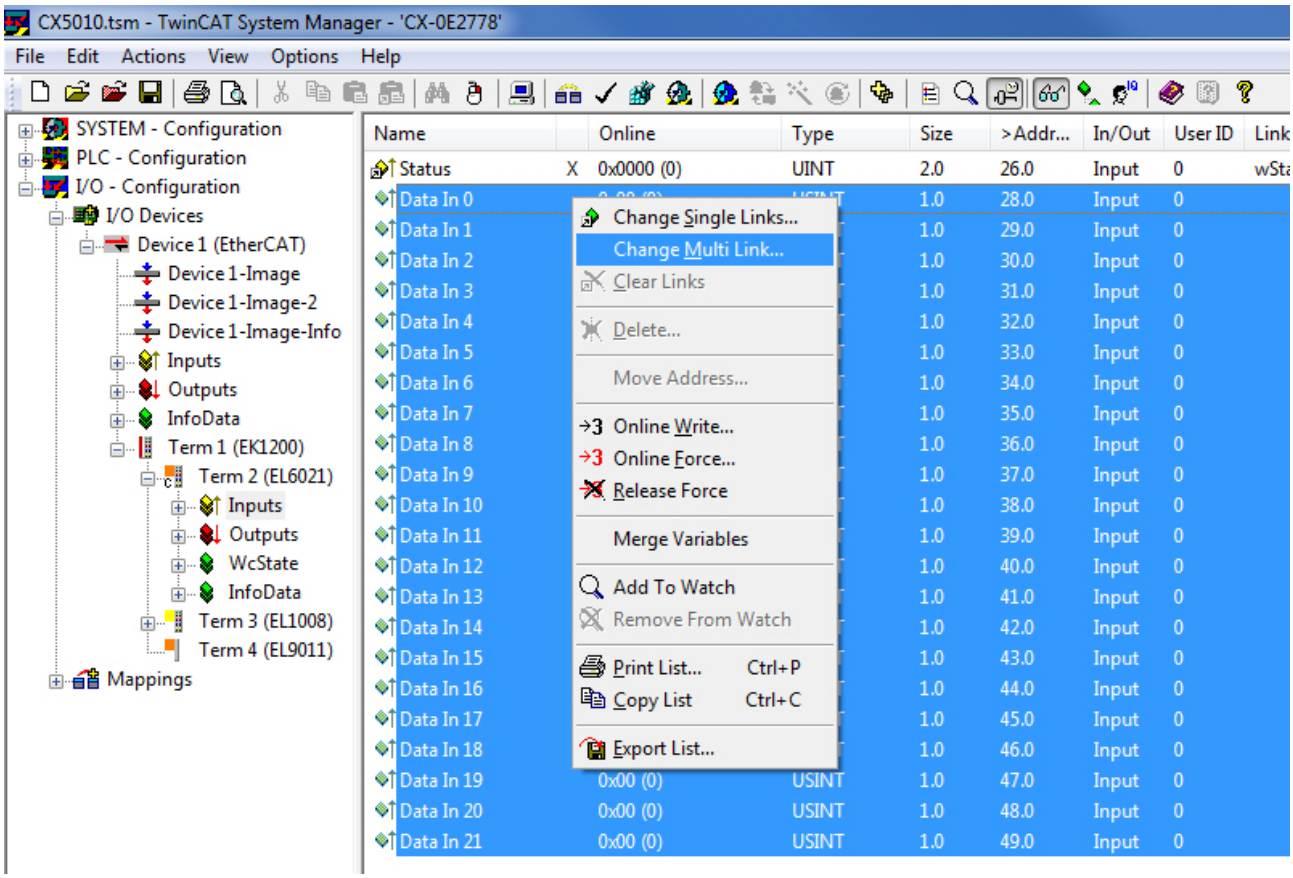


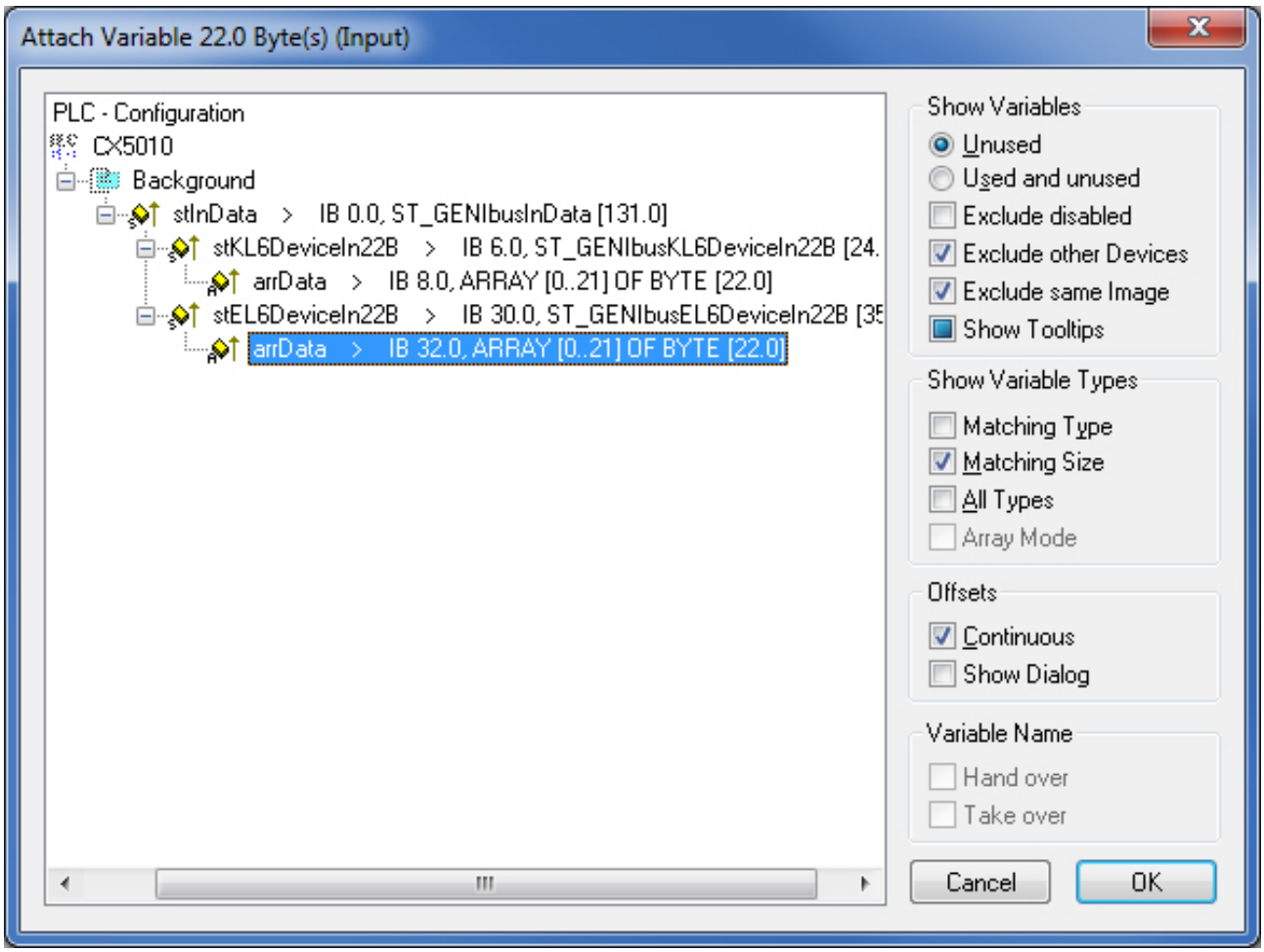
First, just the status is linked with the status variable of the communication input. Note that when doing so the structure for PC communication must be selected.


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
Status 2.0 26.0 Input 0
Data In 0 1.0 28.0 Input 0
Data In 1 1.0 29.0 Input 0
Data In 2 1.0 30.0 Input 0
Data In 3 1.0 31.0 Input 0
Data In 4 1.0 32.0 Input 0
Data In 5 1.0 33.0 Input 0
Data In 6 1.0 34.0 Input 0
Data In 7 1.0 35.0 Input 0
Data In 8 1.0 36.0 Input 0
Data In 9 1.0 37.0 Input 0
Data In 10 1.0 38.0 Input 0
Data In 11 1.0 39.0 Input 0
Data In 12 1.0 40.0 Input 0
Data In 13 1.0 41.0 Input 0
Data In 14 1.0 42.0 Input 0
Data In 15 1.0 43.0 Input 0
Data In 16 1.0 44.0 Input 0
Data In 17 1.0 45.0 Input 0
Data In 18 1.0 46.0 Input 0
Data In 19 0x00 (0) USINT 1.0 47.0 Input 0
Data In 20 0x00 (0) USINT 1.0 48.0 Input 0
Data In 21 0x00 (0) USINT 1.0 49.0 Input 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in

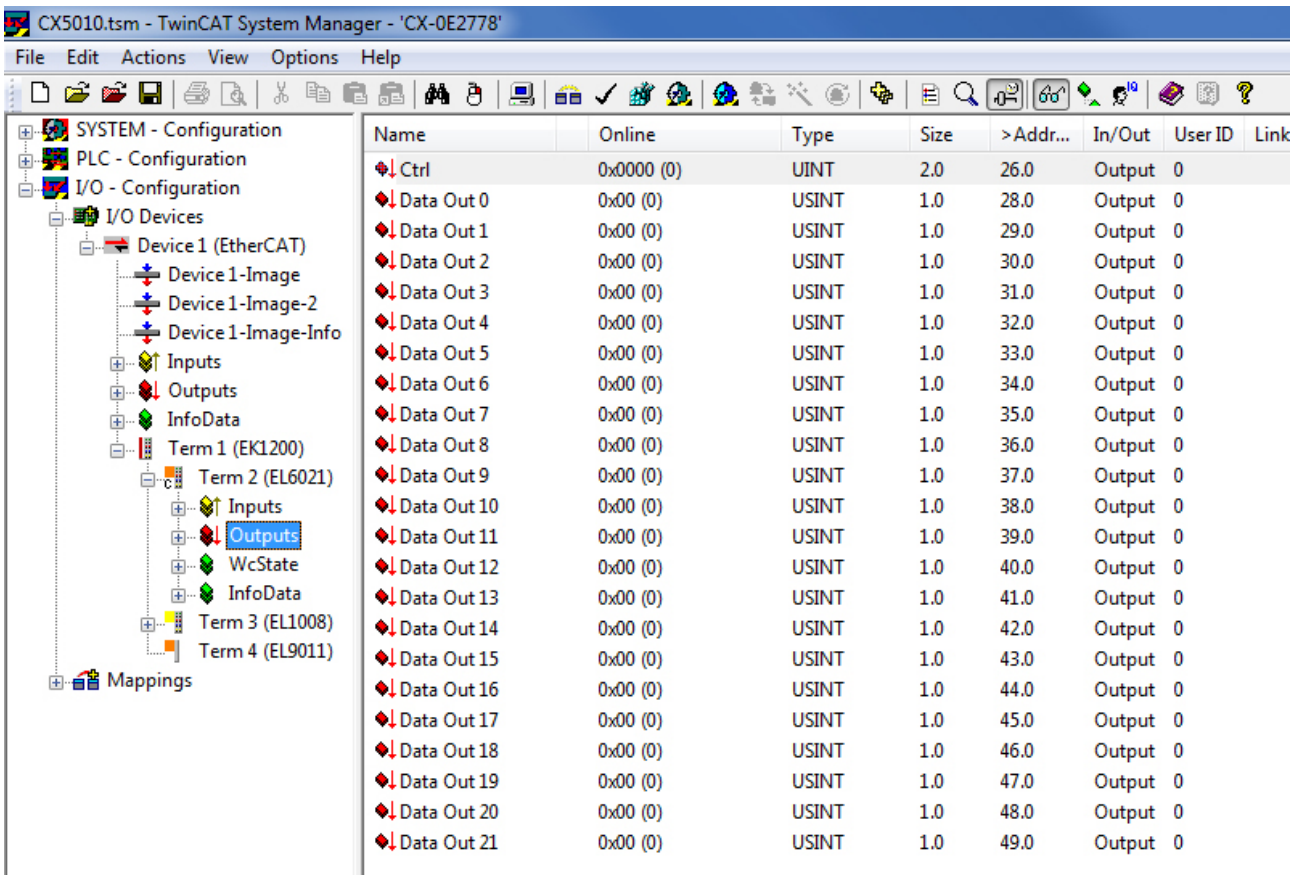

Attach Variable Status (Input)
PLC - Configuration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stKL6DeviceIn22B > IB 6.0, ST_GENIbusKL6DeviceIn22B [24.
wStatus > IB 6.0, WORD [2.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
wStatus > IB 30.0, WORD [2.0]
stAdsAddr > IB 54.0, ST_GENIbusEL6AMSAddress [8.0]
uiPort > IB 60.0, UINT16 [2.0]
uiState > IB 62.0, UINT16 [2.0]
stPcComDeviceIn > IB 65.0, ST_GENIbusPcComDeviceIn64B
wStatus > IB 65.0, WORD [2.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK

