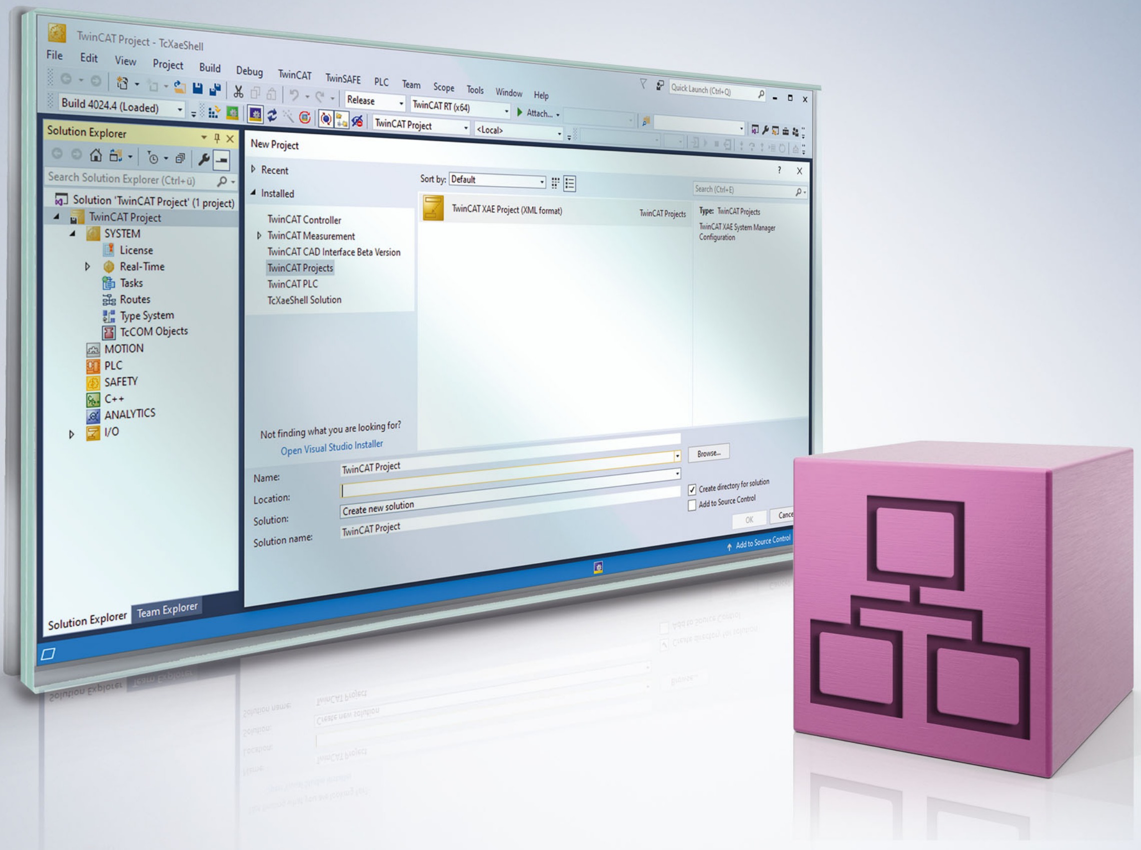
BECKHOFF New Automation Technology

Manual | EN

TF6280

TwinCAT 3 | EtherNet/IP Adapter



2024-02-21 | Version: 1.3.2

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.
The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all 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 notice.
No claims to modify 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 and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

Patents

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

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

Copyright

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are 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 technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

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 Overview

In combination with a network-capable Beckhoff PC, the function TF6280 TwinCAT EtherNet/IP Adapter can be used to create an Ethernet/IP adapter.

Up to eight adapters can be parameterized with a physical interface. A virtual MAC address is formed, through which up to eight EtherNet/IP adapters can be operated on a PC via an Ethernet interface.

Technical data	**TF6280**							
Requires	TC1200 from build 4020							
Target system	Windows XP, Windows 7/8, Windows CE							
Performance class (pp)	20	30	40	50	60	70	80	90
	–	–	X	X	X	X	X	X

Ordering information	
TF6280-00pp	TC3 EtherNet/IP Adapter

The function TF6280 TwinCAT EtherNet/IP Adapter enables data exchange with an EtherNet/IP master. Both multicast and broadcast are supported. The function TF6280 TwinCAT EtherNet/IP Adapter can behave like eight EtherNet/IP adapters.

For sample, it is possible to:

- connect a master with eight adapters
- connect up to eight masters with 8 adapters

This way more data can be transported or the master can be operated with different cycle times.

In an EtherNet/IP network, the TF6280 behaves as an adapter device. No further configuration via an EtherNet/IP master is required. The configurator in TwinCAT 3.1 is used for the configuration, e.g. by specifying the IP settings and the number of data. The only requirement for a connection to be established is that the data itself must be set in the same way in the EtherNet/IP master.

EtherNet/IP

EtherNet/IP (Ethernet Industrial Protocol, EIP) is a real-time Ethernet protocol, which was disclosed and standardized by the ODVA (Open DeviceNet Vendor Association). The protocol is based on TCP, UDP and IPv4.

Further information can be found at www.odva.org or https://en.wikipedia.org/wiki/Ethernet/IP.

3 Prerequisites

Software

TF6280 is included in **TwinCAT** version **3.1** build **4020.28**. No further installation is required.

Older product versions

Older versions are beta versions. Delete any older EtherNet/IP device configurations and create a new configuration.

Hardware

For using the TF6280, the target system has to have an Intel® network chipset (see: Verifying the hardware [▸ 9]).

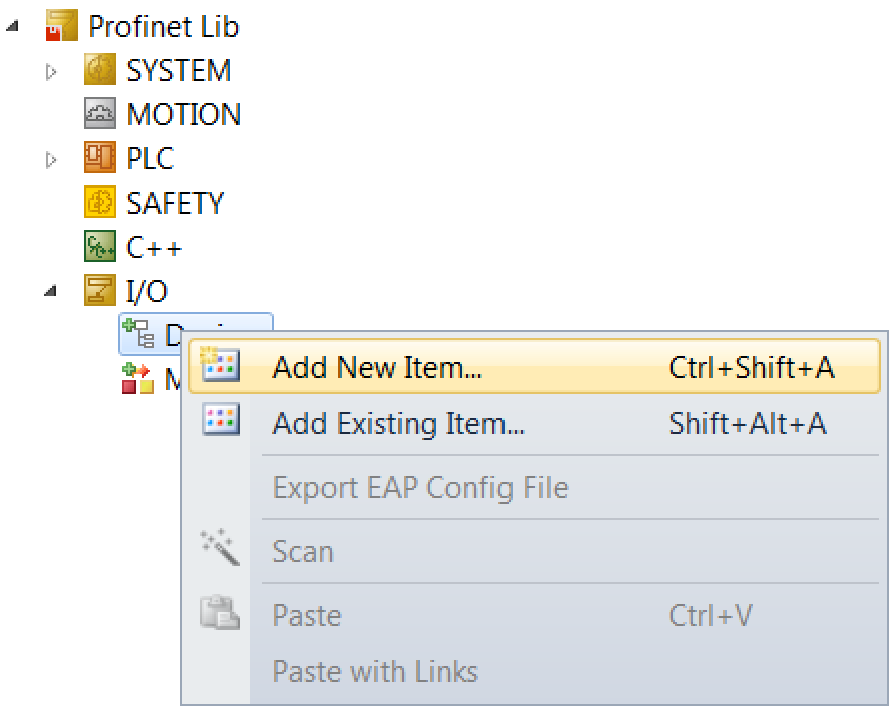
Beckhoff PC

Beckhoff PC systems are usually preconfigured for the operation of EtherNet/IP devices.

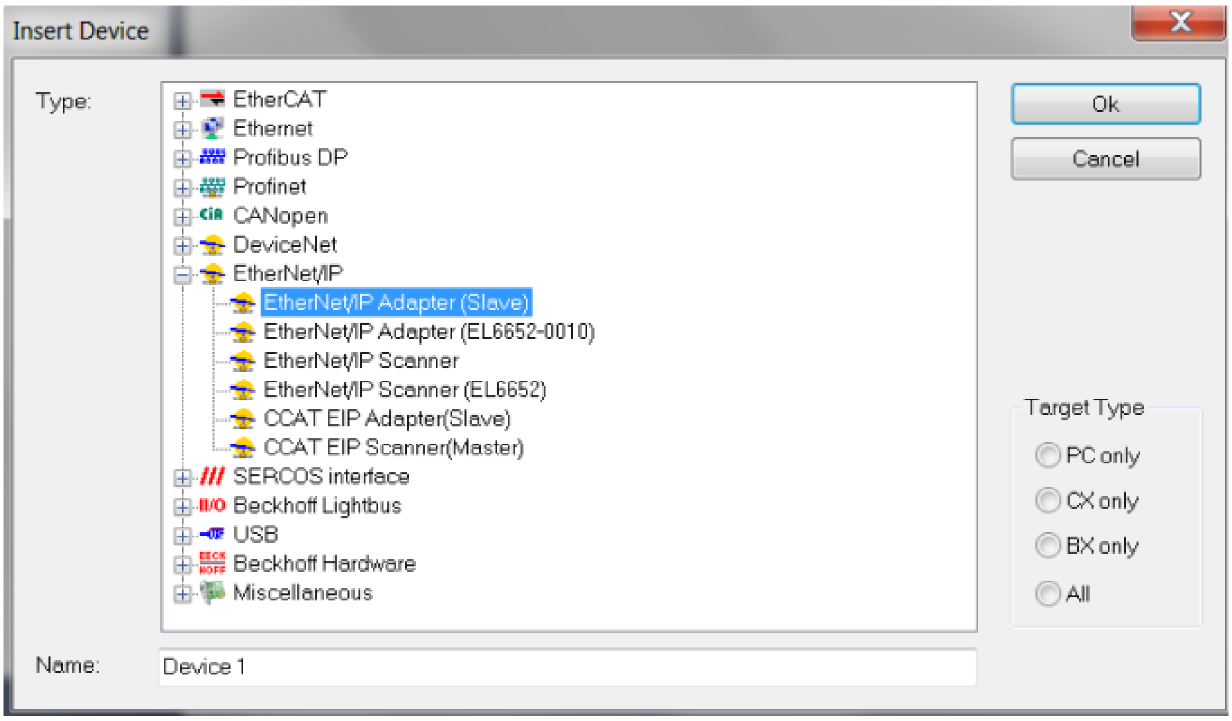
3.1 Verifying the hardware

Check whether the network interface is suitable

1. Create an EtherNet/IP slave. Right-click on **Devices** and add a new device (**Add New Item…**).



2. Select **EtherNet/IP Adapter (Slave).**



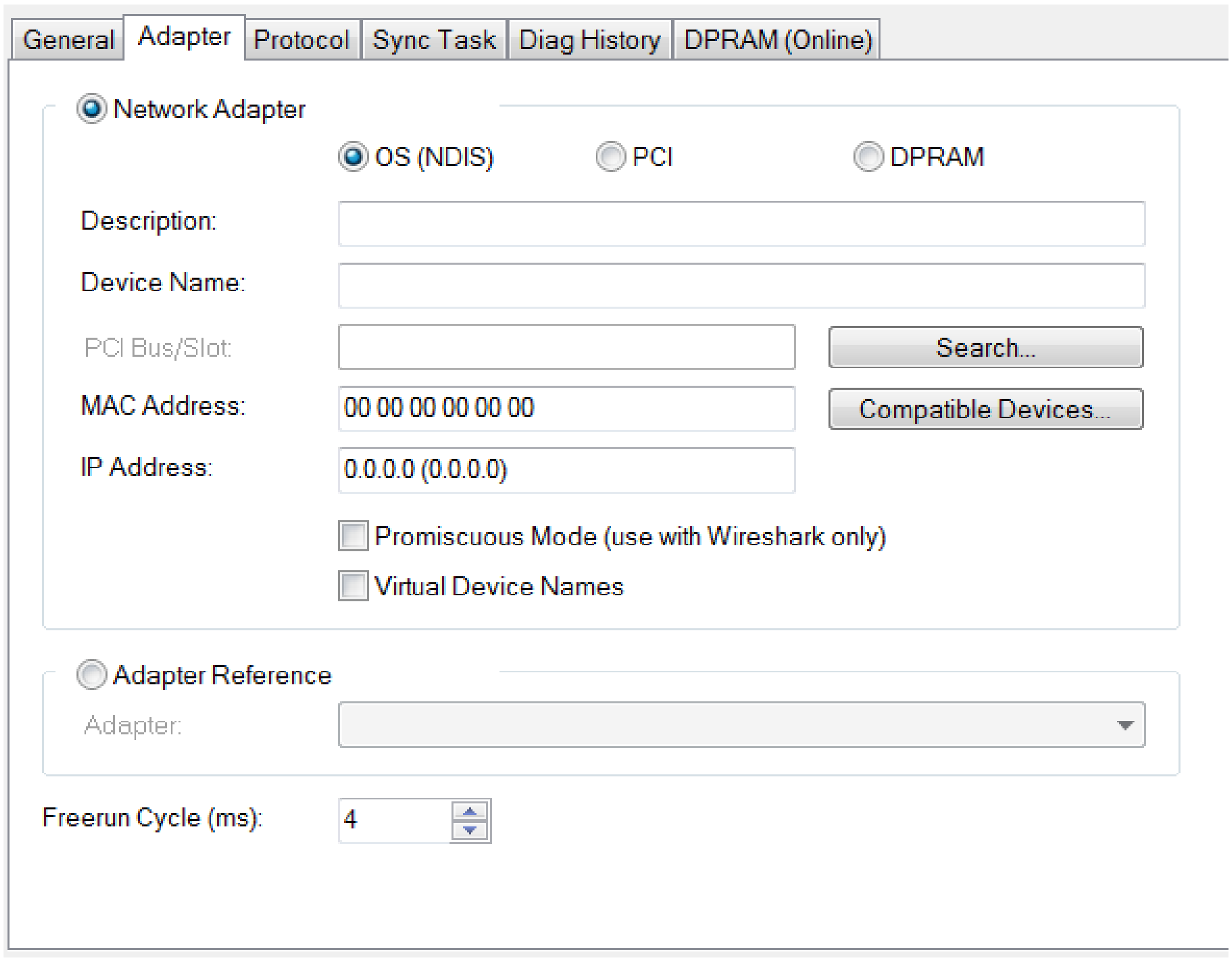
Insert Device

Type:
- EtherCAT
- Ethernet
- Profibus DP
- Profinet
- CANopen
- DeviceNet
- EtherNet/IP
  - EtherNet/IP Adapter (Slave)
  - EtherNet/IP Adapter (EL6652-0010)
  - EtherNet/IP Scanner
  - EtherNet/IP Scanner (EL6652)
  - CCAT EIP Adapter(Slave)
  - CCAT EIP Scanner(Master)
- SERCOS interface
- Beckhoff Lightbus
- USB
- Beckhoff Hardware
- Miscellaneous

Name: Device 1

Target Type: PC only / CX only / BX only / All

3. Now select the adapter and find the appropriate Ethernet interface (**Search**…).



[✓ Network Adapter]

[✓ OS (NDIS)] PCI DPRAM

Description:

Device Name:

PCI Bus/Slot:

MAC Address: 00 00 00 00 00 00

IP Address: 0.0.0.0 (0.0.0.0)

☐ Promiscuous Mode (use with Wireshark only)

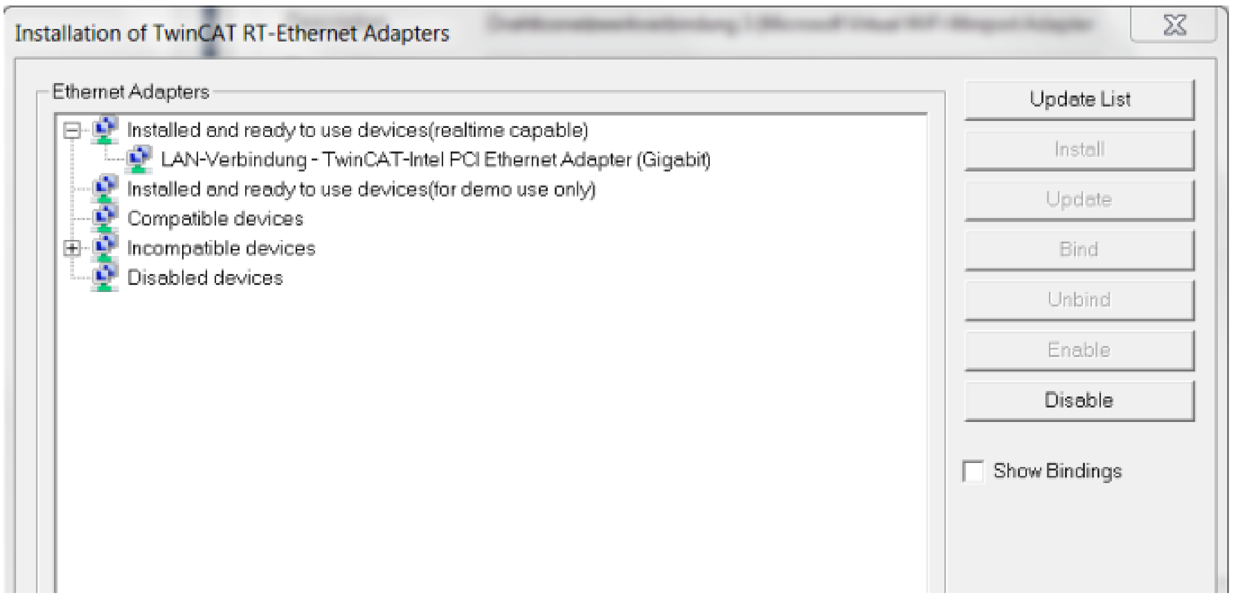
☐ Virtual Device Names

Adapter Reference

Adapter:

Freerun Cycle (ms): 4

4. Select a “real-time capable” interface under **Compatible devices**.



Installation of TwinCAT RT-Ethernet Adapters

Ethernet Adapters

- Installed and ready to use devices(realtime capable)
  - LAN-Verbindung - TwinCAT-Intel PCI Ethernet Adapter (Gigabit)
- Installed and ready to use devices(for demo use only)
- Compatible devices
- Incompatible devices
- Disabled devices

⇨ You can install the real-time driver.

No “real-time capable” network interface available

If the list contains no network interfaces under **Compatible devices**, the TF6280 function cannot be used on the present hardware.

4 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

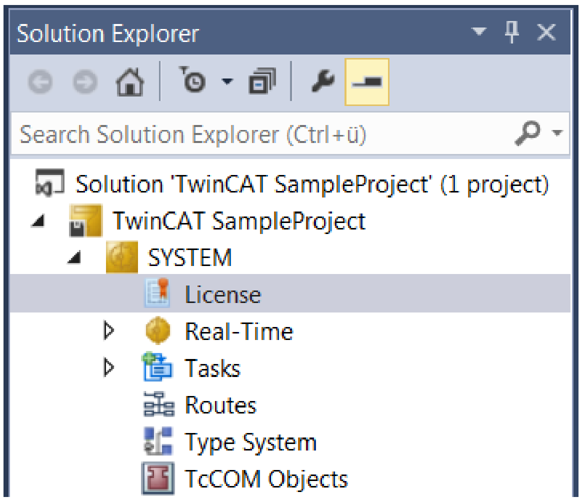
Licensing the full version of a TwinCAT 3 Function

A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "TwinCAT 3 Licensing".

Licensing the 7-day test version of a TwinCAT 3 Function

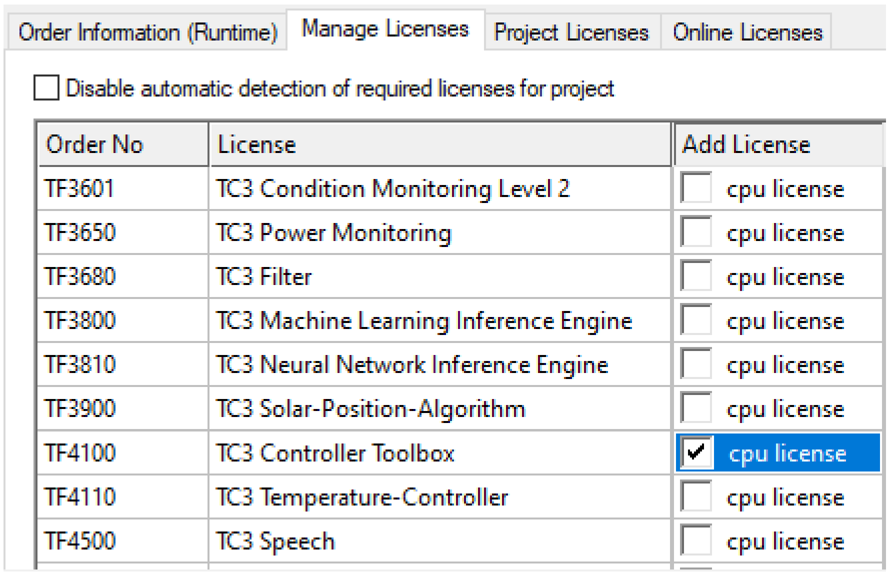
A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
   ⇨ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.
4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



⇨ The TwinCAT 3 license manager opens.

5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").

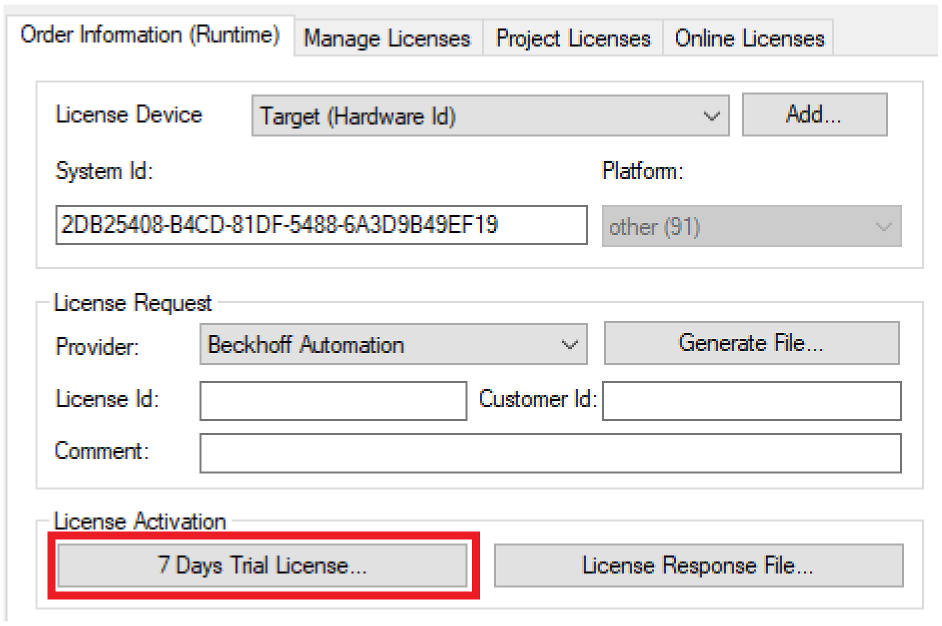


Order Information (Runtime) | Manage Licenses | Project Licenses | Online Licenses

☐ Disable automatic detection of required licenses for project

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	☐ cpu license
TF3650	TC3 Power Monitoring	☐ cpu license
TF3680	TC3 Filter	☐ cpu license
TF3800	TC3 Machine Learning Inference Engine	☐ cpu license
TF3810	TC3 Neural Network Inference Engine	☐ cpu license
TF3900	TC3 Solar-Position-Algorithm	☐ cpu license
TF4100	TC3 Controller Toolbox	✓ cpu license
TF4110	TC3 Temperature-Controller	☐ cpu license
TF4500	TC3 Speech	☐ cpu license

6. Open the **Order Information (Runtime)** tab.
   ⇨ In the tabular overview of licenses, the previously selected license is displayed with the status "missing"**.**
7. Click **7-Day Trial License...** to activate the 7-day trial license.



Order Information (Runtime) | Manage Licenses | Project Licenses | Online Licenses

License Device: Target (Hardware Id)

System Id: 2DB25408-B4CD-81DF-5488-6A3D9B49EF19

Platform: other (91)

License Request

Provider: Beckhoff Automation

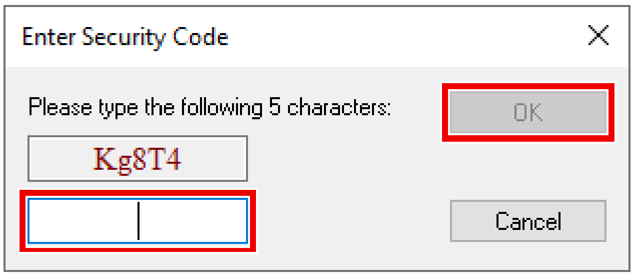
License Id:

Customer Id:

Comment:

License Activation

   ⇨ A dialog box opens, prompting you to enter the security code displayed in the dialog.