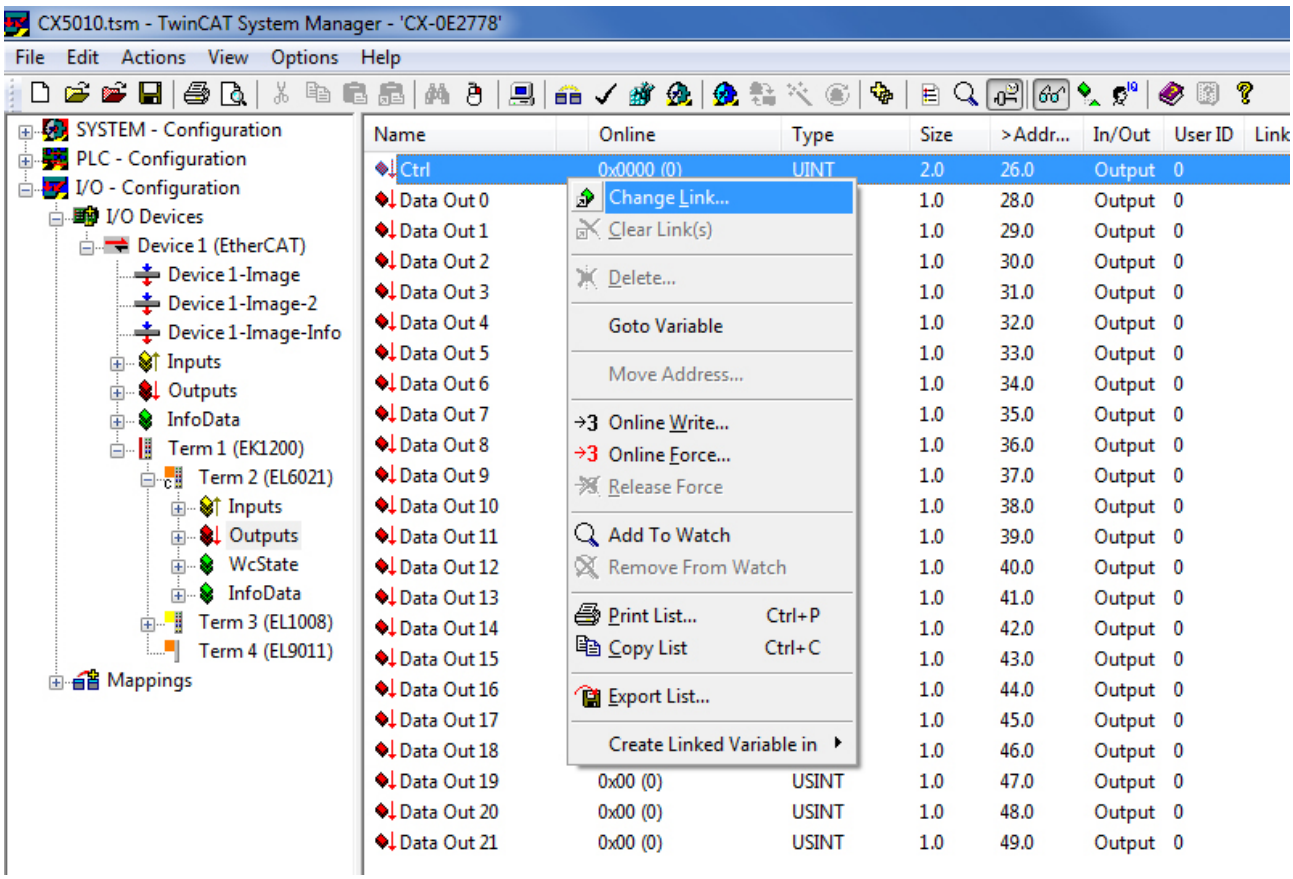
After that the data bytes can be conveniently linked to the corresponding variables by multi-link. The selection of several variables can be achieved by clicking on “Data1” and pressing the ↓ key with the Shift key pressed.

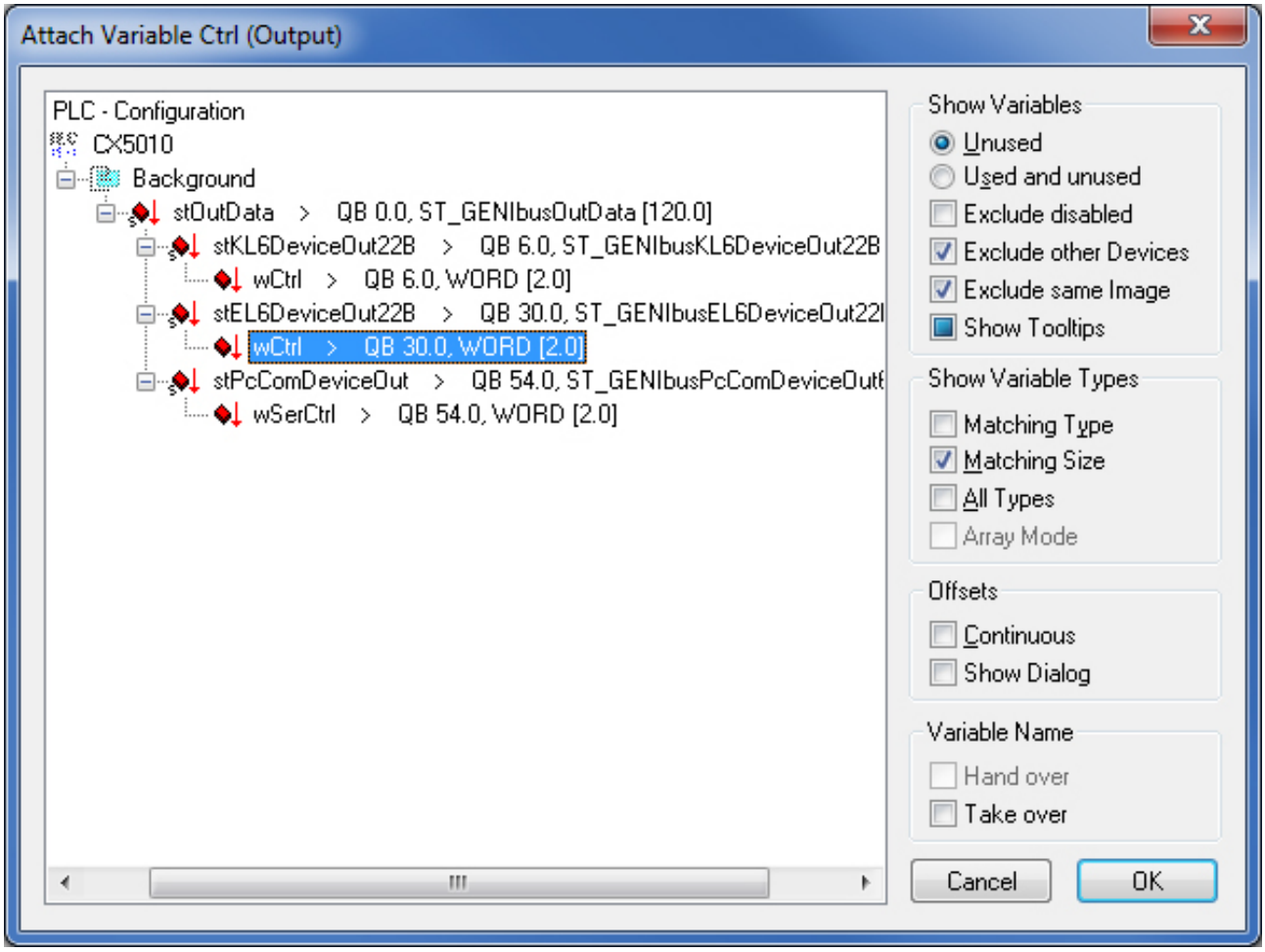


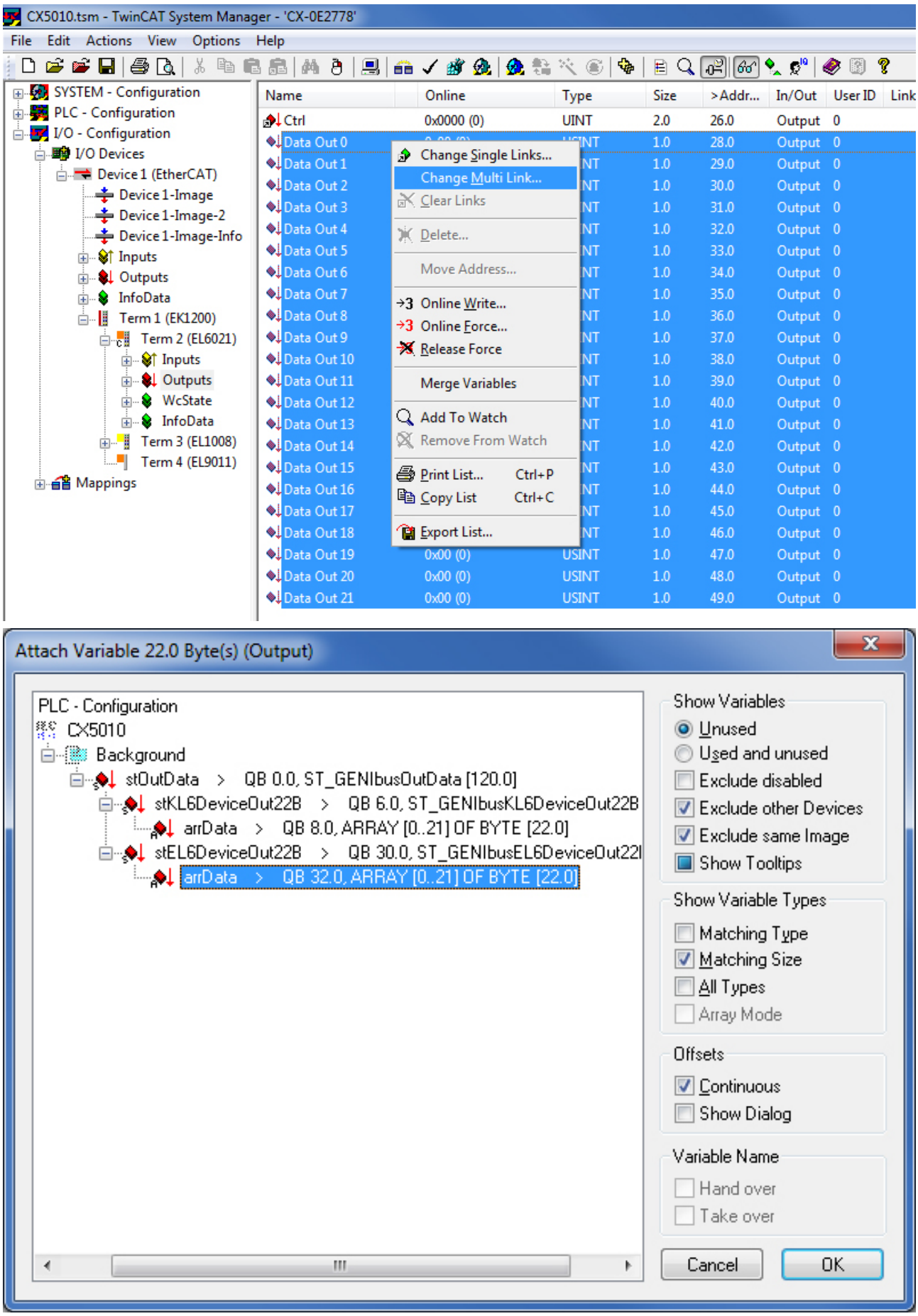


The output variables must be linked in the same way:

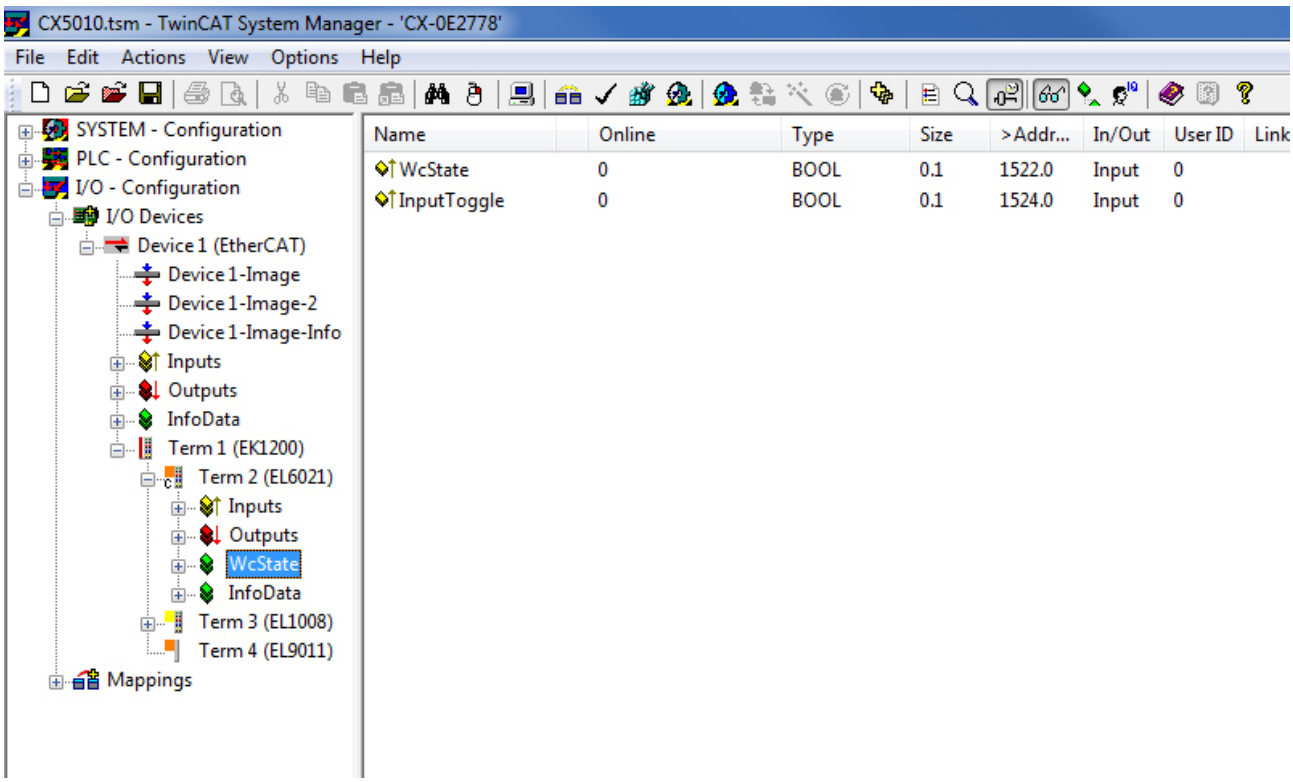


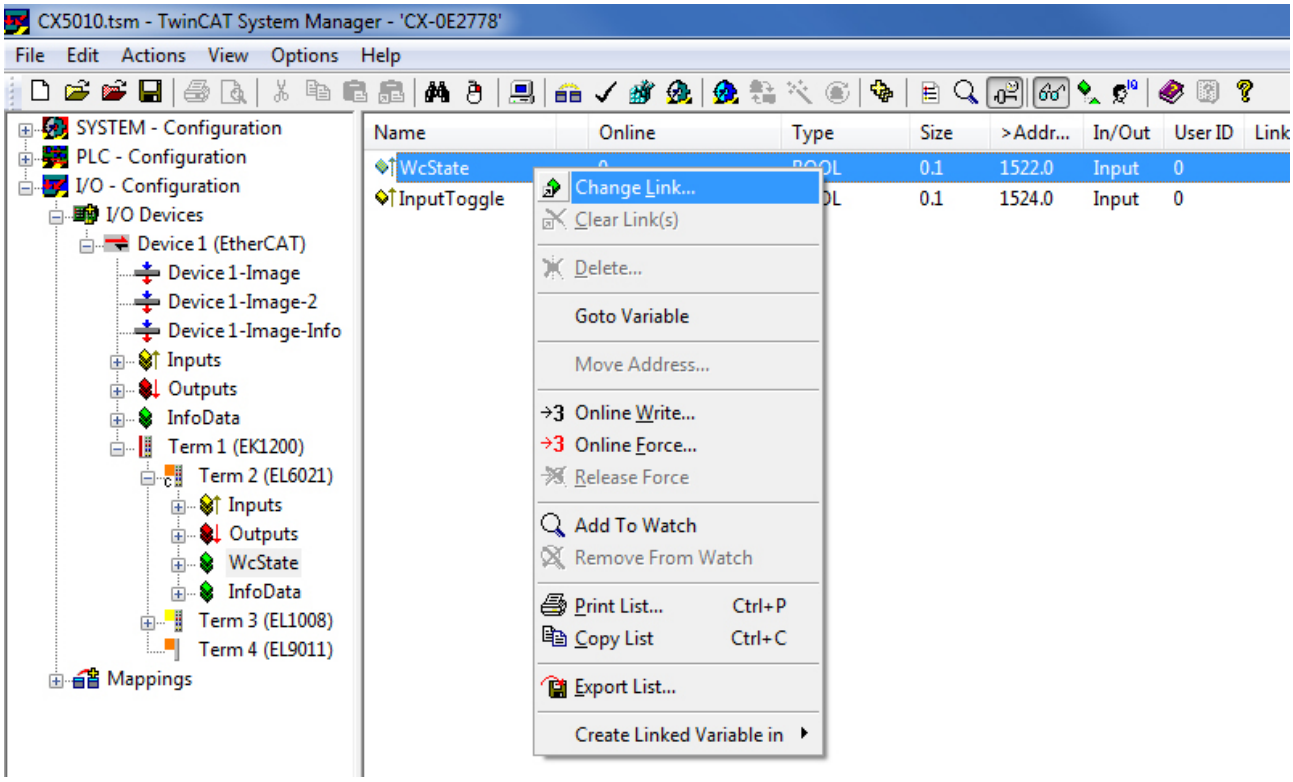

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
Ctrl 0x0000 (0) UINT 2.0 26.0 Output 0
Data Out 0 0x00 (0) USINT 1.0 28.0 Output 0
Data Out 1 0x00 (0) USINT 1.0 29.0 Output 0
Data Out 2 0x00 (0) USINT 1.0 30.0 Output 0
Data Out 3 0x00 (0) USINT 1.0 31.0 Output 0
Data Out 4 0x00 (0) USINT 1.0 32.0 Output 0
Data Out 5 0x00 (0) USINT 1.0 33.0 Output 0
Data Out 6 0x00 (0) USINT 1.0 34.0 Output 0
Data Out 7 0x00 (0) USINT 1.0 35.0 Output 0
Data Out 8 0x00 (0) USINT 1.0 36.0 Output 0
Data Out 9 0x00 (0) USINT 1.0 37.0 Output 0
Data Out 10 0x00 (0) USINT 1.0 38.0 Output 0
Data Out 11 0x00 (0) USINT 1.0 39.0 Output 0
Data Out 12 0x00 (0) USINT 1.0 40.0 Output 0
Data Out 13 0x00 (0) USINT 1.0 41.0 Output 0
Data Out 14 0x00 (0) USINT 1.0 42.0 Output 0
Data Out 15 0x00 (0) USINT 1.0 43.0 Output 0
Data Out 16 0x00 (0) USINT 1.0 44.0 Output 0
Data Out 17 0x00 (0) USINT 1.0 45.0 Output 0
Data Out 18 0x00 (0) USINT 1.0 46.0 Output 0
Data Out 19 0x00 (0) USINT 1.0 47.0 Output 0
Data Out 20 0x00 (0) USINT 1.0 48.0 Output 0
Data Out 21 0x00 (0) USINT 1.0 49.0 Output 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in

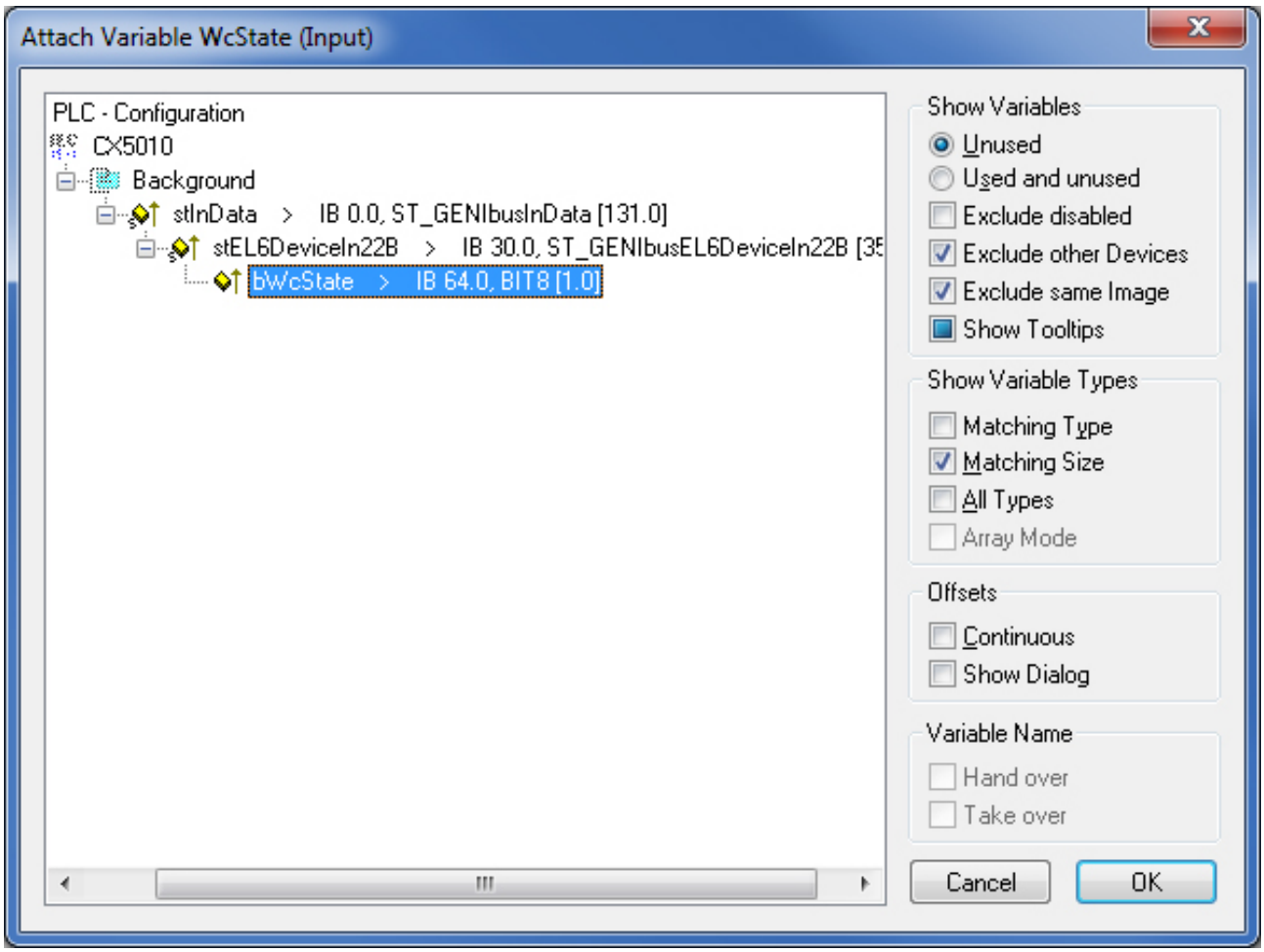

Attach Variable Ctrl (Output)
PLC - Configuration
CX5010
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
wCtrl > QB 6.0, WORD [2.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
wCtrl > QB 30.0, WORD [2.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutI
wSerCtrl > QB 54.0, WORD [2.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK

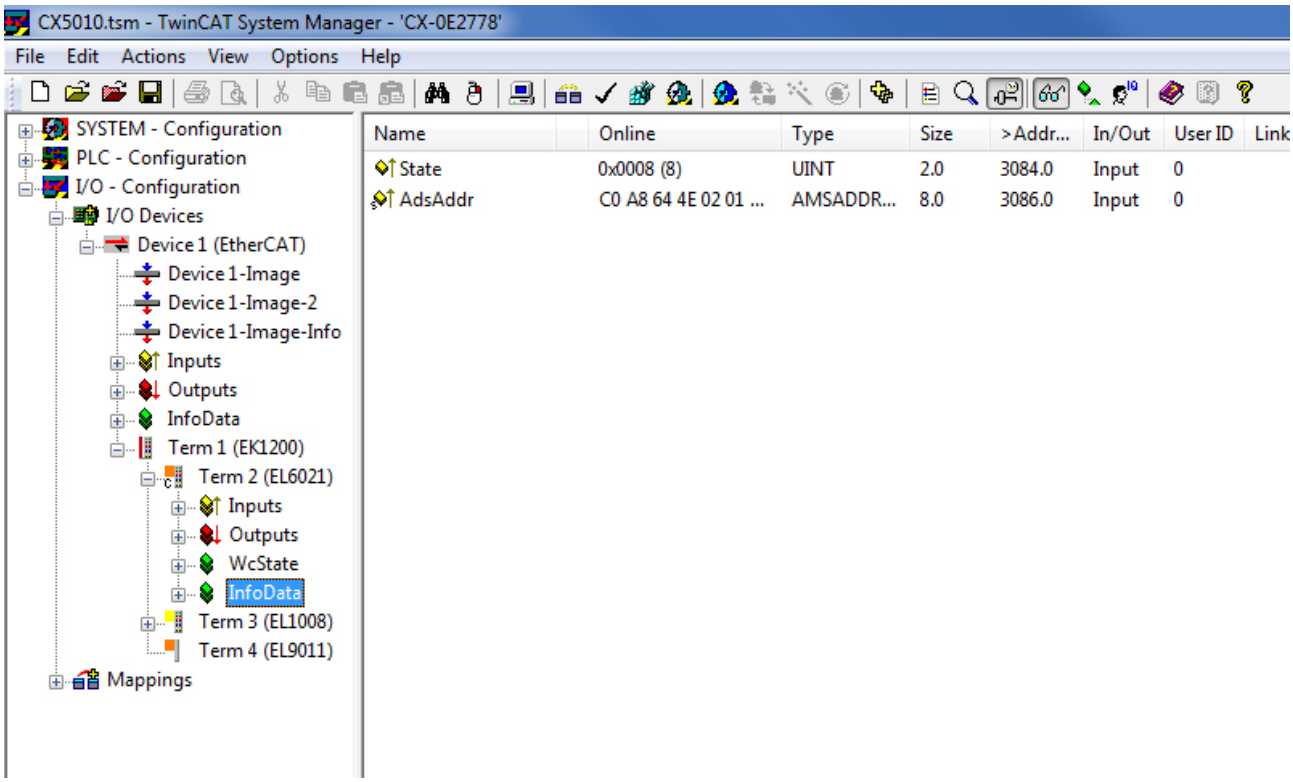


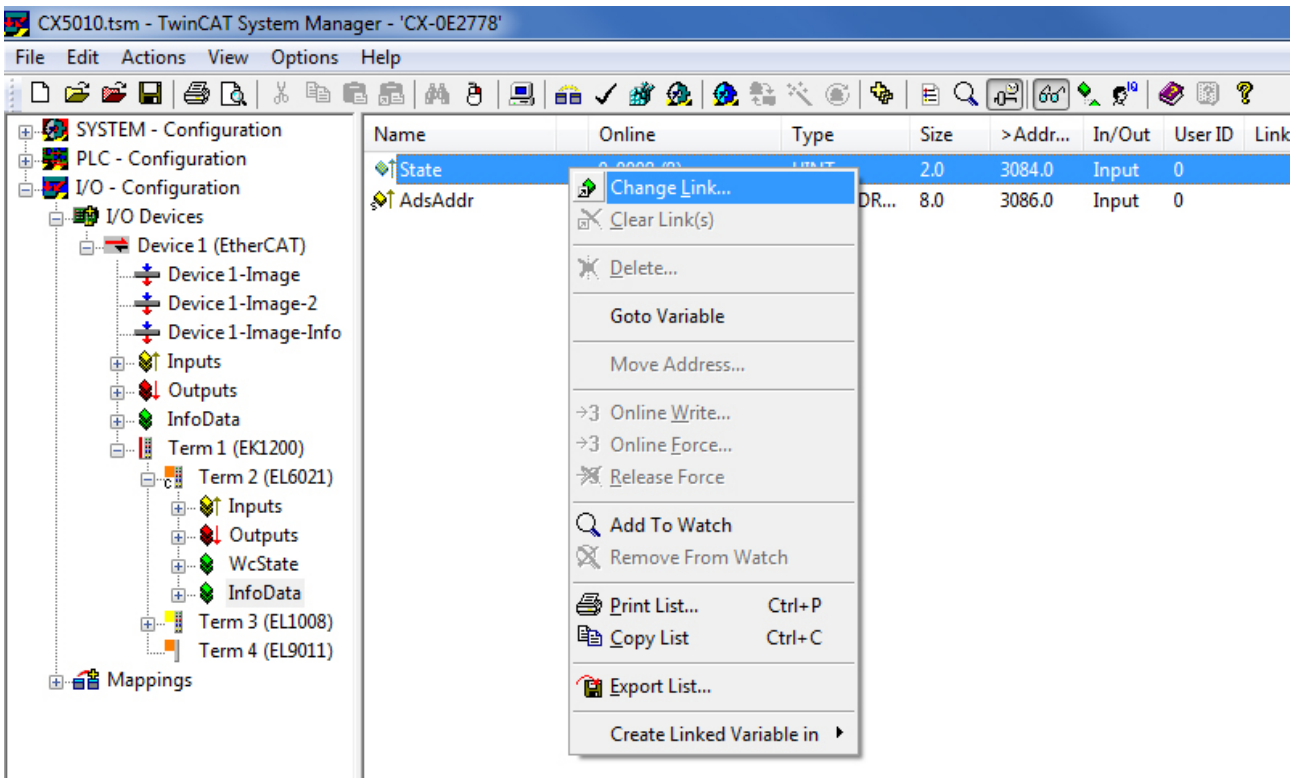
Moreover, the variables of the groups *WcState* and *InfoData* have to be linked:

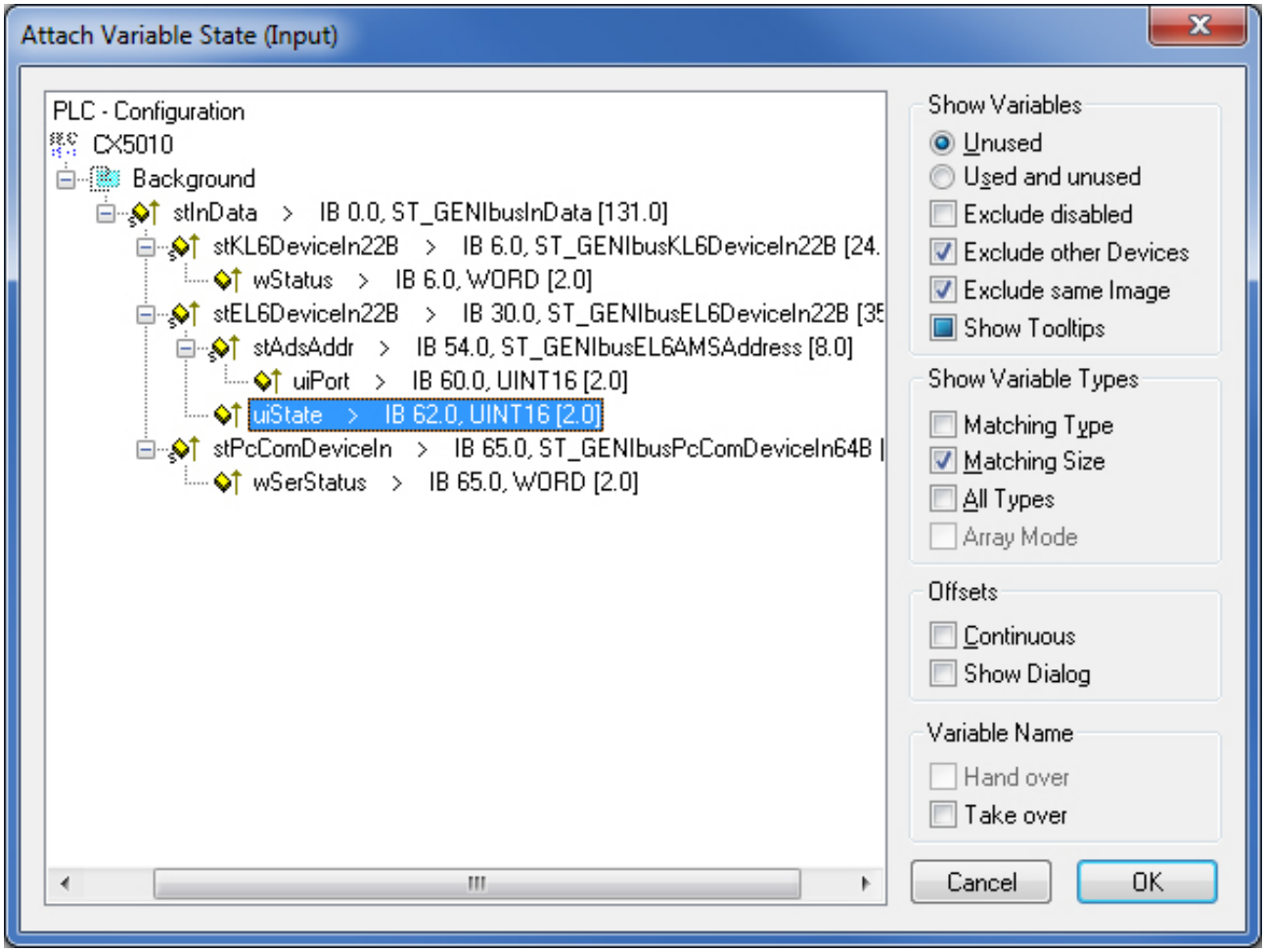

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
WcState 0 BOOL 0.1 1522.0 Input 0
InputToggle 0 BOOL 0.1 1524.0 Input 0

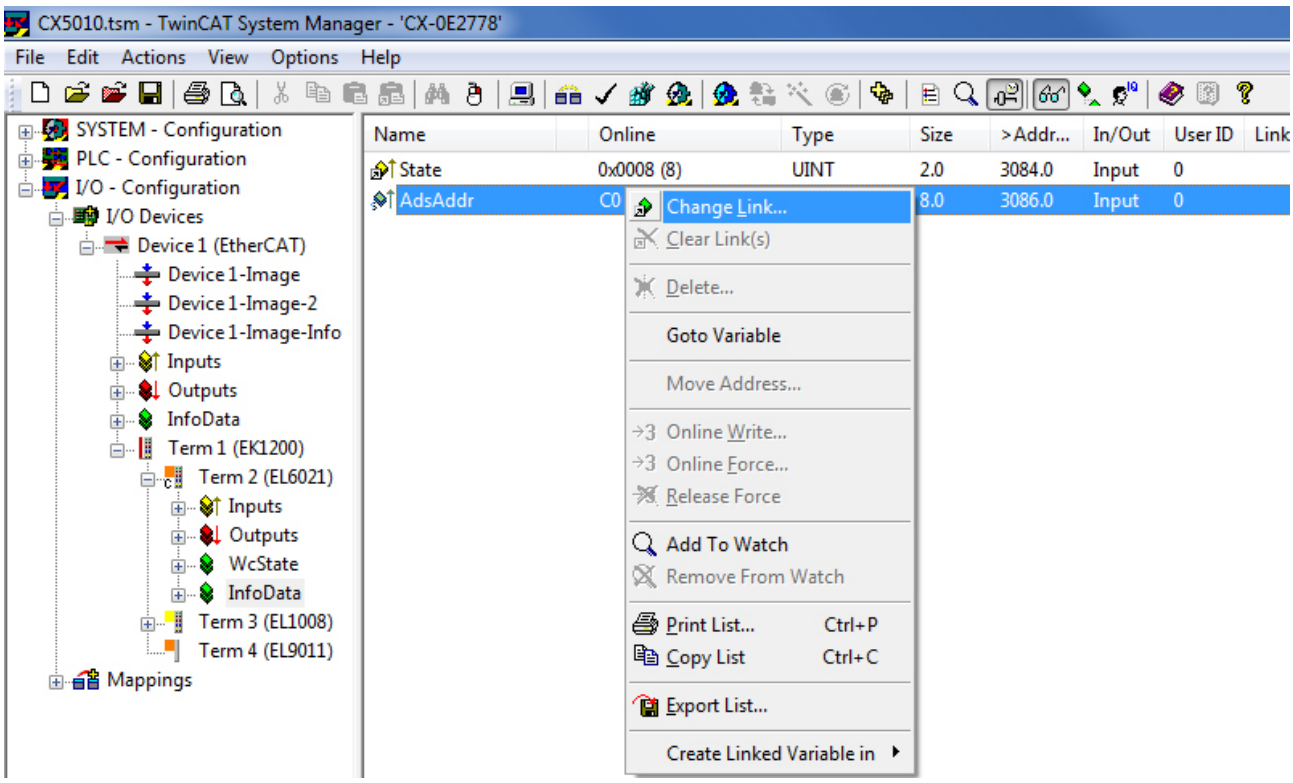

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
WcState 0 BOOL 0.1 1522.0 Input 0
InputToggle 0 BOOL 0.1 1524.0 Input 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in

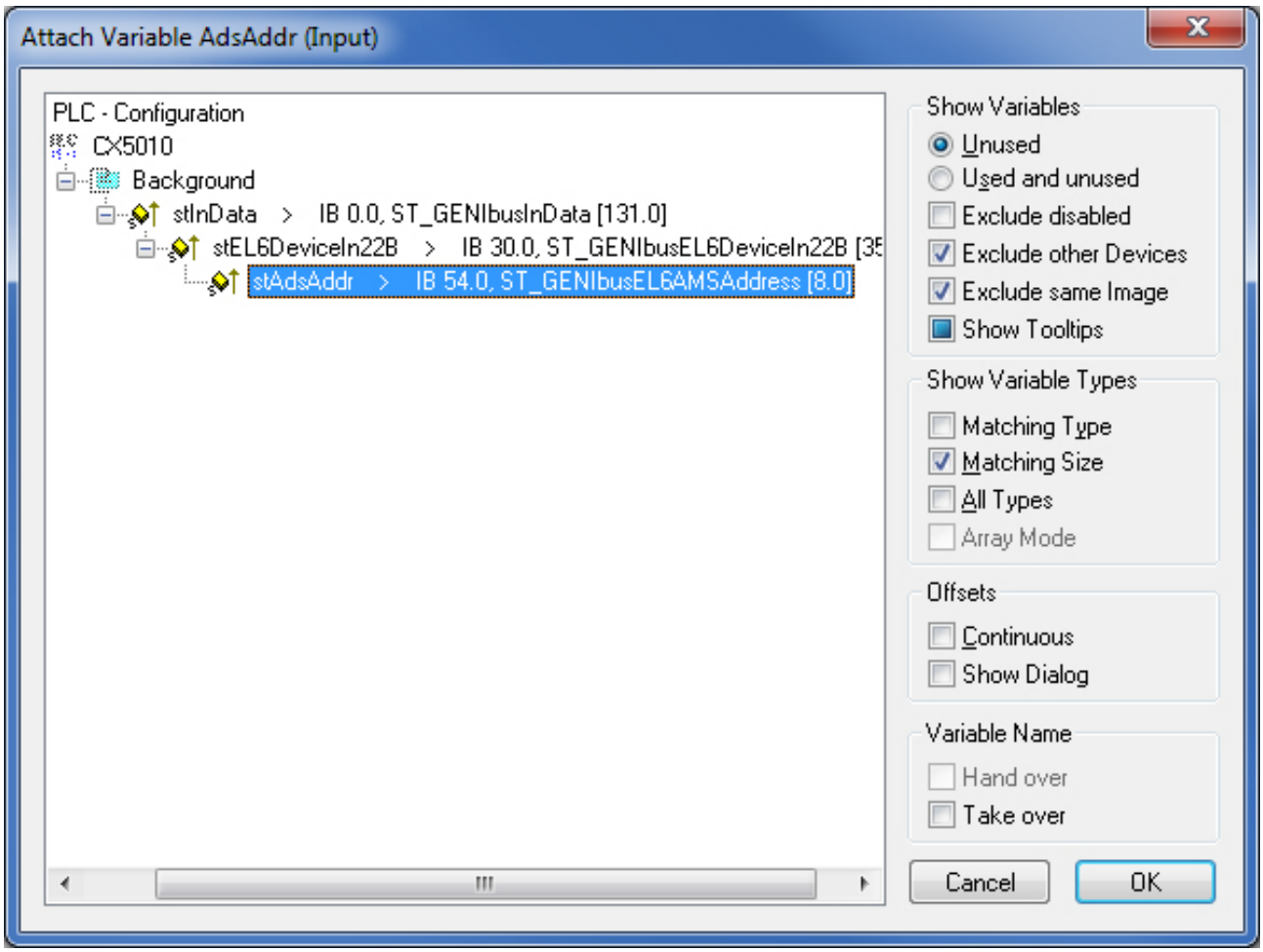

Attach Variable WcState (Input)
PLC - Configuration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
bWcState > IB 64.0, BIT8 [1.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
State 0x0008 (8) UINT 2.0 3084.0 Input 0
AdsAddr C0 A8 64 4E 02 01 ... AMSADDR... 8.0 3086.0 Input 0


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
State 2.0 3084.0 Input 0
AdsAddr 8.0 3086.0 Input 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in


Attach Variable State (Input)
PLC - Configuration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stKL6DeviceIn22B > IB 6.0, ST_GENIbusKL6DeviceIn22B [24.
wStatus > IB 6.0, WORD [2.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
stAdsAddr > IB 54.0, ST_GENIbusEL6AMSAddress [8.0]
uiPort > IB 60.0, UINT16 [2.0]
uiState > IB 62.0, UINT16 [2.0]
stPcComDeviceIn > IB 65.0, ST_GENIbusPcComDeviceIn64B
wSerStatus > IB 65.0, WORD [2.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
File Edit Actions View Options Help
SYSTEM - Configuration
PLC - Configuration
I/O - Configuration
I/O Devices
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Mappings
Name Online Type Size >Addr... In/Out User ID Link
State 0x0008 (8) UINT 2.0 3084.0 Input 0
AdsAddr C0 8.0 3086.0 Input 0
Change Link...
Clear Link(s)
Delete...
Goto Variable
Move Address...
Online Write...
Online Force...
Release Force
Add To Watch
Remove From Watch
Print List... Ctrl+P
Copy List Ctrl+C
Export List...
Create Linked Variable in


Attach Variable AdsAddr (Input)
PLC - Configuration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
stAdsAddr > IB 54.0, ST_GENIbusEL6AMSAddress [8.0]
Show Variables
Unused
Used and unused
Exclude disabled
Exclude other Devices
Exclude same Image
Show Tooltips
Show Variable Types
Matching Type
Matching Size
All Types
Array Mode
Offsets
Continuous
Show Dialog
Variable Name
Hand over
Take over
Cancel
OK

5 Programming

5.1 General Information

Installation

The “TcGENIbus.lib” library is installed as standard from TwinCAT 2.11 build 2253 (R3 and x64 Engineering).