Enter Security Code

Please type the following 5 characters:

Kg8T4

8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.
   ⇨ In the tabular overview of licenses, the license status now indicates the expiry date of the license.

10. Restart the TwinCAT system.

⇨ The 7-day trial version is enabled.

5 Configuration

The most important settings in order to establish a connection with an EtherNet/IP scanner are:

- the IP address,
- the assembly instance numbers and thus the length of the data
- and the correct cycle time.

IP address:

The IP address can be assigned freely, although it should be from the same network class as the master. Otherwise a gateway must be entered, in order to route the protocol accordingly.

Assembly instance numbers:

The assembly instance numbers are permanently assigned and must be correctly set in the master. This also always includes the number of data or the size of the process image.

Cycle time:

The task cycle time in the TF6280 may not exceed the time on the master side, although it can be a fraction of that time. If, for sample, an EtherNet/IP cycle time of 10 ms is set on the master side, the task cycle time on the slave side can be 10 ms, 5 ms, 2 ms or 1 ms.

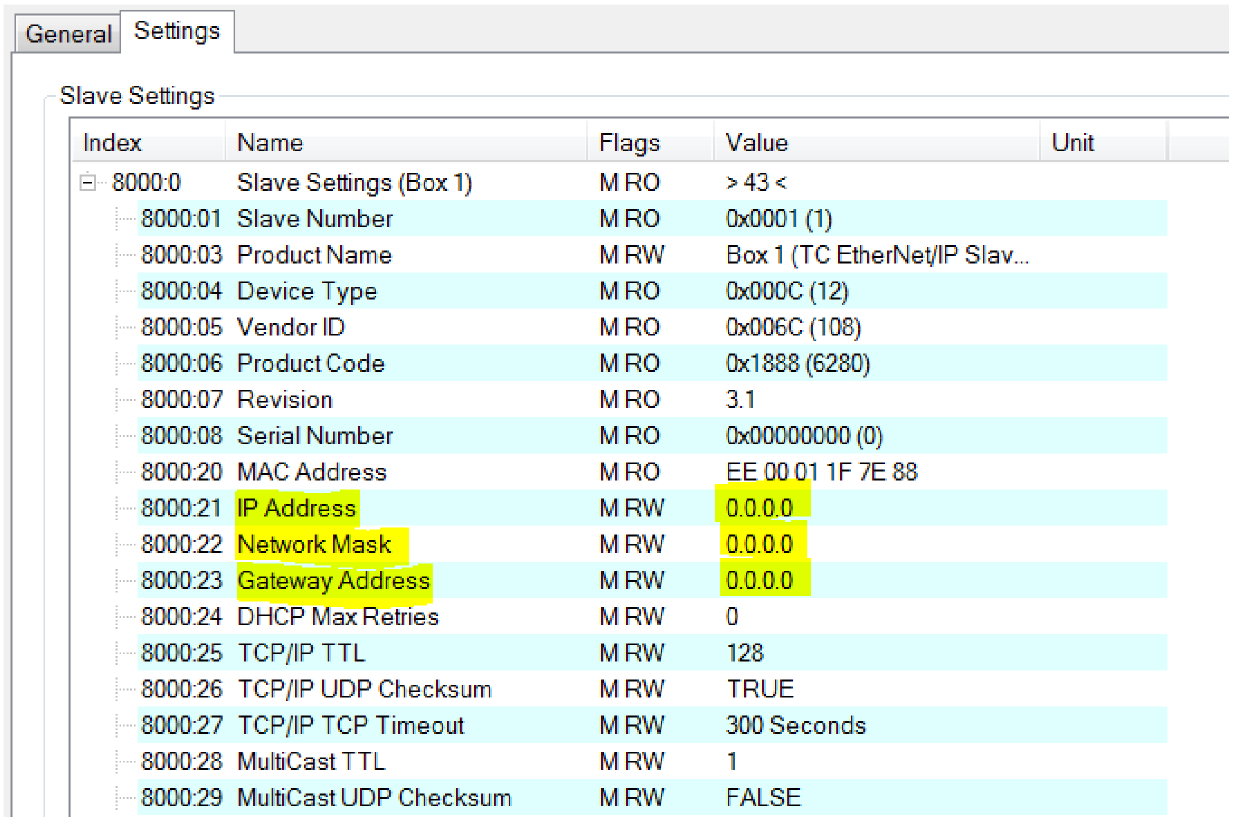
Recommended cycle time

EtherNet/IP enables cycle times of 1 ms or higher. The task can always be operated with 1 ms, as long as the system load of your systems [▶ 8] permits this.

5.1 Creating an EtherNet/IP slave

Once you have added an EtherNet/IP adapter, a slave is automatically added to your configuration.

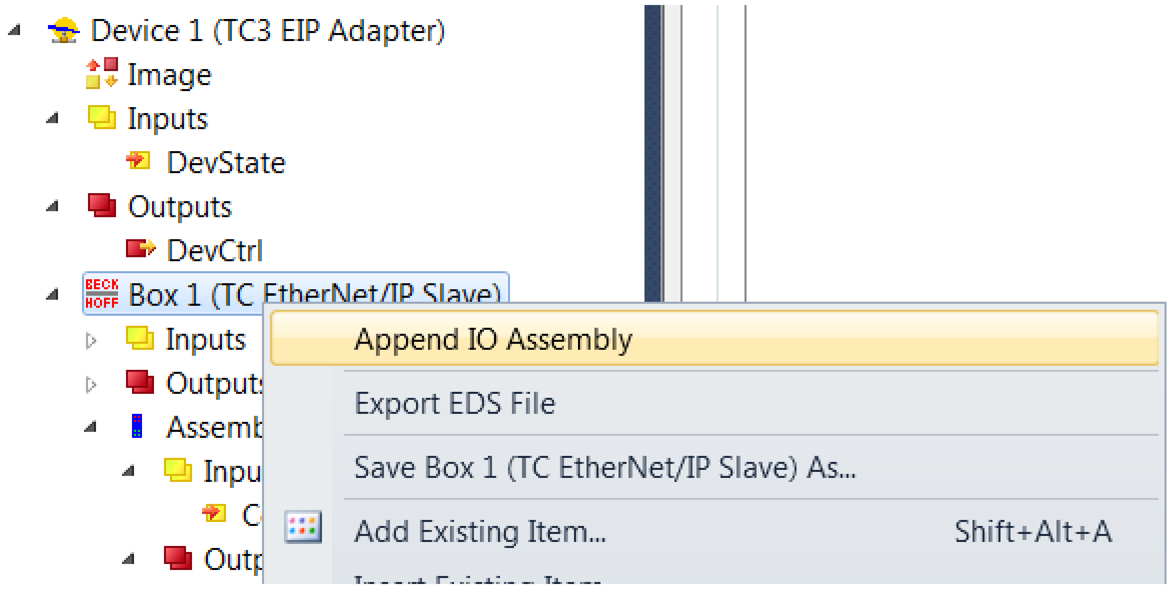
1. Set the IP address of the slave. (The IP address does not have to be the same as the IP address of the operating system.) Click on the box and switch to the **Settings** tab. Here you can set the **IP address**, the **network mask** and the **gateway address**.



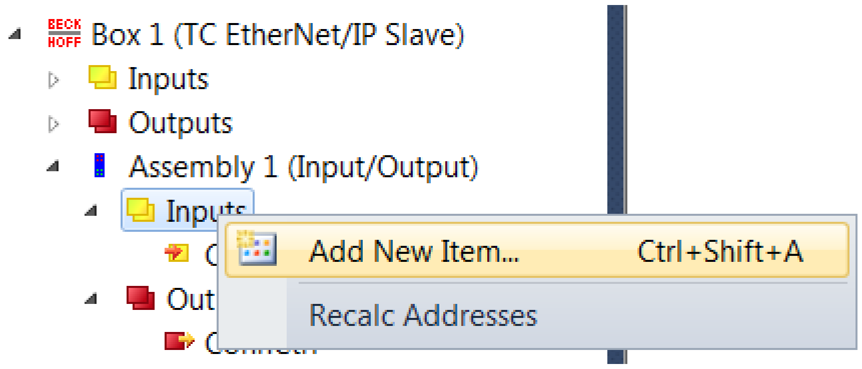
Index	Name	Flags	Value	Unit
8000:0	Slave Settings (Box 1)	M RO	> 43 <	
8000:01	Slave Number	M RO	0x0001 (1)	
8000:03	Product Name	M RW	Box 1 (TC EtherNet/IP Slav...	
8000:04	Device Type	M RO	0x000C (12)	
8000:05	Vendor ID	M RO	0x006C (108)	
8000:06	Product Code	M RO	0x1888 (6280)	
8000:07	Revision	M RO	3.1	
8000:08	Serial Number	M RO	0x00000000 (0)	
8000:20	MAC Address	M RO	EE 00 01 1F 7E 88	
8000:21	IP Address	M RW	0.0.0.0	
8000:22	Network Mask	M RW	0.0.0.0	
8000:23	Gateway Address	M RW	0.0.0.0	
8000:24	DHCP Max Retries	M RW	0	
8000:25	TCP/IP TTL	M RW	128	
8000:26	TCP/IP UDP Checksum	M RW	TRUE	
8000:27	TCP/IP TCP Timeout	M RW	300 Seconds	
8000:28	MultiCast TTL	M RW	1	
8000:29	MultiCast UDP Checksum	M RW	FALSE	

1 a) If the IP address is to be issued by a DHCP server in your network, enter the value `0.0.0.0` in the **IP address** field.
1 b) If the IP address of the operating system is to be used, enter the value `255.255.255.255` in the **IP address** field. The subnet mask and the gateway address can be used unchanged. When TwinCAT starts, the EtherNet/IP driver then uses the IP address of the system.
Please note the firewall settings [▶ 18].

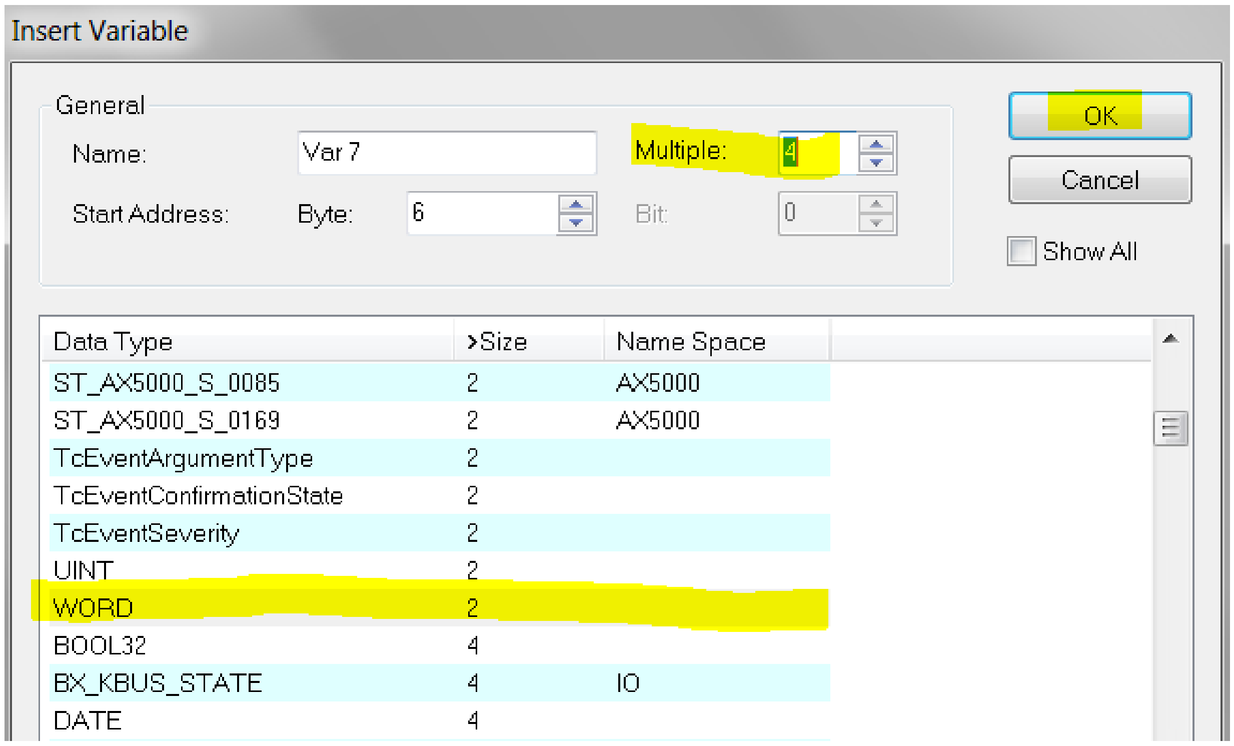
2. Click on the box and select **Append IO Assembly**.