Further libraries required

For PC systems (x86) and Embedded PCs (CXxxxx):

- Standard.lib
- TcBase.lib
- TcSystem.lib
- TcUtilities.lib
- TcEtherCAT.lib

Memory usage

Some of the PLC program memory is already used up by integrating the library. Depending on the application program it is therefore possible that the remaining memory space may be insufficient.

5.2 Function blocks

Base commands

Name	Description
FB_GENIbusCommunication [▶ 46]	Sequentially reads the GENIbus commands from these three buffers and forwards the GENIbus commands to the serial interface.

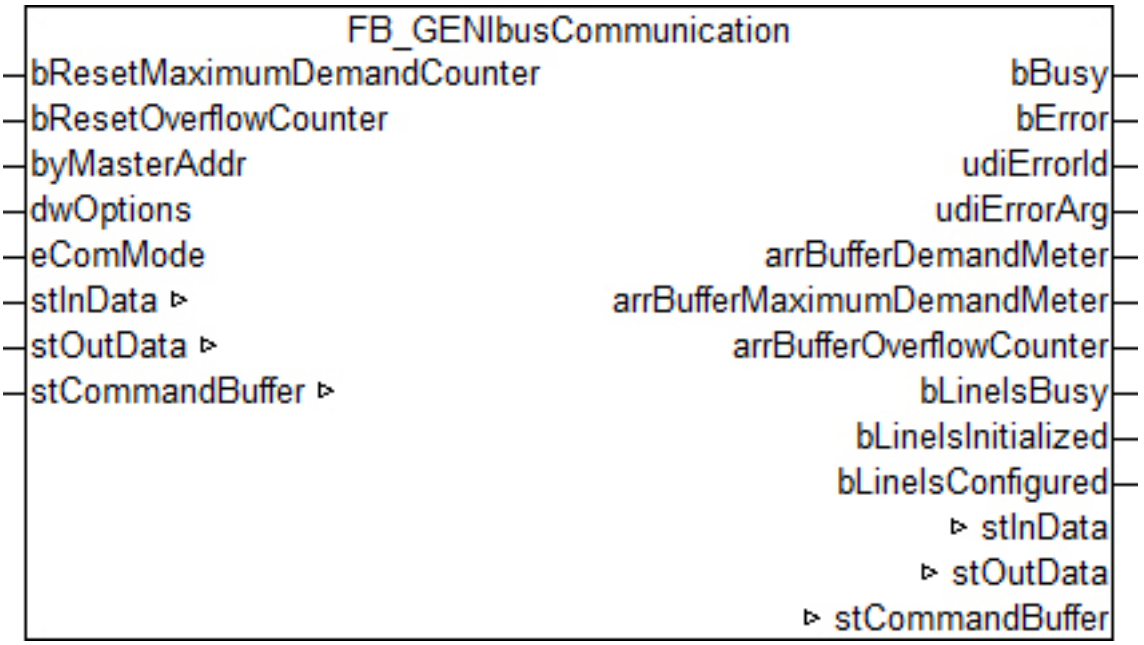
Basic commands

Name	Description
FB_GENIbusGet [▶ 48]	Reads a value from a GENIbus device.
FB_GENIbusSet [▶ 49]	Writes a value in a GENIbus device or executes a command (Class-3 IDs).
FB_GENIbusInfo [▶ 50]	Reads out the notification area of an ID.
FB_GENIbusGetMValue [▶ 51]	Reads a measured value from a GENIbus device.

Pumps

Name	Description
FB_GENIbusMagnaPump [▶ 53]	Represents a universal application for a Grundfos Magna pump.

5.2.1 FB_GENIbusCommunication



The function blocks for the GENIbus commands do not directly access the process image of the selected serial interface; instead, they place the individual GENIbus commands into three different buffers. The FB_GENIbusCommunication() function block sequentially reads the GENIbus commands from these three buffers and forwards the GENIbus commands to the serial interface. This prevents multiple function blocks accessing the process image of the serial interface at the same time. Each of these three buffers is processed with a different priority (high, medium or low). The user of the PLC library can use the eCommandPriority [▶ 57] parameter, which is available in most function blocks, to determine the priority with which the FB_GENIbusCommunication() function block processes the respective GENIbus command.

The buffers in which the GENIbus commands are placed are all contained in a variable of the type ST_GENIbusCommandBuffer [▶ 59]. There is one instance of the FB_GENIbusCommunication() function block and one variable of the type ST_GENIbusCommandBuffer [▶ 59] per serial interface. If possible, the FB_GENIbusCommunication() function block should be called in a separate, faster task.

The extent to which the buffers are utilized can be determined from the outputs of the block. Three arrays are output for this in which each element (0, 1 or 2) represents one of the three buffers (high, middle or low). If you detect regular overflow for one of the three buffers, you should consider the following:

- How heavily are the individual PLC tasks utilized? The TwinCAT System Manager offers various appropriate utilities for the analysis.
- Try reducing the cycle time of the task in which the FB_GENIbusCommunication() function block is called. The value should not exceed 6 ms. Ideally it should be 2 ms.
- Check the cycle time of the PLC task in which the blocks for the individual GENIbus commands are called. This value should be between 10 ms and 60 ms.
- If possible avoid polling (regular reading) of values. Only read values when they are actually required.

VAR_INPUT

```
bResetMaximumDemandCounter  : BOOL;
bResetOverflowCounter       : BOOL;
byMasterAddr                : BYTE;
dwOptions                   : DWORD := 0;
eComMode                    : E_GENIbusComMode;
```

bResetMaximumDemandCounter: A rising edge resets the stored value of the maximum command buffer utilization, *arrBufferMaximumDemandMeter* (0 - 100%, see VAR_OUTPUT).

bResetOverflowCounter: A rising edge resets the stored value of the number of command buffer overflows, *arrBufferOverflowCounter* (see VAR_OUTPUT).

byMasterAddr: Specifies the address that the TwinCAT controller should have within the GENIbus line. Possible input range: 0 - 31.

dwOptions: Reserved for future applications.

eComMode: The selection [▶ 57] of the serial communication interface must be entered at this parameter. If a KL terminal or an EtherCAT Terminal is in use, then a configuration of the connection parameters is internally and automatically started:

- Baud Rate: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1

Unfortunately this is not possible for PC-based interfaces; in this case the parameters must be directly entered in the TwinCAT System Manager.

VAR_OUTPUT

```
bBusy                       : BOOL;
bError                      : BOOL;
udiErrorId                  : UDINT;
udiErrorArg                 : UDINT;
arrBufferDemandMeter        : ARRAY[0..2] OF BYTE;
arrBufferMaximumDemandMeter : ARRAY[0..2] OF BYTE;
arrBufferOverflowCounter    : ARRAY[0..2] OF UINT;
bLineIsBusy                 : BOOL;
bLineIsInitialized          : BOOL;
bLineIsConfigured           : BOOL;
```

bBusy: A GENIbus command is converted into a serial telegram and transmitted. This flag is reset again following a successful response or following an abort after an error.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

arrBufferDemandMeter: Occupation of respective buffer (0 - 100%).

arrBufferMaximumDemandMeter: Previous maximum occupancy of the respective buffer (0 - 100%).

arrBufferOverflowCounter: Number of buffer overflows to date.

bLineIsBusy: This output is set if the serial communication is active.

bLineIsInitialized: If the block is being called for the first time (e.g. when the controller is starting up) an initialization process is executed. No GENIbus commands can be processed during this time.

bLineIsConfigured: This output indicates with TRUE that the terminal has been successfully configured with the above serial parameters. This output is automatically set if the interface is a PC interface since the user has to enter the parameters himself in the TwinCAT System Manager.

Since an error may not interrupt the execution of the function block, bError, udiErrorId and udiErrorArg are initially reset in every PLC cycle and then re-evaluated.

VAR_IN_OUT

```
stInData              : ST_GENIbusInData;
stOutData             : ST_GENIbusOutData;
stCommandBuffer       : ST_GENIbusCommandBuffer;
```

stInData: Reference to the structure [▶ 60] containing the input process image for communication with the serial interface.

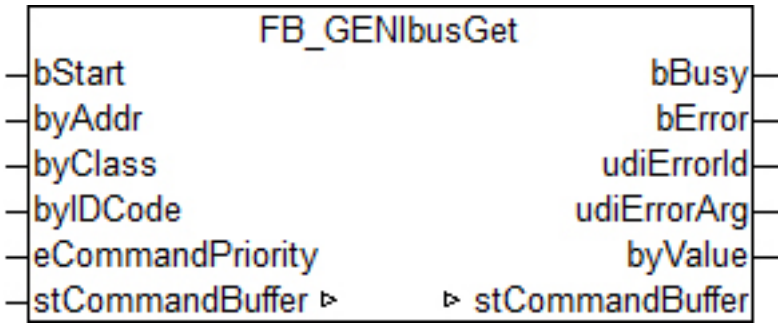
stOutData: Reference to the structure [▶ 62] containing the output process image for communication with the serial interface.

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the GENIbus function blocks.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.2.2 FB_GENIbusGet



This function block reads a value from a GENIbus device.

VAR_INPUT

```
bStart            : BOOL;
byAddr            : BYTE := 0;
byClass           : BYTE := 2;
byIDCode          : BYTE := 0;
eCommandPriority  : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: A rising edge at this input starts the reading process.

byAdress : Address of the GENIbus device to be addressed: valid entries: 1 - 200. This corresponds to the setting as is made directly on the GENIbus device. Conversion to the actual address range 32 – 231 (see GENIbus standard), takes place internally in the function block. A broadcast command via address 255 is naturally not permitted.

byClass/byIDCode: Class and ID code of the memory location to be read. GET commands are permissible only for classes 2, 4, 5 and 7 – an error is output for all other entries. Conversely, there is no restriction of the ID code entry, since these ranges are not without gaps and may possibly be extended.

eCommandPriority: Priority [▶ 57] (high, medium or low) with which the command is processed by the PLC library.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
udiErrorId    : UDINT;
udiErrorArg   : UDINT;
byValue       : BYTE;
```

bBusy: Starting with the edge at *bStart*, this output remains TRUE until the command has been processed.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

byValue: Output of the read value.

VAR_IN_OUT

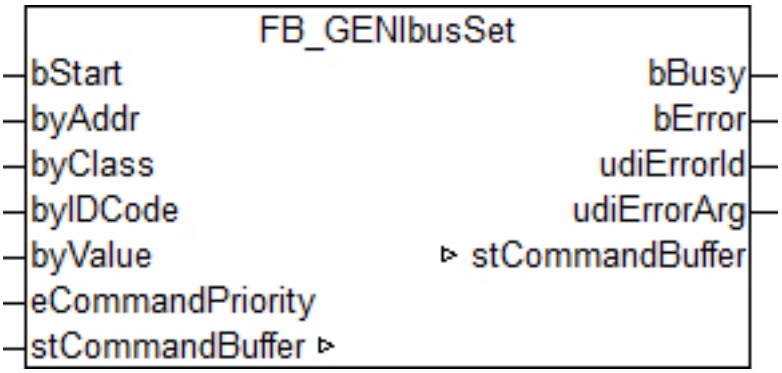
```
stCommandBuffer    : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.2.3 FB_GENIbusSet



This function block writes a value in a GENIbus device or executes a command (Class-3 IDs).

VAR_INPUT

```
bStart             : BOOL;
byAddr             : BYTE := 0;
byClass            : BYTE := 2;
byIDCode           : BYTE := 0;
byValue            : BYTE;
eCommandPriority   : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: A rising edge at this input starts the setting process.

byAdress : address of the GENIbus device to be addressed: valid entries: 1 - 200. This corresponds to the setting as is made directly on the GENIbus device. Conversion to the actual address range 32 – 231 (see GENIbus standard), takes place internally in the function block.

byClass/byIDCode: Class and ID code of the memory location to be written. SET commands are permissible only for classes 3, 4 and 5 – an error is output for all other entries. Conversely, there is no restriction of the ID code entry, since these ranges are not without gaps and may possibly be extended.

byValue: Value to be written. In the case of Class-3 IDs this entry is ignored.

eCommandPriority: Priority [▶ 57] (high, medium or low) with which the command is processed by the PLC library.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
udiErrorId    : UDINT;
udiErrorArg   : UDINT;
```

bBusy: Starting with the edge at *bStart*, this output remains TRUE until the command has been processed.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

VAR_IN_OUT

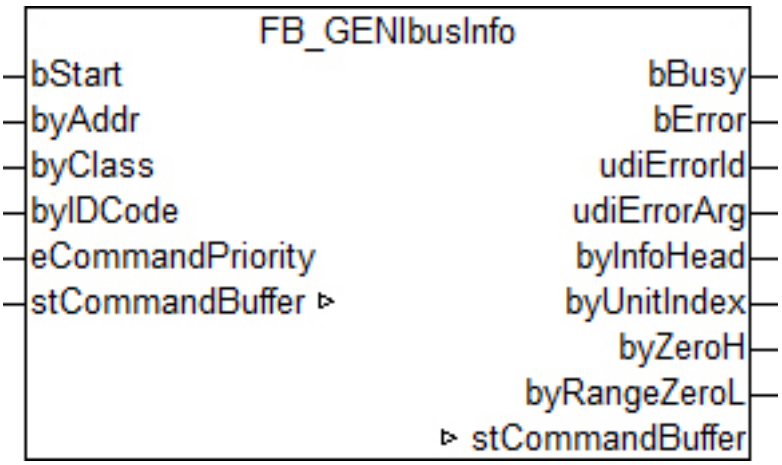
```
stCommandBuffer   : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.2.4 FB_GENIbusInfo



This function reads out the notification area of an ID.