3. To create data under Inputs, right-click on **Add New Item…**


Box 1 (TC EtherNet/IP Slave)
Inputs
Outputs
Assembly 1 (Input/Output)
Inputs
Out
Add New Item... Ctrl+Shift+A
Recalc Addresses

4. Now select the data format and the number of data to be transferred. The number of bytes will be important later. It can be read in the object tree. e.g.: Enter 4 words, i.e. 8 bytes of process data:

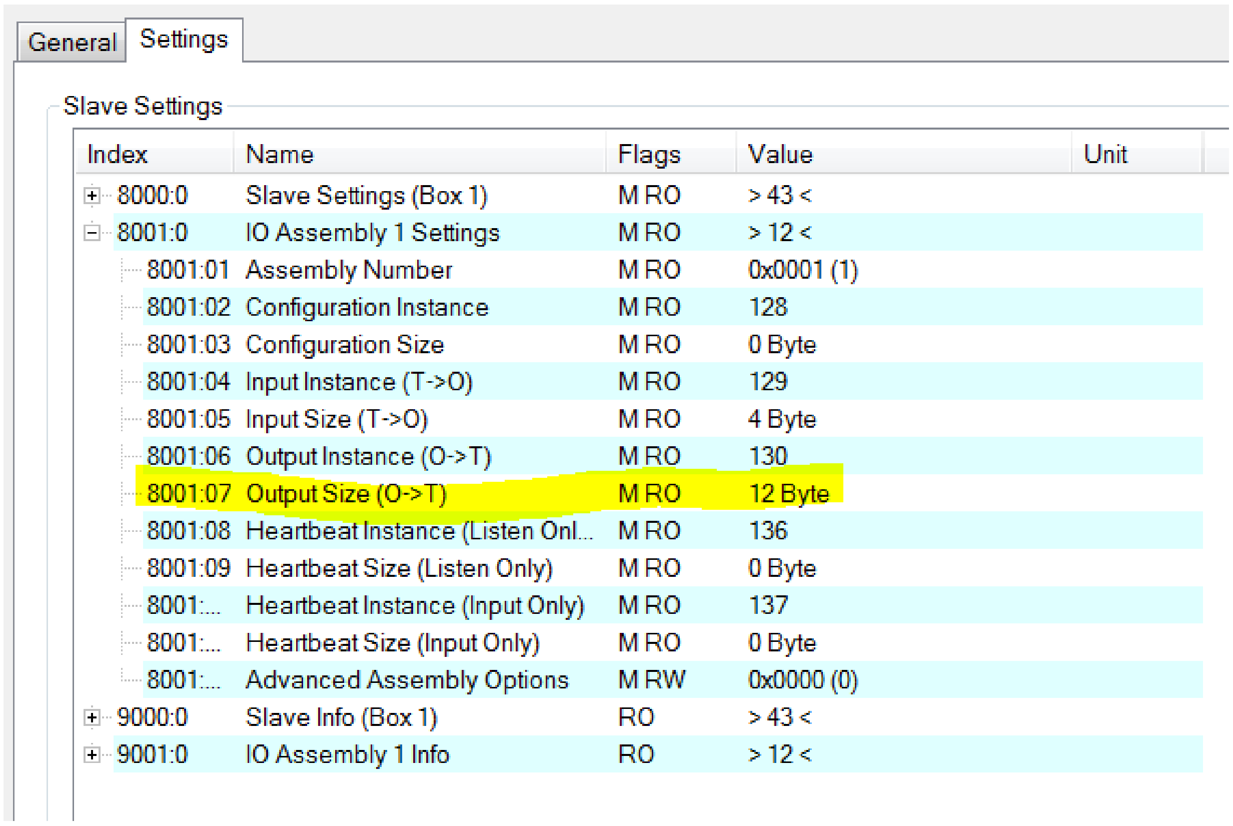


Name: Var 7
Multiple: 4
Start Address: Byte: 6
Bit: 0

Data Type	Size	Name Space
ST_AX5000_S_0085	2	AX5000
ST_AX5000_S_0169	2	AX5000
TcEventArgumentType	2	
TcEventConfirmationState	2	
TcEventSeverity	2	
UINT	2	
WORD	2	
BOOL32	4	
BX_KBUS_STATE	4	IO
DATE	4	

In addition there are 4 bytes for the ConnState. The ConnState currently has no function. It can be used for additional information in the future.

5. Therefore, 12 bytes of process data must be created. Navigate to the box and select the **Settings** tab.



General | Settings

Slave Settings

Index	Name	Flags	Value	Unit
+ 8000:0	Slave Settings (Box 1)	M RO	> 43 <	
− 8001:0	IO Assembly 1 Settings	M RO	> 12 <	
8001:01	Assembly Number	M RO	0x0001 (1)	
8001:02	Configuration Instance	M RO	128	
8001:03	Configuration Size	M RO	0 Byte	
8001:04	Input Instance (T->O)	M RO	129	
8001:05	Input Size (T->O)	M RO	4 Byte	
8001:06	Output Instance (O->T)	M RO	130	
8001:07	Output Size (O->T)	M RO	12 Byte	
8001:08	Heartbeat Instance (Listen Onl...	M RO	136	
8001:09	Heartbeat Size (Listen Only)	M RO	0 Byte	
8001:...	Heartbeat Instance (Input Only)	M RO	137	
8001:...	Heartbeat Size (Input Only)	M RO	0 Byte	
8001:...	Advanced Assembly Options	M RW	0x0000 (0)	
+ 9000:0	Slave Info (Box 1)	RO	> 43 <	
+ 9001:0	IO Assembly 1 Info	RO	> 12 <	

⇨ The length can be found in index field `0x8001:07`. The length is displayed from the master perspective. TwinCAT inputs are outputs in the master, hence the reference to output size here.

6. Now do the same with the outputs of the EtherNet/IP adapter.

⇨ Data creation is now complete. Now link the data with the PLC.

5.1.1 Firewall setting

The firewall must be enabled, if the EtherNet/IP address is to match the IP address of the operating system (OS). It is advisable to enable the firewall if the IP address of the EtherNet/IP scanner deviates from the IP setting of the operating system.

5.1.2 IP Routing

f IP routing is used, the IP address of the OS must be in a different subnet than the IP address of the Ethernet/IP adapter/scanner.
The Regkey can be different depending on the operating system and version, here only as an example, default is "0".
HKEY_LOCAL_MACHINE\ System\ CurrentControlSet\ Services\ Tcpip\ Parameters "IPEnableRouter"

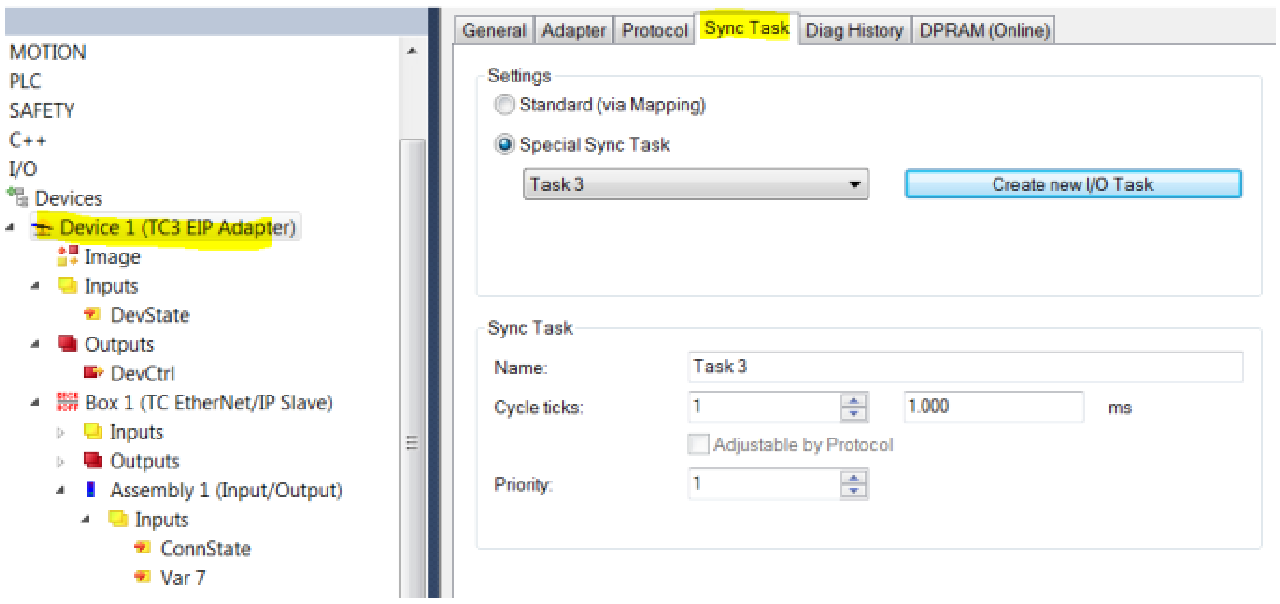
5.2 Setting the cycle time

The cycle time of the EtherNet/IP adapter (slave) is specified by the master. The task on the TwinCAT system must operate with at least the same speed.

Recommended cycle time

EtherNet/IP enables cycle times of 1 ms or higher. The task can always be operated with 1 ms, as long as the system load of your systems [▶ 8] permits this.

To set the task cycle time navigate to the **EIP Adapter device**, then to the **Sync Task** tab and set the time.



Settings

Standard (via Mapping) [✓ Special Sync Task]

Task 3

Sync Task

Name: Task 3

Cycle ticks: 1 1.000 ms

Adjustable by Protocol

Priority: 1

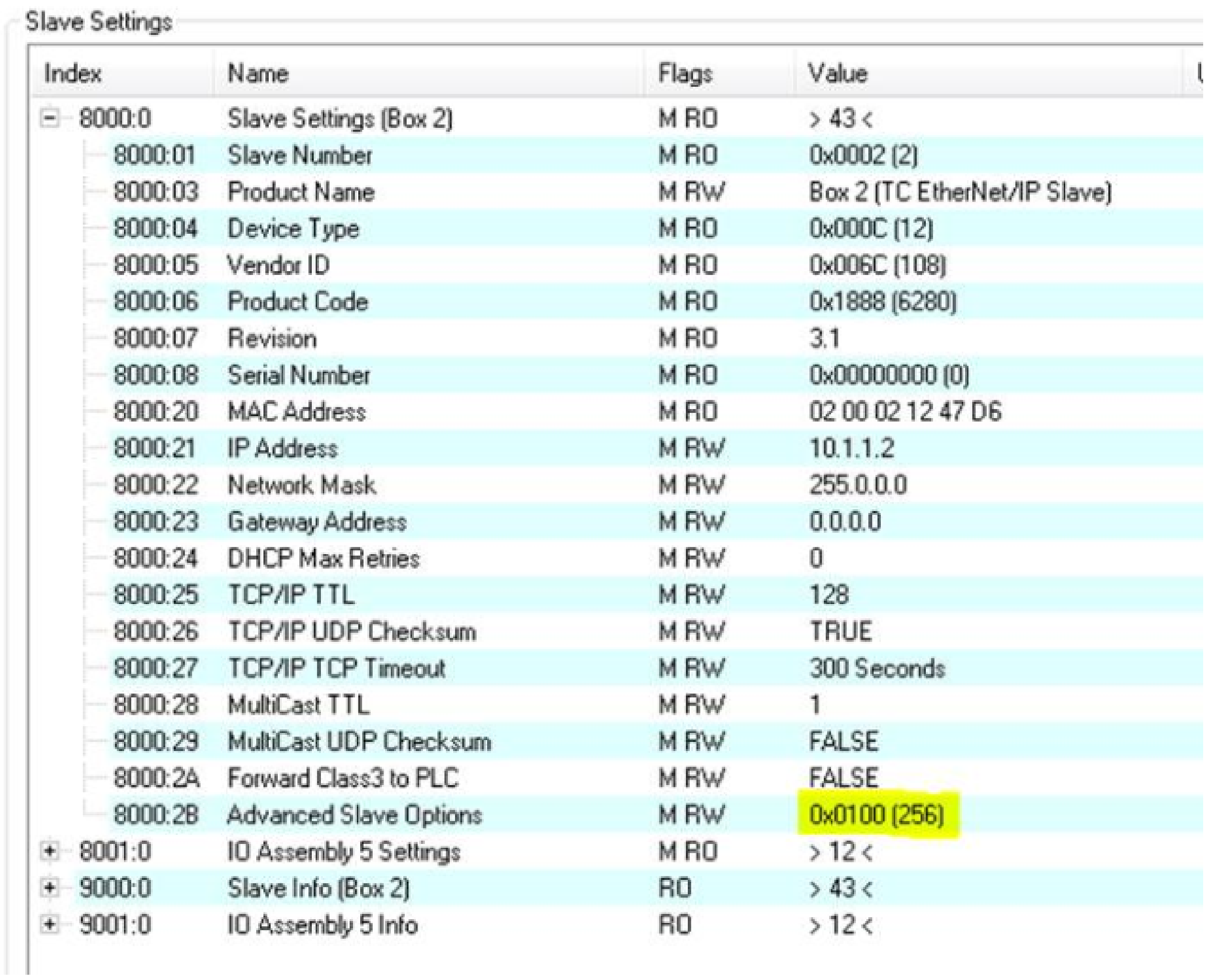
Use a dedicated Sync Task

Use a dedicated Sync Task, since mapping via the PLC can result in the task being stopped, e.g. if a breakpoint is encountered, with the result that the EtherNet/IP connection is interrupted.