VAR_INPUT

```
bStart            : BOOL;
byAddr            : BYTE := 0;
byClass           : BYTE := 2;
byIDCode          : BYTE := 0;
eCommandPriority  : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: A rising edge at this input starts the reading process.

byAdress : Address of the GENIbus device to be addressed: valid entries: 1 - 200. This corresponds to the setting as is made directly on the GENIbus device. Conversion to the actual address range 32 – 231 (see GENIbus standard), takes place internally in the function block. A broadcast command via address 255 is naturally not permitted.

byClass/byIDCode: Class and ID code of the memory location to be read. INFO commands are permissible only for classes 2, 3, 4 and 5 – an error is output for all other entries. Conversely, there is no restriction of the ID code entry, since these ranges are not without gaps and may possibly be extended.

eCommandPriority: Priority [▶ 57] (high, medium or low) with which the command is processed by the PLC library.

VAR_OUTPUT

```
bBusy          : BOOL;
bError         : BOOL;
udiErrorId     : UDINT;
udiErrorArg    : UDINT;
byInfoHead     : BYTE;
byUnitIndex    : BYTE;
byZeroH        : BYTE;
byRangeZeroL   : BYTE;
```

bBusy: Starting with the edge at *bStart*, this output remains TRUE until the command has been processed.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

byInfoHead: Scaling information

byUnitIndex: Sign and unit – coded.

byZeroH: Zero point in the case of normal range and zero-point scaling OR high-byte zero point in the case of extended scaling.

byRangeZeroL: Range in the case of normal range and zero-point scaling OR low-byte zero point in the case of extended scaling.

VAR_IN_OUT

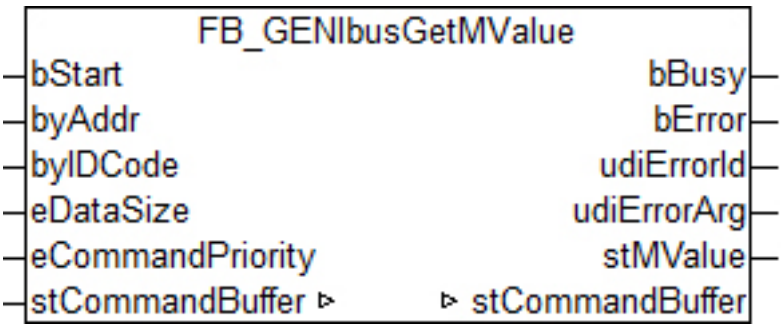
```
stCommandBuffer    : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.2.5 FB_GENIbusGetMValue



This function block reads a measured value from a GENIbus device. The operation is thereby exclusively restricted to values of class 2. Only the ID code of the high byte and the length of the measured value need to be specified; the type of scaling and the unit of the measured value are determined by an internal INFO query. A structure at the *stMValue* output provides all important information about the value.

VAR_INPUT

```
bStart            : BOOL;
byAddr            : BYTE := 0;
byIDCode          : BYTE := 0;
eDataSize         : E_GENIBusMDataSize;
eCommandPriority  : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: A rising edge at this input starts the reading process.

byAddr : Address of the GENIbus device to be addressed: valid entries: 1 - 200. This corresponds to the setting as is made directly on the GENIbus device. Conversion to the actual address range 32 – 231 (see GENIbus standard), takes place internally in the function block. A broadcast command via address 255 is naturally not permitted.

byIDCode: ID code of the value to be read. In the case of 16, 24 and 32-bit values, the ID of the high byte must be specified here, and the following order is always assumed: ID = hi, ID+1 = lo1, ID+2 = lo2, ID+3 = lo3.

eDataSize: Data size [▶ 57] of the measured value: 8, 16, 24 or 32 bytes.

eCommandPriority: Priority [▶ 57] (high, medium or low) with which the command is processed by the PLC library.

Example: Read-out of the total pumped volume of water. For this case is:

- byIdCode = 121
- eDataSize = eGENIbusMSize32Bit

temp_in_3 '	2, 118			R	Temperature input 3 (113) value
t_bear_de	2, 119		5	R	Motor bearing temperature Drive End (DE)
t_bear_nde	2, 120		5	R	Motor bearing temp. None Drive End (NDE)
volume_hi	2, 121	1 m^3	5	R	Pumped volume (accumulated value of actual pump flow). Reset by command **RESET_HIST**
volume_lo1	2, 122				
volume_lo2	2, 123				
volume_lo3	2, 124				
spec_energy_hi	2, 125	1 Wh/m^3	5	R	Specific energy consumption
spec_energy_lo	2, 126		5		
grf_sensor_press	2, 127	INFO		R	Grundfos sensor pressure measurement GSP
grf_sensor_temp	2, 128	INFO		R	Grundfos sensor temperature measurement GST

Source: Grundfos documentation "Operating the MAGNA3 and MGE model H/I via the GENIpro interface - Edition 01.00.35 - April 2015".

VAR_OUTPUT

```
bBusy          : BOOL;
bError         : BOOL;
udiErrorId     : UDINT;
udiErrorArg    : UDINT;
stMValue       : ST_GENIbusMValue;
```

bBusy: Starting with the edge at *bStart*, this output remains TRUE until the command has been processed.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

stMValue : Output [▶ 62] of the read value.

VAR_IN_OUT

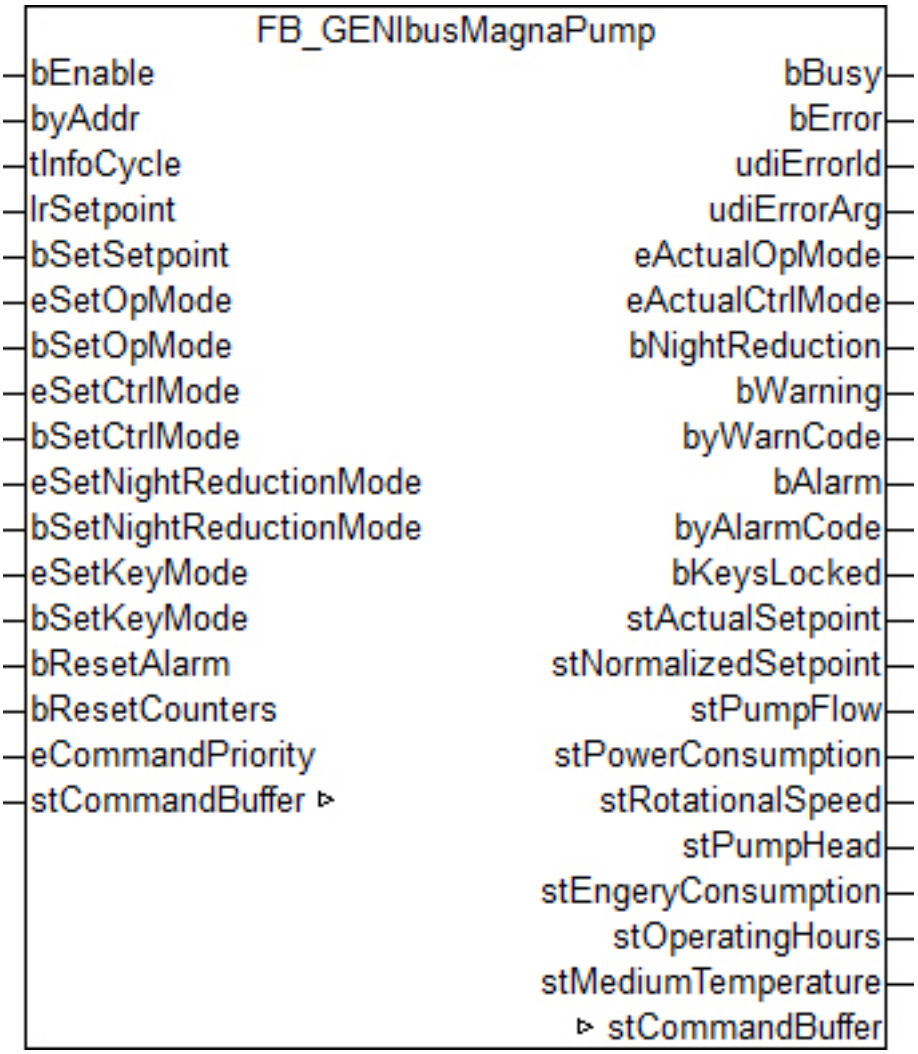
```
stCommandBuffer    : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.2.6 FB_GENIbusMagnaPump



This function block represents a universal application for a Grundfos Magna pump. The fundamental operating modes can be set and important parameters read out.

The values shown in bold lettering inside square brackets represent the class and ID with which the commands are executed, or the information acquired. These values are listed in the Grundfos documentation “Operating the MAGNA3 and MGE model H/I via the GENIpro interface - Edition 01.00.35 - April 2015”.

VAR_INPUT

```
bEnable                 : BOOL;
byAddr                  : BYTE := 0;
tInfoCycle              : TIME := t#5s;
lrSetpoint              : LREAL;
bSetSetpoint            : BOOL;
eSetOpMode              : E_GENIbusOpMode := eGENIbusOpModeStop;
bSetOpMode              : BOOL;
eSetCtrlMode            : E_GENIbusCtrlMode := eGENIbusCtrlModeConstFreq;
bSetCtrlMode            : BOOL;
eSetNightReductionMode  : E_GENIbusNightReductionMode := eGENIbusNightReductionModeOff;
bSetNightReductionMode  : BOOL;
eSetKeyMode             : E_GENIbusKeyMode;
bSetKeyMode             : BOOL;
bResetAlarm             : BOOL;
bResetCounters          : BOOL;
eCommandPriority        : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bEnable: The function block is activated by setting this input.

byAdress : Address of the GENIbus device to be addressed: valid entries: 1 - 200. This corresponds to the setting as is made directly on the GENIbus device. Conversion to the actual address range 32 – 231 (see GENIbus standard), takes place internally in the function block.

A broadcast (collective) command to several pumps is also possible. The value at this input must then be 255. The value queries are deactivated in the case of the broadcast command.

tInfoCycle: Specifies the interval at which the value-query commands are to be output. This entry is limited to a minimum of 1 s. Conversely, the entry “0s” is permitted and means that no query takes place.

lrSetpoint: Setpoint entry **[5, 1]**. The entry is in percent and refers to the specified limits, depending on the method of control. A more precise description is given in the respective documentation from the Grundfos company.

bSetSetpoint: A rising edge at this input transmits the set setpoint.

eSetOpMode: This input is used to set one of the following operating modes [▶ 58]:

- Stop **[3, 5]**
- Start **[3, 6]**
- Minimum curve **[3, 25]**
- Maximum curve **[3, 26]**

bSetOpMode: A rising edge at this input transmits the set operating mode.

eSetCtrlMode: This input is used to set one of the following control modes [▶ 57]:

- Constant frequency **[3, 22]**
- Proportional pressure **[3, 23]**
- Constant pressure **[3, 24]**
- Auto-adapting **[3, 52]**

bSetCtrlMode: A rising edge at this input transmits the set control mode.

eSetNightReductionMode: This input is used to select or deselect the night setback mode [▶ 58]. **[4, 170]**

bSetNightReductionMode: A rising edge at this input transmits the set selection [▶ 58].

eSetKeyMode: Locking of the manual operation [▶ 57] on the pump can be selected with the aid of this input. The lock only blocks the parameterization menu, not the keys themselves. **[3, 30/31]**

bSetKeyMode: A rising edge at this input transmits the set selection.

bResetAlarm: A rising edge at this input resets the currently pending alarm on the device. **[3, 2]**

bResetCounters: A rising edge at this input resets counters, such as operating hours or energy. **[3, 36]**

eCommandPriority: Priority [▶ 57] (high, medium or low) with which the command is processed by the PLC library.

VAR_OUTPUT

```
bBusy                : BOOL;
bError               : BOOL;
udiErrorId           : UDINT;
udiErrorArg          : UDINT;
eActualOpMode        : E_GENIbusActOpMode;
eActualCtrlMode      : E_GENIbusActCtrlMode;
bNightReduction      : BOOL;
bWarning             : BOOL;
byWarnCode           : BYTE;
bAlarm               : BOOL;
byAlarmCode          : BYTE;
bKeysLocked          : BOOL;
stActualSetpoint     : ST_GENIbusMValue;
stNormalizedSetpoint : ST_GENIbusMValue;
stPumpFlow           : ST_GENIbusMValue;
stPowerConsumption   : ST_GENIbusMValue;
stRotationalSpeed    : ST_GENIbusMValue;
stPumpHead           : ST_GENIbusMValue;
stEngeryConsumption  : ST_GENIbusMValue;
stOperatingHours     : ST_GENIbusMValue;
stMediumTemperature  : ST_GENIbusMValue;
```

bBusy: This output is always TRUE when a command or query is being processed.

bError: This output is switched to TRUE as soon as an error occurs during the execution of a command. The command-specific error code is contained in *udiErrorId*.

udiErrorId: Contains the command-specific error code of the most recently executed command. It is set back to 0 by the reactivation of the function block via the *bStart* input. See Error codes [▶ 65].

udiErrorArg: If applicable, contains an extended description of the error code.

eActualOpMode: Currently valid operating mode [▶ 56]. **[2, 81]**

eActualCtrlMode: Currently valid control mode [▶ 56]. **[2, 81]**

bNightReduction: Night setback is selected. **[2, 84]**

bWarning: A warning message is pending.

byWarnCode: Code of the current warning message. **[2, 156]**

bAlarm: An alarm is pending.

byAlarmCode: Code of the current alarm. **[2, 158]**

bKeysLocked: Locking of manual operation on the pump is activated. **[4, 170]**

stActualSetpoint: Currently set setpoint [▶ 62]; the unit displayed depends on the control mode. **[2, 48]**

stNormalizedSetpoint: Currently normalized setpoint. **[2, 49]**

stPumpFlow: Flow rate. **[2, 39]**

stPowerConsumption: Power consumption. **[2, 34]**

stRotationalSpeed: Speed. **[2, 35/36]**

stPumpHead: Pump head. **[2, 37]**

stEngeryConsumption: Energy consumption. **[2, 152/153]**

stOperatingHours: Operating hour meter. **[2, 24/25]**

stMediumTemperature: Pumped water (medium) temperature. **[2, 58]**

Since an error may not interrupt the execution of the function block, bError, udiErrorId and udiErrorArg are initially reset in every PLC cycle and then re-evaluated. For the determination of sporadically occurring errors, an error memory must therefore be programmed external to the function block.

VAR_IN_OUT

```
stCommandBuffer       : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Reference to the structure [▶ 59] for communication (buffer) with the FB_GENIbusCommunication [▶ 46]() function block.

Requirements

Development environment	Target platform	Required libraries
TwinCAT v2.11 R3/x64 from Build 2253	PC / CX	TcGENIbus library from V1.0.0

5.3 Data types

5.3.1 E_GENIbusACK

ACK (Acknowledge-Code) from the response telegram.

```
TYPE E_GENIbusACK :
(
  eGENIbusACKOk           := 0,
  eGENIbusACKUnknownClass := 1,
  eGENIbusACKUnknownId    := 2,
  eGENIbusACKIllegalOp    := 3
);
END_TYPE
```

5.3.2 E_GENIbusActCtrlMode

The read-out current control mode.

```
TYPE E_GENIbusActCtrlMode :
(
  eGENIbusActCtrlModeUnknown    := 0,
  eGENIbusActCtrlModeConstFreq  := 1,
  eGENIbusActCtrlModeConstPress := 2,
  eGENIbusActCtrlModePropPress  := 3,
  eGENIbusActCtrlModeAutoAdapt  := 4
);
END_TYPE
```

5.3.3 E_GENIbusActOpMode

The read-out current operating mode.

```
TYPE E_GENIbusActOpMode :
(
  eGENIbusActOpModeUnknown := 0,
  eGENIbusActOpModeStop    := 1,
  eGENIbusActOpModeStart   := 2,
  eGENIbusActOpModeMin     := 3,
  eGENIbusActOpModeMax     := 4,
  eGENIbusActOpModeHand    := 5,
  eGENIbusActOpUserDef     := 6
);
END_TYPE
```

5.3.4 E_GENIbusAddrType

Addressing type.

```
TYPE E_GENIbusAddrType :
(
  eGENIbusAddrTypeSingle    := 0,
  eGENIbusAddrTypeMulti     := 1,
  eGENIbusAddrTypeBroadcast := 2
);
END_TYPE
```

5.3.5 E_GENIbusCommandPriority

Command priority.

```
TYPE E_GENIbusCommandPriority :
(
  eGENIbusCommandPriorityHigh   := 0,
  eGENIbusCommandPriorityMiddle := 1,
  eGENIbusCommandPriorityLow    := 2
);
END_TYPE
```

5.3.6 E_GENIbusComMode

Selection of the serial communication interface.

```
TYPE E_GENIbusComMode :
(
  eGENIbusComMode_Unknown := 0,
  eGENIbusComMode_KL6_5B  := 1,
  eGENIbusComMode_KL6_22B := 2,
  eGENIbusComMode_EL6_22B := 3,
  eGENIbusComMode_PC_64B  := 4
);
END_TYPE
```

5.3.7 E_GENIbusCtrlMode

Adjustable control modes.

```
TYPE E_GENIbusCtrlMode :
(
  eGENIbusCtrlModeUnknown    := 0,
  eGENIbusCtrlModeConstFreq  := 1,
  eGENIbusCtrlModeConstPress := 2,
  eGENIbusCtrlModePropPress  := 3,
  eGENIbusCtrlModeAutoAdapt  := 4
);
END_TYPE
```

5.3.8 E_GENIbusKeyMode

Disablement of the parameterization option on the GENIbus device.

```
TYPE E_GENIbusKeyMode :
(
  eGENIbusKeyModeLocked   := 0,
  eGENIbusKeyModeUnlocked := 1
);
END_TYPE
```

5.3.9 E_GENIBusMDataSize

Bit size of the value to be read from the GENIbus device.

```
TYPE E_GENIBusMDataSize :
(
  eGENIbusMSize8Bit  := 0,
  eGENIbusMSize16Bit := 1,
  eGENIbusMSize24Bit := 2,
  eGENIbusMSize32Bit := 3
);
END_TYPE
```

5.3.10 E_GENIbusNightReductionMode

Night setback mode on/off.

```
TYPE E_GENIbusNightReductionMode :
(
  eGENIbusNightReductionModeOff := 0,
  eGENIbusNightReductionModeOn  := 1
);
END_TYPE
```

5.3.11 E_GENIbusOpMode

Adjustable control modes.

```
TYPE E_GENIbusOpMode :
(
  eGENIbusOpModeUnknown := 0,
  eGENIbusOpModeStop    := 1,
  eGENIbusOpModeStart   := 2,
  eGENIbusOpModeMin     := 3,
  eGENIbusOpModeMax     := 4
);
END_TYPE
```

5.3.12 E_GENIbusOS

OS (Operation-Specifier) in the command telegram.

```
TYPE E_GENIbusOS :
(
  eGENIbusGET  := 0,
  eGENIbusSET  := 1,
  eGENIbusINFO := 2
);
END_TYPE
```

5.3.13 E_GENIbusSD

SD (Start Delimiter) in the command or response telegram.

```
TYPE E_GENIbusSD :
(
  eGENIbusNull        := 16#0,
  eGENIbusDatareply   := 16#24,
  eGENIbusDatamessage := 16#26,
  eGENIbusDatarequest := 16#27
);
END_TYPE
```

5.3.14 E_GENIbusSIF

SIF (Scale Information Format) in the response telegram.

```
TYPE E_GENIbusSIF :
(
eGENIbusNoScaleInfo     := 0,
  eGENIbusBitWiseScaled := 1,
  eGENIbusScaled816     := 2,
```

```
  eGENIbusScaledExt     := 3
);
END_TYPE
```

5.3.15 ST_GENIbusCommandBuffer

Global command buffer for commands and their responses.

```
TYPE ST_GENIbusCommandBuffer :
STRUCT
  arrMessageQueue  : ARRAY[0..2] OF ST_GENIbusMessageQueue;
  stResponseTable  : ST_GENIbusResponseTable;
  udiMessageHandle : UDINT;
END_STRUCT
END_TYPE
```

arrMessageQueue: Input buffer [▶ 61] for the commands. Through the field declaration there is a choice of 3 different buffers: for high, medium, and low priority.

stResponseTable: Buffer [▶ 64] for the command response.

udiMessageHandle: Pointer to the current buffer element.

5.3.16 ST_GENIbusComRegisterData

Register address and contents for the parameterization of terminals.

```
TYPE ST_GENIbusComRegisterData :
STRUCT
  byRegister : BYTE;
  wValue     : WORD;
END_STRUCT
END_TYPE
```

byRegister: Register address.

wValue: Register contents.

5.3.17 ST_GENIbusEL6AMSAddress

Structure for linking in the input process image; should be used for the communication of an EL6xxx terminal.

```
TYPE ST_GENIbusEL6AMSAddress :
STRUCT
  arrNetId : ARRAY[0..5] OF USINT;
  uiPort   : UINT;
END_STRUCT
END_TYPE
```

5.3.18 ST_GENIbusEL6DeviceIn22B

Structure for linking in the input process image; must be used for the communication of an EL6xxx terminal.

```
TYPE ST_GENIbusEL6DeviceIn22B :
STRUCT
  wStatus   : WORD;
  arrData   : ARRAY[0..21] OF BYTE;
  stAdsAddr : ST_GENIbusEL6AMSAddress;
  uiState   : UINT;
  bWcState  : BOOL;
END_STRUCT
END_TYPE
```

Also see about this

- ST_GENIbusEL6AMSAddress [▶ 59]

5.3.19 ST_GENIbusEL6DeviceOut22B

Structure for linking in the output process image; must be used for the communication of an EL6xxx terminal.

```
TYPE ST_GENIbusEL6DeviceOut22B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.20 ST_GENIbusInData

Structure for linking the input image of the process variables. There is a choice of four different possible structures, of which finally only one is to be linked:

```
TYPE ST_GENIbusInData :
STRUCT
  stKL6DeviceIn5B  : ST_GENIbusKL6DeviceIn5B;
  stKL6DeviceIn22B : ST_GENIbusKL6DeviceIn22B;
  stEL6DeviceIn22B : ST_GENIbusEL6DeviceIn22B;
  stPcComDeviceIn  : ST_GENIbusPcComDeviceIn64B;
END_STRUCT
END_TYPE
```

stKL6DeviceIn5B: Input process [▶ 60] image of a 5-byte data terminal with standard communication bus, e.g. KL6021.

stKL6DeviceIn22B: Input process [▶ 60] image of a 22-byte data terminal with standard communication bus, e.g. KL6041.

stEL6DeviceIn22B: Input process [▶ 59] image of a 22-byte EtherCAT data terminal, e.g. EL6021.

stPcComDeviceIn: Input process [▶ 63] image of a serial PC interface.

5.3.21 ST_GENIbusKL6DeviceIn22B

Structure for linking in the input process image; must be used for the communication of an EL6xxx terminal with 22-byte process image.

```
TYPE ST_GENIbusKL6DeviceIn22B :
STRUCT
  wStatus : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.22 ST_GENIbusKL6DeviceIn5B

Structure for linking in the input process image; must be used for the communication of an EL6xxx terminal with 5-byte process image.

```
TYPE ST_GENIbusKL6DeviceIn5B :
STRUCT
  byStatus : BYTE;
  arrData  : ARRAY[0..4] OF BYTE;
END_TYPE
```

5.3.23 ST_GENIbusKL6DeviceOut22B

Structure for linking in the output process image; must be used for the communication of an EL6xxx terminal with 22-byte process image.

```
TYPE ST_GENIbusKL6DeviceOut22B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.24 ST_GENIbusKL6DeviceOut5B

Structure for linking in the output process image; must be used for the communication of an EL6xxx terminal with 5-byte process image.

```
TYPE ST_GENIbusKL6DeviceOut5B :
STRUCT
  byCtrl  : BYTE;
  arrData : ARRAY[0..4] OF BYTE;
END_TYPE
```

5.3.25 ST_GENIbusMessageQueue

Command buffer.

```
TYPE ST_GENIbusMessageQueue :
STRUCT
  arrBuffer                   : ARRAY[1..GENIBUS_COMMAND_BUFFER_ENTRIES] OF ST_GENIbusMessageQueueI
tem;
  byBufferReadPointer         : BYTE;
  byBufferWritePointer        : BYTE;
  byBufferDemandCounter       : BYTE;
  byBufferMaximumDemandCounter : BYTE;
  uiBufferOverflowCounter     : UINT;
  bLockSemaphore              : BOOL;
END_STRUCT
END_TYPE
```

arrBuffer: Command buffer.

byBufferReadPointer: Pointer to the current element of the command buffer.

byBufferWritePointer: Pointer to the current element of the receive buffer.

byBufferDemandCounter: Current buffer demand.

byBufferMaximumDemandCounter: Maximum buffer demand.

uiBufferOverflowCounter: Number of buffer overflows.

bLockSemaphore: Write protection during the processing of a command.

Also see about this

- ST_GENIbusMessageQueueItem [▶ 61]

5.3.26 ST_GENIbusMessageQueueItem

Individual element in the command buffer.

```
TYPE ST_GENIbusMessageQueueItem :
STRUCT
  byAddr           : BYTE;
  eAddrType        : E_GENIbusAddrType;
  eSD              : E_GENIbusSD;
  arrAPDUs         : ARRAY[1..GENIBUS_MAX_APDU_NUMBER] OF ST_GENIbusRequestClassEntry;
  byRFS            : BYTE;
  udiMessageHandle : UDINT;
END_STRUCT
END_TYPE
```

byAddr: Device destination address.

eAddrType: Single, multiple, or collective command [▶ 57].

eSD: Start delimiter [▶ 58] of the telegram.

arrAPDUs: Collection of the APDU [▶ 64]s (Application Program Data Units) to be transmitted.

byRFS: Not yet used: "Request from Slave".

udiMessageHandle: Pointer to the current buffer element.

5.3.27 ST_GENIbusMValue

Structure with the contents of a read device value, e.g. flow rate or speed.