5.3 Changing EtherNet/IP settings

For the setting, the Store Category [▶ 21] must be specified in the TwinCAT system configuration. This is entered in the object F8000:2B "Advanced Options" in all EtherNet/IP devices.
If the corresponding bit is set, the IP address from the memory is used. If no value is entered, the bit is ignored, and the parameters of the TwinCAT system are used.

In the following sample bit 8 (0x0100) is set, which means that Store Category 1 is selected, which affects the IP settings (index 0x8000: 21…23).



Slave Settings

Index	Name	Flags	Value
8000:0	Slave Settings (Box 2)	M RO	> 43 <
8000:01	Slave Number	M RO	0x0002 (2)
8000:03	Product Name	M RW	Box 2 (TC EtherNet/IP Slave)
8000:04	Device Type	M RO	0x000C (12)
8000:05	Vendor ID	M RO	0x006C (108)
8000:06	Product Code	M RO	0x1888 (6280)
8000:07	Revision	M RO	3.1
8000:08	Serial Number	M RO	0x00000000 (0)
8000:20	MAC Address	M RO	02 00 02 12 47 D6
8000:21	IP Address	M RW	10.1.1.2
8000:22	Network Mask	M RW	255.0.0.0
8000:23	Gateway Address	M RW	0.0.0.0
8000:24	DHCP Max Retries	M RW	0
8000:25	TCP/IP TTL	M RW	128
8000:26	TCP/IP UDP Checksum	M RW	TRUE
8000:27	TCP/IP TCP Timeout	M RW	300 Seconds
8000:28	MultiCast TTL	M RW	1
8000:29	MultiCast UDP Checksum	M RW	FALSE
8000:2A	Forward Class3 to PLC	M RW	FALSE
8000:2B	Advanced Slave Options	M RW	0x0100 (256)
8001:0	IO Assembly 5 Settings	M RO	> 12 <
9000:0	Slave Info (Box 2)	RO	> 43 <
9001:0	IO Assembly 5 Info	RO	> 12 <

To use Store Category 1 and 2, 0x0300 should be entered in object 8000:2B. Only bits 8 and 9 should be used. All other bits are reserved and must not be used.

ADS function blocks are used for reading or writing the settings from/to the PLC.

5.3.1 Object description

Offset	Name	Data Type	SubIndex	Store Category	
				1	2
0x00..0x01	ID	UINT16	1		
0x02..0x03	Reserved	UINT16	-		
0x04..0x23	Product Name	BYTE[32], STRING(31)	3		X
0x24..0x27	Device Type	UINT32	4		
0x28..0x2B	Vendor ID	UINT32	5		
0x2C..0x2F	Product Code	UINT32	6		X
0x30..0x33	Revision	UINT32	7		
0x34..0x37	Serial Number	UINT32	8		
0x38..0x7D	Reserved	BYTE[70]	-		
0x7E..0x83	MAC Address	BYTE[6]	32		
0x84..0x87	IP Address	UINT32	33	X	
0x88..0x8B	Network Mask	UINT32	34	X	
0x8C..0x8F	Gateway Address	UINT32	35	X	
0x90..0x91	DHCP Max Retries	UINT16	36		
0x92..0x93	TCP/IP TTL	UINT16	37		
0x94..0x95	TCP/IP UDP Checksum	UINT16	38		
0x96..0x97	TCP/IP TCP Timeout	UINT16	39		
0x98..0x99	Multicast TTL	UINT16	40		
0x9A..0x9B	Multicast Checksum	UINT16	41		
0x9C..0x9D	Forward Class3 to PLC	UINT16	42		
0x9E..0x9F	Flags	UINT16	43		
0xA0..0xFF	Reserved	Byte[96]	-		

Store Category

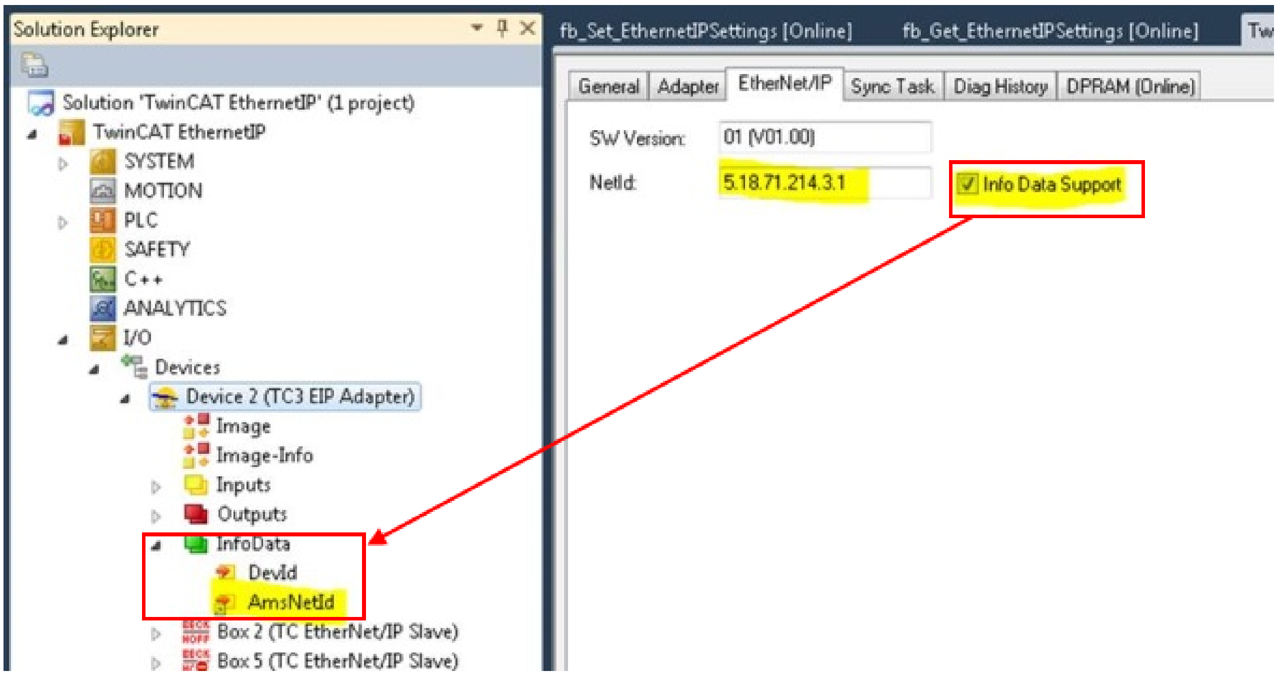
The “Store Category” determines which settings are overwritten with the values from the non-volatile memory. Bits 9 - 8 have to be set accordingly in the project under “Flags”. In order to modify both, both bits must be set.
(Bit9=Cat2, Bit8=Cat1)

5.3.2 ADS-Write command

AmsNetId

The AMSNetId can be found under the “EtherNet/IP” tab in the “NetID” field. When you select the option “Info Data Support” it is linked directly.

The advantage of a direct link is that it always retrieves the current AMSNETID, even if controllers are used that use different AMSNETIDs. The AMSNETID of the EtherNet/IP adapter therefore does not have to be read manually.



ADS port number

For the function “EtherNet/IP Adapter” set the ADS port number to a fixed value of `0xFFFF`.

Slave

IDXGRP: `0x0001F480`
IDXOFFS: `0x00000000`

Setting for setting (4 bytes + object size (256 bytes))

Byte Offset 0: `0x45`
Byte Offset 1: `0x23`
Byte Offset 2: ObjIndex LoByte (e.g. `0x8000` for **slave 1** and `0x8010` for **slave 2**)
Byte Offset 3: ObjIndex HiByte
Byte Offset 4-260: Data of the object (see object description below)

Setting for resetting (4 bytes)

Byte Offset 0: `0x00`
Byte Offset 1: `0x00`
Byte Offset 2: ObjIndex LoByte (e.g. `0x8000` for **slave 1** and `0x8010` for **slave 2**)
Byte Offset 3: ObjIndex HiByte

Accept changes

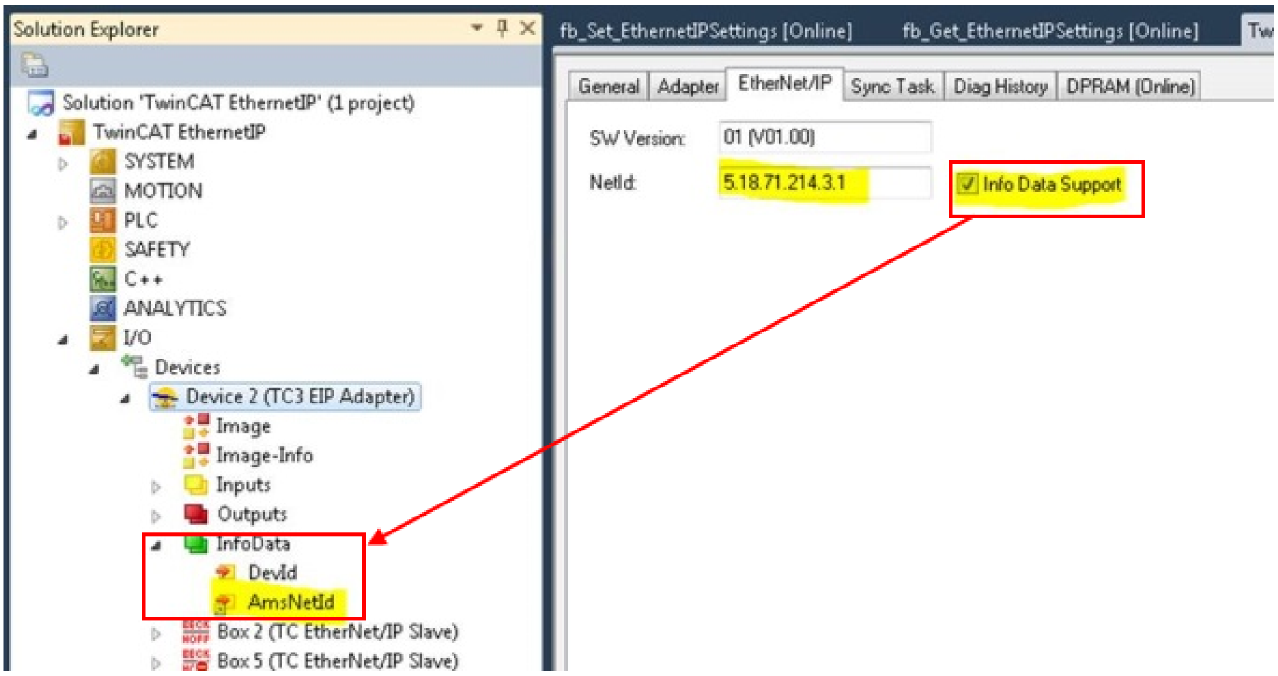
After setting the properties restart TwinCAT for the TF6280, after which the new settings are applied and valid. The settings remain stored and don't have to be loaded again, unless there are changes.

5.3.3 ADS-Read command

AmsNetId

The AMSNetId can be found under the “EtherNet/IP” tab in the “NetID” field. When you select the option “Info Data Support” it is linked directly.

The advantage of a direct link is that it always retrieves the current AMSNETID, even if controllers are used that use different AMSNETIDs. The AMSNETID of the EtherNet/IP adapter therefore does not have to be read manually.



SW Version: 01 (V01.00)
NetId: 5.18.71.214.3.1
[✓] Info Data Support

ADS port number

For the function “EtherNet/IP Adapter” set the ADS port number to a fixed value of `0xFFFF`.

Slave

IDXGRP: `0x1F480`
IDXOFFS: `0x8000` for the **first slave**
IDXOFFS: `0x8010` for the **second slave**
IDXOFFS: `0x8020` for the **third slave**
…
IDXOFFS: `0x8070` for the **eights slave**
LEN: 256

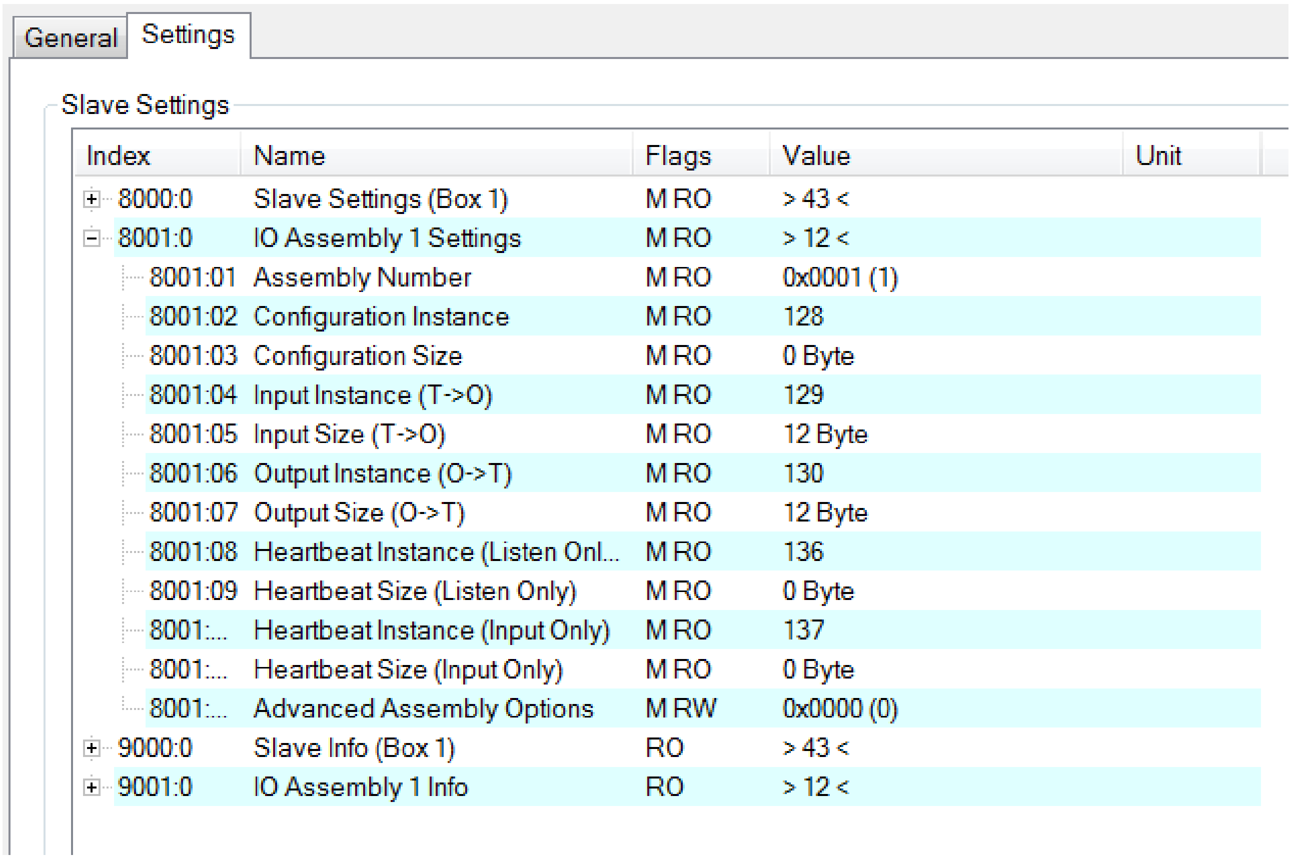
The data are stored in the data array, as described above -> see Object description [▶ 21].

5.3.4 Sample

A sample program can be downloaded: https://infosys.beckhoff.com/content/1033/TF6280_Tc3_EthernetIPSlave/Resources/3105211403/.tszip

5.4 Creating the EtherNet/IP slave in other EtherNet/IP masters

All the information you need is provided in the **Settings** dialog:



General | Settings

Slave Settings

Index	Name	Flags	Value	Unit
8000:0	Slave Settings (Box 1)	M RO	> 43 <	
8001:0	IO Assembly 1 Settings	M RO	> 12 <	
8001:01	Assembly Number	M RO	0x0001 (1)	
8001:02	Configuration Instance	M RO	128	
8001:03	Configuration Size	M RO	0 Byte	
8001:04	Input Instance (T->O)	M RO	129	
8001:05	Input Size (T->O)	M RO	12 Byte	
8001:06	Output Instance (O->T)	M RO	130	
8001:07	Output Size (O->T)	M RO	12 Byte	
8001:08	Heartbeat Instance (Listen Onl...	M RO	136	
8001:09	Heartbeat Size (Listen Only)	M RO	0 Byte	
8001:...	Heartbeat Instance (Input Only)	M RO	137	
8001:...	Heartbeat Size (Input Only)	M RO	0 Byte	
8001:...	Advanced Assembly Options	M RW	0x0000 (0)	
9000:0	Slave Info (Box 1)	RO	> 43 <	
9001:0	IO Assembly 1 Info	RO	> 12 <	

You need

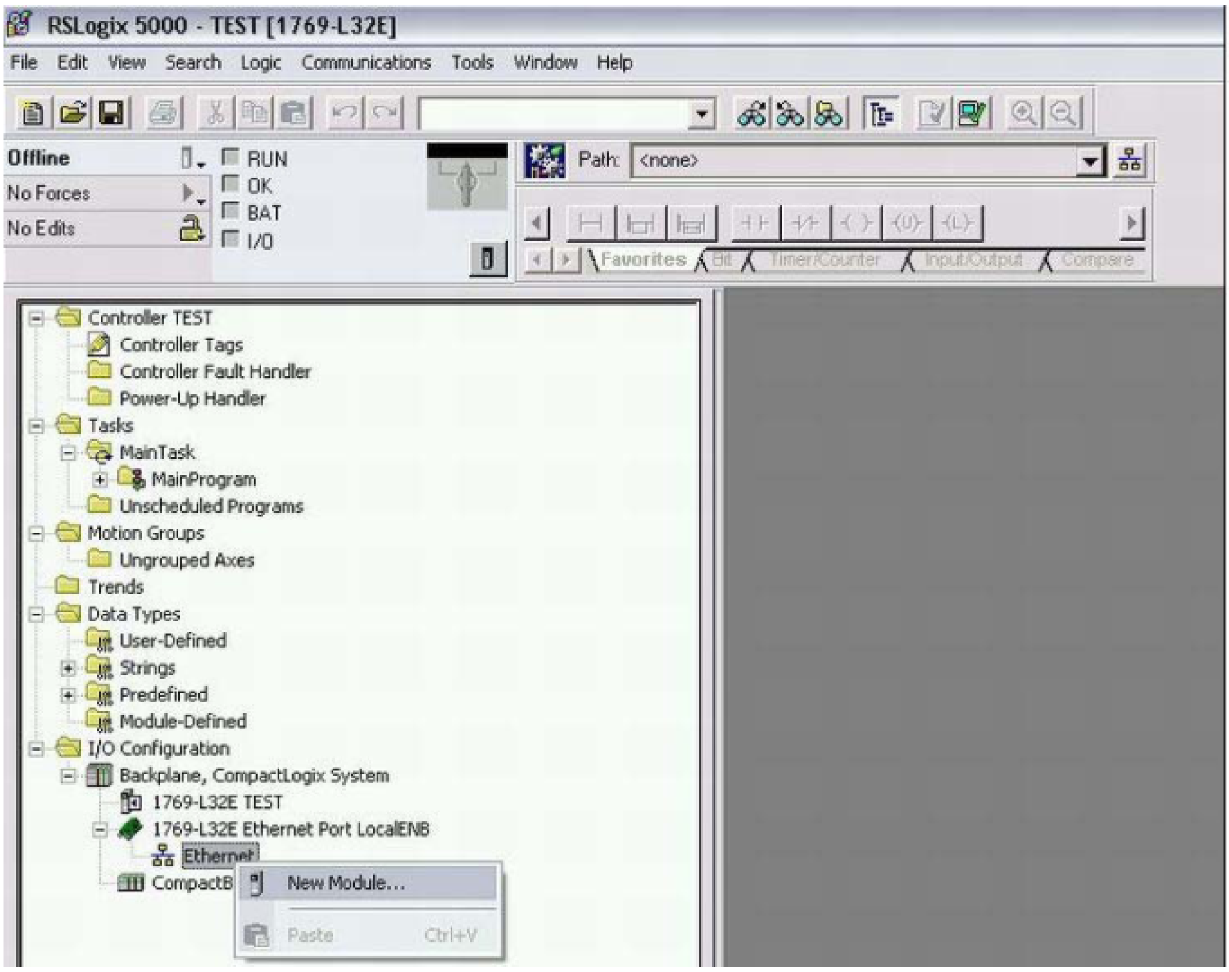
- the IP address of the adapter (see Creating an EtherNet/IP slave [▶ 15])
- the “Assembly Instance” numbers (see Settings tab)
- the number of data (see Settings tab)
- the “Configuration Instance” number 128 length 0
- the “Input Instance” number 129 length 12
- the “Output Instance”-number 130 length 12

The instance numbers are always the same. An export of the EDS file only contains the instance numbers. The number of data still has to be entered.

The EtherNet/IP device (slave) can be integrated via a “generic node” structure or via the EDS file.

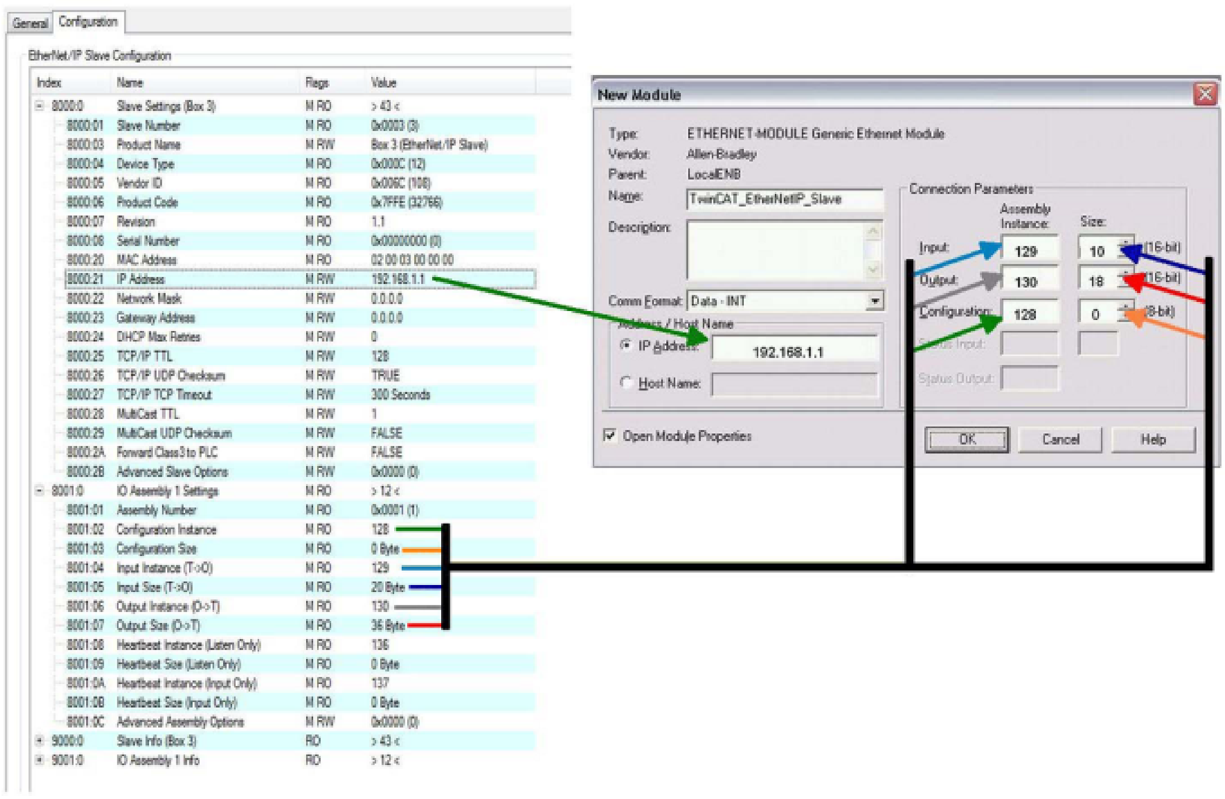
5.4.1 Sample for Rockwell CPUs

1. Under **Ethernet**, **New Module**…, select **Generic Ethernet Module**.



2. Enter the IP address from object `0x8000:21`.
3. Enter 129$_{dec}$ for Input Instance.
4. Enter 130$_{dec}$ for Output Instance and
5. 128$_{dec}$ for Config Instance.