```
TYPE ST_GENIbusMValue :
STRUCT
  lrValue      : LREAL;
  lrPrefix     : LREAL;
  sUnit        : STRING(8);
  eDataSize    : E_GENIBusMDataSize;
  byValueH     : BYTE;
  byValueL1    : BYTE;
  byValueL2    : BYTE;
  byValueL3    : BYTE;
  byInfoHead   : BYTE;
  byUnitIndex  : BYTE;
  byZeroH      : BYTE;
  byRangeZeroL : BYTE;
END_STRUCT
END_TYPE
```

lrValue: Final value determined from the raw data.

lrPrefix: Sign and division (+/- and e.g. 0.1).

sUnit: Unit.

eDataSize: Size [▶ 57] of the measured value (8, 16, 24 or 32 bytes).

byValueH: High byte of the measured value.

byValueL1: Low byte.

byValueL2: Low byte.

byValueL3: Low byte.

byInfoHead: Scaling information

byUnitIndex: Sign and unit – coded.

byZeroH: Zero point in the case of normal range and zero-point scaling OR high-byte zero point in the case of extended scaling.

byRangeZeroL: Range in the case of normal range and zero-point scaling OR low-byte zero point in the case of extended scaling.

5.3.28 ST_GENIbusOutData

Structure for linking the output image of the process variables. There is a choice of four different possible structures, of which finally only one is to be linked:

```
TYPE ST_GENIbusOutData :
STRUCT
  stKL6DeviceOut5B  : ST_GENIbusKL6DeviceOut5B;
  stKL6DeviceOut22B : ST_GENIbusKL6DeviceOut22B;
  stEL6DeviceOut22B : ST_GENIbusEL6DeviceOut22B;
  stPcComDeviceOut  : ST_GENIbusPcComDeviceOut64B;
END_STRUCT
END_TYPE
```

stKL6DeviceOut5B: Output process image of a 5-byte data terminal [▶ 61] with standard communication bus, e.g. KL6021.

stKL6DeviceOut22B: Output process image of a 22-byte data terminal [▶ 60] with standard communication bus, e.g. KL6041.

stEL6DeviceOut22B: Output process image of a 22-byte EtherCAT data terminal [▶ 60], e.g. EL6021.

stPcComDeviceOut: Output process image [▶ 63] of a serial PC interface.

5.3.29 ST_GENIbusPcComDeviceIn64B

Structure for linking in the input process image; must be used for the communication of a serial PC interface.

```
TYPE ST_GENIbusPcComDeviceIn64B :
STRUCT
  wStatus : WORD;
  arrData : ARRAY[0..63] OF BYTE;
END_TYPE
```

5.3.30 ST_GENIbusPcComDeviceOut64B

Structure for linking in the input process image; must be used for the communication of a serial PC interface.

```
TYPE ST_GENIbusPcComDeviceOut64B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..63] OF BYTE;
END_TYPE
```

5.3.31 ST_GENIbusReplyClassEntry

Response structure containing the data of a response APDU for processing within the library.

```
TYPE ST_GENIbusReplyClassEntry :
STRUCT
  byClass      : BYTE;
  eACK         : E_GENIbusACK;
  eOS          : E_GENIbusOS;
  iEntryCount  : INT;
  arrEntry     : ARRAY[0..GENIBUS_MAX_APDU_LENGTH] OF ST_GENIbusReplyDataEntry;
  sASCIIString : STRING(64);
END_STRUCT
END_TYPE
```

byClass: Data class.

eACK: Acknowledge-code [▶ 56] of the GENIbus device.

eOS: Operation display [▶ 58] (GET/SET/INFO).

iEntryCount: Number of data points (ID codes) used within the APDU.

arrEntry: Contents of the data points [▶ 63] (ID codes).

sASCIIString: String evaluation for data class 7.

5.3.32 ST_GENIbusReplyDataEntry

Contents of an element of a response APDU: value and information.

```
TYPE ST_GENIbusReplyDataEntry :
STRUCT
  byValue      : BYTE;
  byInfoHead   : BYTE;
  byUnitIndex  : BYTE;
  byZeroH      : BYTE;
  byRangeZeroL : BYTE;
END_STRUCT
END_TYPE
```

byValue: Raw value.

byInfoHead: Information head containing among other things the scaling information.

byUnitIndex: Sign and unit code.

byZeroH: Zero point in the case of normal range and zero-point scaling OR high-byte zero point in the case of extended scaling.

byRangeZeroL: Range in the case of normal range and zero-point scaling OR low-byte zero point in the case of extended scaling.

5.3.33 ST_GENIbusRequestClassEntry

Command or query structure containing the data of a request APDU for processing within the library.

```
TYPE ST_GENIbusRequestClassEntry :
STRUCT
  byClass      : BYTE;
  eOS          : E_GENIbusOS;
  byEntryCount : BYTE;
  arrEntry     : ARRAY[0..GENIBUS_MAX_APDU_LENGTH] OF ST_GENIbusRequestDataEntry;
END_STRUCT
END_TYPE
```

byClass: Data class.

eOS: Operation display [▶ 58] (GET/SET/INFO).

byEntryCount: Number of data points (ID codes) used within the APDU.

arrEntry: Field with addresses of the data points [▶ 64] (ID codes) and, if applicable, the values to be written.

5.3.34 ST_GENIbusRequestDataEntry

Address and, if applicable, the value to be written within a request APDU.

```
TYPE ST_GENIbusRequestDataEntry :
STRUCT
  byIDCode : BYTE;
  byValue  : BYTE;
END_STRUCT
END_TYPE
```

byIDCode: Address.

byValue: Value to be written.

5.3.35 ST_GENIbusResponseTable

Response buffer.

```
TYPE ST_GENIbusResponseTable :
STRUCT
  arrResponseTableItem           : ARRAY[1..GENIBUS_COMMAND_BUFFER_ENTRIES] OF ST_GENIbusResponseTab
leItem;
  byResponseTableCounter         : BYTE;
  byResponseTableMaxCounter      : BYTE;
  uiResponseTableOverflowCounter : UINT;
  bLockSemaphore                 : BOOL;
END_STRUCT
END_TYPE
```

arrResponseTableItem: Response buffer [▶ 65].

byResponseTableCounter: Current buffer demand.

byResponseTableMaxCounter: Maximum buffer demand.

uiResponseTableOverflowCounter: Number of buffer overflows.

bLockSemaphore: Write protection during the processing of a command.

5.3.36 ST_GENIbusResponseTableItem

Individual element in the response buffer.

```
TYPE ST_GENIbusResponseTableItem :
STRUCT
  byAddr           : BYTE;
  byLength         : BYTE;
  eSD              : E_GENIbusSD;
  arrAPDUs         : ARRAY[1..GENIBUS_MAX_APDU_NUMBER] OF ST_GENIbusReplyClassEntry;
  byRFS            : BYTE;
  udiMessageHandle : UDINT;
  udiErrorId       : UDINT;
END_STRUCT
END_TYPE
```

byAddr: Device destination address.

eAddrType: Single, multiple, or collective command.

eSD: Start delimiter [▶ 58] of the telegram.

arrAPDUs: Collection of the APDUs [▶ 63] (Application Program Data Units) to be transmitted.

byRFS: Not yet used: "Request from Slave".

udiMessageHandle: Pointer to the current buffer element.

udiErrorId: If an error has occurred in the FB_GENIbusCommunication [▶ 46]() function block, the corresponding error code will be saved here for further evaluation.

5.3.37 ST_GENIbusSerComBuffer

Serial communication buffer – for transmission and reception alike.

```
TYPE ST_GENIbusSerComBuffer :
STRUCT
  arrBuffer    : ARRAY[0..GENIBUS_MAX_TELEGRAM_LENGTH] OF BYTE;
  uiDataLength : UINT;
  bBlocked     : BOOL;
END_STRUCT
END_TYPE
```

5.4 Error codes

Value (hex)	Value (dec)	ErrArg	Description
0x0000	0	n/a	No error.
0x8001	32769	n/a	Internal error: no AMS-Net ID is read out. The process image is possibly not correctly linked.
0x8002	32770	n/a	Incorrect baud rate entry.
0x8003	32771	Sub-function-block error number	Internal error while writing the configuration data. *udiErrorArg* contains the error number of the write function block FB_EcCoESdoWrite() of the internally used library TcEtherCAT.lib.
0x8004	32772	n/a	Internal error: Incorrect pointer assignment *pRegComIn*/ *pRegComOut*. One of the two pointers points to the address 0.
0x8005	32773	n/a	Timeout error during the register communication. The attempt to configure the terminal with the proper communication-parameters was not successful. The

Value (hex)	Value (dec)	ErrArg	Description
			reason could be either a defective terminal or a wrong configuration of the communication to the PLC or a K-Bus-overload. Please check the variable-linking in the system-manager based on the examples. Make sure, the communication-variables are assigned to the fast task in the system-manager. If you use analogue-terminals, increase the task-cycle-time of the fast communication-task beginning with 5ms. If you do so, do not forget to reassign the communication-variables to the fast task in the system-manager after rescanning the PLC-project.
0x8019	32793	n/a	Invalid master address. Valid range: 0 - 31.
0x8020	32800	Sub-function-block error number	Error while configuring a KL6xxx (5 bytes of data). *udiErrorArg* contains the error number of the internal configuration function block.
0x8021	32801	Sub-function-block error number	Error while configuring a KL6xxx (22 bytes of data). *udiErrorArg* contains the error number of the internal configuration function block.
0x8022	32802	Sub-function-block error number	Error while configuring a KL6xxx (22 bytes of data). *udiErrorArg* contains the error number of the internal configuration function block.
0x8023	32803	1	Incorrect communication type (*eGENIbusComMode* input).
		2	Incorrect pointer assignment. One of the two addresses of the selected input/output variable (*stGENIbusInData*/ *stGENIbusOutData*) points to the address 0.
		3	Communication via an EtherCAT Terminal is selected. However, the EL6xxx terminal is not in the ?OP state?.
		4	The EL6xxx terminal contains incorrect data. This is signalled by the fact that the input variable ?WC State? is set to 1.
0x8024	32804	Sub-function-block error number	Error during the creation of the serial telegram. *udiErrorArg* contains the error number of the internal function block.
0x8025	32805	Sub-function-block error number	Error during the serial data transmission. *udiErrorArg* contains the error number of the internal function block.
0x8026	32806	Sub-function-block error number	Error during the evaluation of the serial telegram. *udiErrorArg* contains the error number of the internal function block.
0x8027	32807	n/a	Timeout error during the transmit-receive cycle.
0x8030	32816	n/a	Index error while transmitting the telegram.
0x8031	32817	n/a	Index error while receiving the telegram.
0x8032	32818	n/a	Incorrect data length while receiving the telegram.
0x8033	32819	n/a	Timeout error while receiving the telegram.
0x8034	32820	n/a	Timeout error while sending the telegram. 100 PLC-cycles have passed while the terminal was not able to send the data. The telegram was skipped.
0x8040	32832	Incorrect OS	The response telegram contains an unknown ?Operation Specifier? (OS), see *GENIbus Protocol Specification.*
0x8041	32833	n/a	Telegram length error.
0x8042	32834	n/a	Telegram CRC check error.
0x8045	32837	Maximum number of APDUs	Error during the conversion to a telegram: too many APDU entries. *udiErrorArg* shows the maximum possible number of APDU entries.

Value (hex)	Value (dec)	ErrArg	Description
0x8049	32841	n/a	Invalid device (slave) address. Valid range: 1 - 200.
0x8050	32848	n/a	Incorrect class entry *byClass.*
0x8051	32849	n/a	Incorrect entry *eCommandPriority.*
0x8052	32850	n/a	Incorrect entry *eSetOpMode.*
0x8053	32851	n/a	Incorrect entry *eSetCtrlMode.*
0x8054	32852	n/a	Incorrect entry *eSetNightReductionMode.*
0x8055	32853	n/a	Incorrect entry *eSetKeyMode.*
0x8056	32854	n/a	Command buffer overflow (*stCommandBuffer*): not all previously transmitted commands have been processed.
0x8057	32855	n/a	Timeout error (runtime monitoring) with the response telegram.
0x8058	32856	n/a	The response telegram of the GENIbus device reports ? Data Class Unknown?, see *GENIbus Protocol Specification*, feedback entry ?ACK?.
0x8059	32857	n/a	The response telegram of the GENIbus device reports ? Data Item ID Unknown?, see *GENIbus Protocol Specification*, feedback entry ?ACK?.
0x805A	32858	n/a	The response telegram of the GENIbus device reports ? Operation illegal or Data Class write buffer is full?, see *GENIbus Protocol Specification*, feedback entry ?ACK?.
0x805B	32859	n/a	Unknown?ACK?entry in response telegram.
0x805C	32860	transferred error number	The FB_GENIbusCommunication() function block has already detected an error and entered it in the response structure *stResponseTableItem*. *udiErrorArg* contains the error number of the FB_GENIbusCommunication() function block.
0x805D	32861	transferred error number	An internal error has occurred during the scaling. *udiErrorArg* contains the internal error number.
0x8060	32864	n/a	Data size (*eDataSize*) invalid.
0x8061	32865	n/a	Invalid Scale-Info parameter in the telegram (*eSIF*), see *GENIbus Protocol Specification*, feedback entry ?SIF?.
0x8062	32866	n/a	Invalid combination of data size and Scale-Info.
0x8063	32867	n/a	No info data available.
0x8064	32868	n/a	The read-out unit index is not assigned to any unit, i.e. it doesn?t exist in the internal tables.

6 Appendix

6.1 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: https://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963 157
Fax:	+49 5246 963 9157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963 460
Fax:	+49 5246 963 479
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963 0
Fax:	+49 5246 963 198
e-mail:	info@beckhoff.com
web:	https://www.beckhoff.com

More Information:
www.beckhoff.com/tx1200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com