⇨ The data length is dependent on the Comm format.



Index	Name	Flags	Value
8000:0	Slave Settings (Box 3)	M RO	> 43 <
8000:01	Slave Number	M RO	0x0003 (3)
8000:03	Product Name	M RW	Box 3 (EtherNet/IP Slave)
8000:04	Device Type	M RO	0x000C (12)
8000:05	Vendor ID	M RO	0x006C (108)
8000:06	Product Code	M RO	0x7FFE (32766)
8000:07	Revision	M RO	1.1
8000:08	Serial Number	M RO	0x00000000 (0)
8000:20	MAC Address	M RO	02 00 03 00 00 00
8000:21	IP Address	M RW	192.168.1.1
8000:22	Network Mask	M RW	0.0.0.0
8000:23	Gateway Address	M RW	0.0.0.0
8000:24	DHCP Max Retries	M RW	0
8000:25	TCP/IP TTL	M RW	128
8000:26	TCP/IP UDP Checksum	M RW	TRUE
8000:27	TCP/IP TCP Timeout	M RW	300 Seconds
8000:28	MultiCast TTL	M RW	1
8000:29	MultiCast UDP Checksum	M RW	FALSE
8000:2A	Forward Class3 to PLC	M RW	FALSE
8000:2B	Advanced Slave Options	M RW	0x0000 (0)
8001:0	IO Assembly 1 Settings	M RO	> 12 <
8001:01	Assembly Number	M RO	0x0001 (1)
8001:02	Configuration Instance	M RO	128
8001:03	Configuration Size	M RO	0 Byte
8001:04	Input Instance (T->O)	M RO	129
8001:05	Input Size (T->O)	M RO	20 Byte
8001:06	Output Instance (O->T)	M RO	130
8001:07	Output Size (O->T)	M RO	36 Byte
8001:08	Heartbeat Instance (Listen Only)	M RO	136
8001:09	Heartbeat Size (Listen Only)	M RO	0 Byte
8001:0A	Heartbeat Instance (Input Only)	M RO	137
8001:0B	Heartbeat Size (Input Only)	M RO	0 Byte
8001:0C	Advanced Assembly Options	M RW	0x0000 (0)
9000:0	Slave Info (Box 3)	RO	> 43 <
9001:0	IO Assembly 1 Info	RO	> 12 <

New Module

Type: ETHERNET-MODULE Generic Ethernet Module
Vendor: Allen-Bradley
Parent: LocalENB
Name: TwinCAT_EtherNetIP_Slave
Description:
Comm Format: Data - INT
Address / Host Name
IP Address: 192.168.1.1
Host Name:

Connection Parameters

	Assembly Instance:	Size:	
Input:	129	10	(16-bit)
Output:	130	18	(16-bit)
Configuration:	128	0	(8-bit)
Status Input:			
Status Output:			

☑ Open Module Properties

Note the properties of the selected Comm format

In the above sample the Comm format *INT* was selected, which means the number of data from objects `0x8001:05` and `0x8001:07` have to be divided by 2, since in TwinCAT they are specified in bytes and in the RSLogix in word length (INT).

An odd number of bytes must be rounded up. This also applies even if the Comm format is set to DINT, in which case you must round up to the next whole number.

System limitations

In the case of Multicast, pay attention to the high network loads that this causes, especially in systems with many or short cycle times. A high network load may possibly impair communication.

5.5 Acyclic communication

5.5.1 Common Industrial Protocol (CIP)

The Common Industrial Protocol (CIP) is an object-oriented peer-to-peer protocol that enables connections between industrial devices (sensors, actuators) and higher-level devices (controllers). CIP is independent of physical media and the data link layer. CIP has two main purposes: transport of control-oriented data connected to I/O devices, and transport of information relating to the system to be controlled, such as configuration parameters or diagnostics.

CIP uses abstract objects to describe a device. A CIP device consists of a group of objects. Objects describe the available communication services, the externally visible behavior of the device, and a way in which information can be retrieved and exchanged. CIP objects are divided into classes, instances and attributes. A class is a set of objects that all represent the same component. An instance is the current representation of a particular object. Each instance has the same attributes, but possibly with different attribute values. The individual objects are addressed via a node address, which for EtherNet/IP is the IP address, plus a class, instance and attributes.

- Object

    - An abstract representation of a particular component within a product.
- Class
    - A set of objects that all represent the same type of system component. A class is a generalization of an object. All objects in a class are identical in form and behavior, but can contain different attribute values.
- Instance
    - A specific and real specimen of an object.
      Example: Berlin is an instance of the Capital object class.
- Attribute
    - A description of a property or characteristic of an object. Typically, attributes provide status information or control the operation of an object.

(Source: The CIP Networks Library Volume 1: Common Industrial Protocol, Edition 3.22)

The following objects are used internally by Beckhoff and are therefore reserved:

1. Identity Object → Class 0x1
2. Message Router Object → Class 0x2
3. Assembly Object → Class 0x4
4. Connection Manager Object → Class 0x6
5. TCP/IP Interface Object → Class 0xF5
6. Ethernet Link Object → Class 0xF6

5.5.2 Forward Message to AMS Port

"Explicit Messaging" is used to send information and data that does not require continuous updates. "Explicit Messaging" allows you to configure and monitor the parameters of a slave device in the Ethernet/IP network.

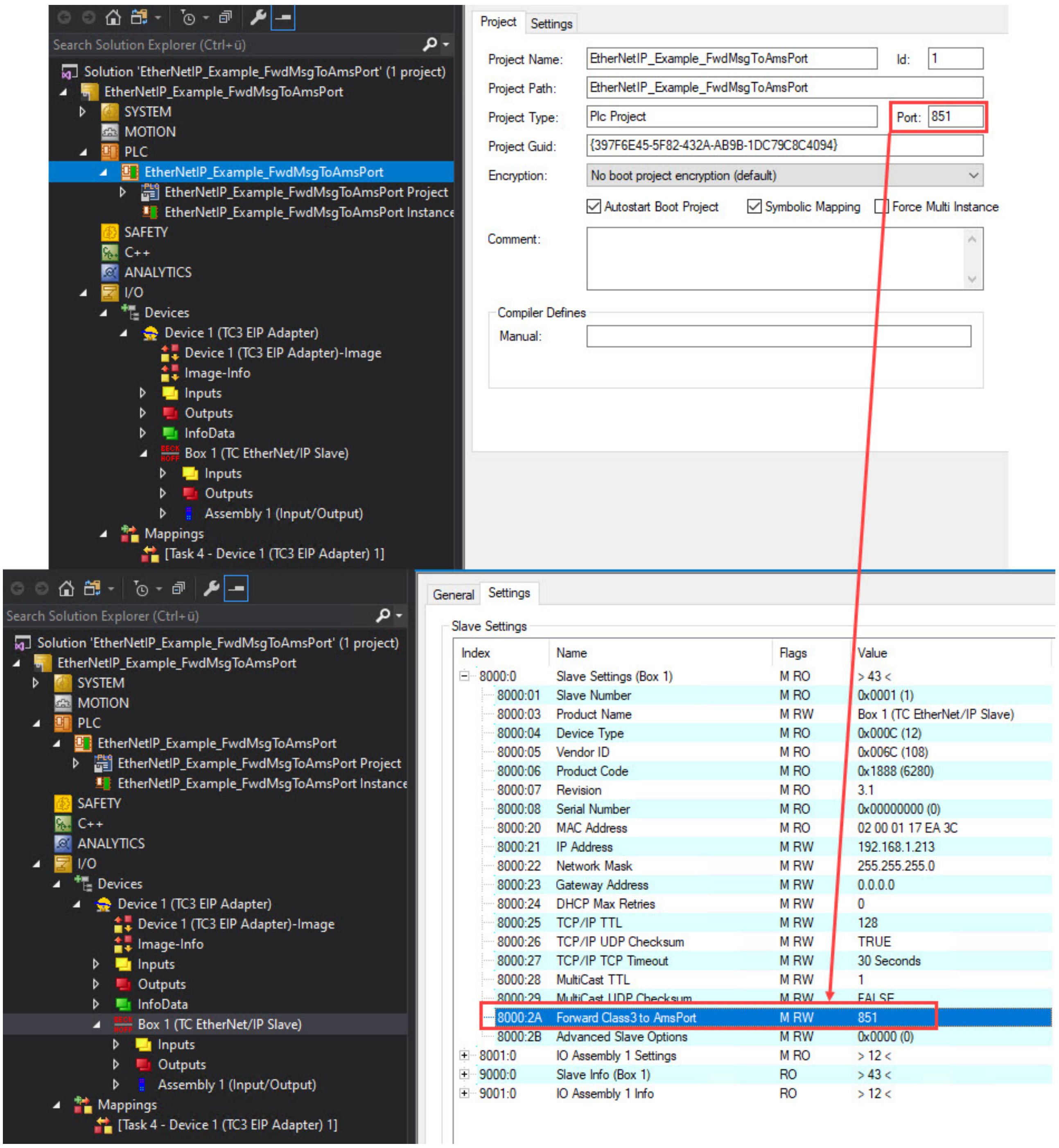
The "FwdMsgToAmsPort" feature allows acyclic requests from Ethernet/IP scanners to be processed.

The following example shows implementation of acyclic communication between a TwinCAT 3 controller and an RS Logix controller.

TwinCAT 3 implementation:

✓ Requirement: Ethernet/IP driver version, min. V1.23

1. To enable the FwdMsgToAmsPort feature, enter the AmsPort of the PLC (in the example 851) in the slave/master settings (0x8000:2A/0xF800:2A) in TwinCAT.



Solution 'EtherNetIP_Example_FwdMsgToAmsPort' (1 project)
- EtherNetIP_Example_FwdMsgToAmsPort
  - SYSTEM
  - MOTION
  - PLC
    - EtherNetIP_Example_FwdMsgToAmsPort
      - EtherNetIP_Example_FwdMsgToAmsPort Project
      - EtherNetIP_Example_FwdMsgToAmsPort Instance
  - SAFETY
  - C++
  - ANALYTICS
  - I/O
    - Devices
      - Device 1 (TC3 EIP Adapter)
        - Device 1 (TC3 EIP Adapter)-Image
        - Image-Info
        - Inputs
        - Outputs
        - InfoData
        - Box 1 (TC EtherNet/IP Slave)
          - Inputs
          - Outputs
          - Assembly 1 (Input/Output)
    - Mappings
      - [Task 4 - Device 1 (TC3 EIP Adapter) 1]

Project | Settings

Project Name:	EtherNetIP_Example_FwdMsgToAmsPort
Id:	1
Project Path:	EtherNetIP_Example_FwdMsgToAmsPort
Project Type:	Plc Project
Port:	851
Project Guid:	{397F6E45-5F82-432A-AB9B-1DC79C8C4094}
Encryption:	No boot project encryption (default)
Autostart Boot Project	✓
Symbolic Mapping	✓
Force Multi Instance	
Comment:	
Compiler Defines – Manual:	

General | Settings

Slave Settings

Index	Name	Flags	Value
8000:0	Slave Settings (Box 1)	M RO	> 43 <
8000:01	Slave Number	M RO	0x0001 (1)
8000:03	Product Name	M RW	Box 1 (TC EtherNet/IP Slave)
8000:04	Device Type	M RO	0x000C (12)
8000:05	Vendor ID	M RO	0x006C (108)
8000:06	Product Code	M RO	0x1888 (6280)
8000:07	Revision	M RO	3.1
8000:08	Serial Number	M RO	0x00000000 (0)
8000:20	MAC Address	M RO	02 00 01 17 EA 3C
8000:21	IP Address	M RW	192.168.1.213
8000:22	Network Mask	M RW	255.255.255.0
8000:23	Gateway Address	M RW	0.0.0.0
8000:24	DHCP Max Retries	M RW	0
8000:25	TCP/IP TTL	M RW	128
8000:26	TCP/IP UDP Checksum	M RW	TRUE
8000:27	TCP/IP TCP Timeout	M RW	30 Seconds
8000:28	MultiCast TTL	M RW	1
8000:29	MultiCast UDP Checksum	M RW	FALSE
8000:2A	Forward Class3 to AmsPort	M RW	851
8000:2B	Advanced Slave Options	M RW	0x0000 (0)
8001:0	IO Assembly 1 Settings	M RO	> 12 <
9000:0	Slave Info (Box 1)	RO	> 43 <
9001:0	IO Assembly 1 Info	RO	> 12 <

2. ADSRDWRT requests from the Ethernet/IP driver (IDGRP: 0x848180E9 IOFFS: SlaveId (Adapter)) to the PLC task are registered as indications and this enables them to be processed. The ADSRDWRTIND function block is used for this purpose.

   ⇨ The first entry in the indication registered by the Ethernet/IP driver is a 32-byte (8xULONG) header:

```
TYPE DUT_MsgToAmsPortHeader:
STRUCT
    nServiceCode:UDINT;
    nClassId:UDINT;
    nInstanceId:UDINT;
    nAttributeId:UDINT;
    nReservedId:UDINT;
    nGeneralStatus:UDINT;
    nAdditionalStatus:UDINT;
    nDataLen:UDINT;
END_STRUCT
END_TYPE
```

```
TYPE DUT_IncomingMsgRequest:
STRUCT
    reqHdr:DUT_MsgtoAmsPortHeader;
    reqData:ARRAY [0…991] OF BYTE;
END_STRUCT
END_TYPE
```

```
TYPE DUT_OutgoingMsgResponse:
STRUCT
    resHdr:DUT_magToAmsPortHeader;
    resData:ARRAY [0…9991] OF BYTE;
End_Struct
END_TYPE
```

The same header is also used for the response.

3. The actual read/write data follows directly after the header (nDataLen <> 0 should be set according to the data length). The maximum supported data length is 992 bytes (+ 32-byte header = 1024 bytes).Potential classes/instances/attribute values

	Min	Max
Class	1	0xFFFF
Instance	1	0xFFFF
Attribute	1	0xFFFF

4. After an indication has been processed, a response must be sent to the source device via the ADSRDWRTRES function block.

```
PROGRAM MAIN
VAR
    i                   : INT;
    IdxGroup            : UDINT;        //Ethernet/IP-Treiber -> 16#848180E9
    IdxOffset           : UDINT;        //SlaveId (Adapter) bzw. 0xFFFF (Scanner)
    fbADSRDWRTINDEX : ADSRDWRTINDEX;
    fbAdsRdWrRes        : ADSRDWRTRES;
    request             : DUT_IncomingMsgRequest;
    response            : DUT_OutgoingMsgResponse;
    nResponseLen        : UINT;
    nAdsResult          : UDINT:=0;
    nAdsResponsesSent   : UDINT;
    Attributes          : ARRAY [1..4] OF STRING :=['TestReadOnlyAttribute1','TestReadOnlyAttrib
ute2','TestReadOnlyAttribute3','TestReadWriteAttribute4'];
END_VAR
```

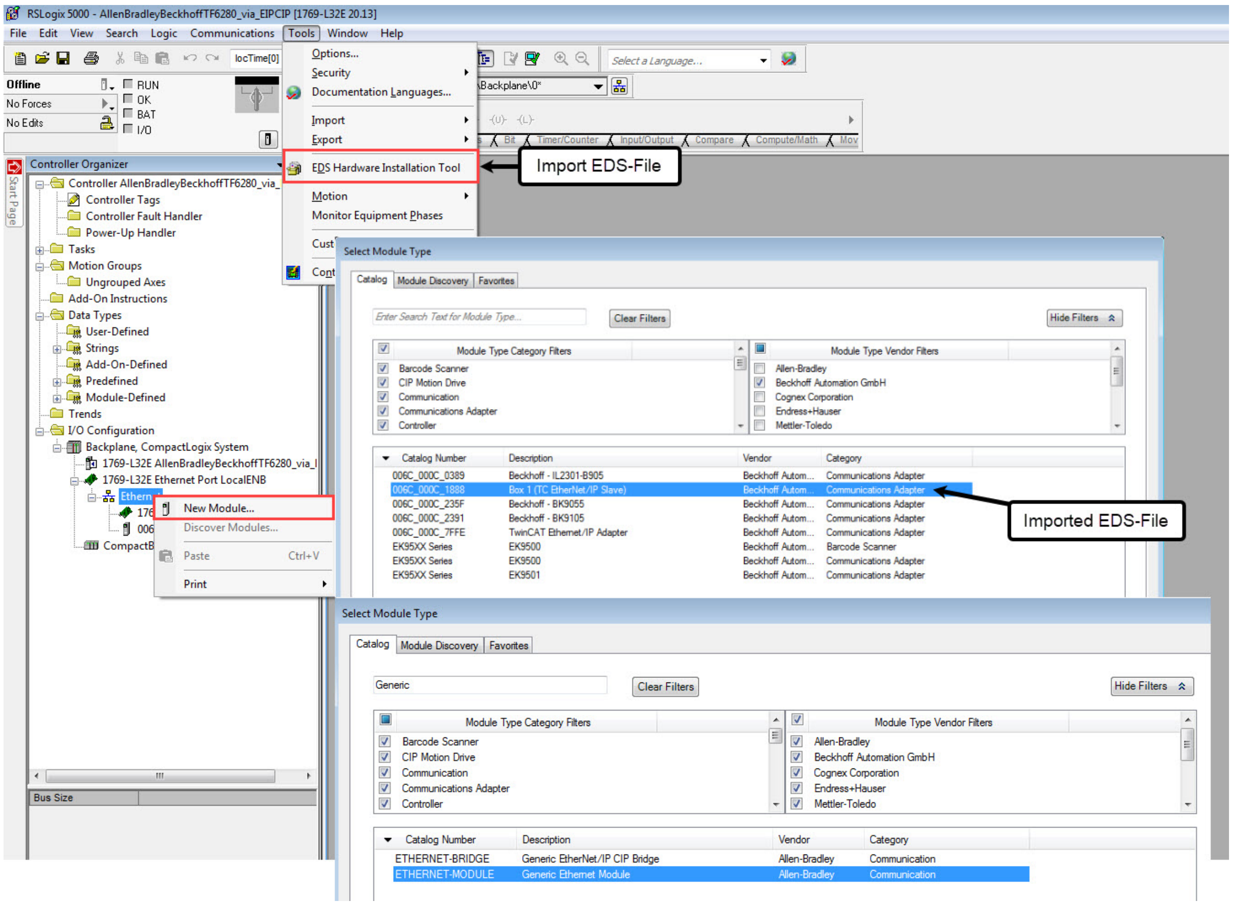
```

CASE i OF
0:    //check for ADSReadWrite-Requests
fbADSRDWRTINDEX (
CLEAR:=FALSE ,
MINIDXGRP:= 16#84000000,
VALID=> ,
NETID=> ,
PORT=> ,
INVOKEID=> ,
IDXGRP=> ,
IDXOFFS=> ,
);

IF fbADSRDWRTINDEX.VALID THEN
IdxGroup:= fbADSRDWRTINDEX.IDXGRP;
IdxOffset:= fbADSRDWRTINDEX.IDXOFFS ;
      MEMSET(ADR(request), 0, SIZEOF(request));
      MEMSET(ADR(response), 0, SIZEOF(response));
      nResponseLen:=0;
      //check for Indication Request = Ethernet/IP-driver -> 16#848180E9
      IF IdxGroup = 16#848180E9 THEN
          //check for Indication.datalength >= DUT_MsgToAmsPortHeader
          IF fbADSRDWRTINDEX.WRTLENGTH >= SIZEOF(request.reqHdr) THEN
              MEMCPY(ADR(request.reqHdr), fbADSRDWRTINDEX.DATAADDR, SIZEOF(request.reqHdr));
          END_IF
          //check for Indication.datalength > DUT_MsgToAmsPortHeader >>> save additional data
          IF fbADSRDWRTINDEX.WRTLENGTH > SIZEOF(request.reqHdr) THEN
              MEMCPY(ADR(request.reqData), fbADSRDWRTINDEX.DATAADDR+SIZEOF(request.reqHdr), fbADSRDW
RTINDEX.WRTLENGTH-SIZEOF(request.reqHdr));
          END_IF
          i:=10;
      ELSE
          i:=20;
      END_IF
END_IF
```

```
10:    //new Ind from EthIp-Drv received
  response.resHdr.nServiceCode := request.reqHdr.nServiceCode OR CONST.CN_SC_REPLY_MASK;
  response.resHdr.nGeneralStatus := 0;
  response.resHdr.nAdditionalStatus := 0;
  response.resHdr.nDataLen := 0;
  IF request.reqHdr.nServiceCode = CONST.CN_SC_GET_ATTR_SINGLE OR request.reqHdr.nServiceCode = CONS
T.CN_SC_SET_ATTR_SINGLE THEN
      i:=11;
  ELSE
      response.resHdr.nGeneralStatus := CONST.CN_GRC_BAD_SERVICE;
      nResponseLen := SIZEOF(response.resHdr);
      i:=20;
  END_IF

11:    //case decision for request
  CASE request.reqHdr.nClassId OF
  16#1000:    //erlaubte Beispiel Class 0x10000
          CASE request.reqHdr.nInstanceId OF
          16#1:    //erlaubte Beispiel Instance 0x1
                  CASE request.reqHdr.nAttributeId OF    //
Attributes 1-4 erlaubt; only attr 4 is settable
                  1,2,3:    IF request.reqHdr.nServiceCode = CONST.CN_SC_SET_ATTR_SINGLE THEN
                               response.resHdr.nGeneralStatus := CONST.CN_GRC_ATTR_NOT_SETTABLE;
                               nResponseLen := SIZEOF(response.resHdr);
                               i:=20;
                          ELSE
                               i:=12;
                          END_IF
                  4:        IF request.reqHdr.nServiceCode = CONST.CN_SC_SET_ATTR_SINGLE THEN
                              i:=14;
                          ELSE
                              i:=12;
                          END_IF
                  ELSE
                      response.resHdr.nGeneralStatus := CONST.CN_GRC_UNDEFINED_ATTR;
                      nResponseLen := SIZEOF(response.resHdr);
                      i:=20;
                  END_CASE
          ELSE
              response.resHdr.nGeneralStatus := CONST.CN_GRC_BAD_CLASS_INSTANCE;
              nResponseLen := SIZEOF(response.resHdr);
              i:=20;
          END_CASE
  ELSE
      response.resHdr.nGeneralStatus := CONST.CN_GRC_BAD_CLASS_INSTANCE;
      nResponseLen := SIZEOF(response.resHdr);
      i:=20;
  END_CASE

12:    //GetAttribute
  response.resHdr.nGeneralStatus         := CONST.CN_GRC_SUCCESS;
  MEMCPY(ADR(response.resData), ADR(Attributes[request.reqHdr.nAttributeId]), SIZEOF(Attributes[requ
est.reqHdr.nAttributeId]));
  response.resHdr.nDataLen := INT_TO_UINT(LEN(Attributes[request.reqHdr.nAttributeId]));
  nResponseLen := UDINT_TO_UINT(response.resHdr.nDataLen) + SIZEOF(response.resHdr);
  i:=20;

14:    //SetAttribute
  response.resHdr.nGeneralStatus         := CONST.CN_GRC_SUCCESS;
  IF request.reqHdr.nDataLen <= SIZEOF(STRING)-1 THEN
      MEMCPY(ADR(Attributes[request.reqHdr.nAttributeId]), ADR(request.reqData), request.reqHdr.nDat
aLen);
  ELSE
      response.resHdr.nGeneralStatus     := CONST.CN_GRC_BAD_DATA;
  END_IF
  nResponseLen := SIZEOF(response.resHdr);
  i:=20;

20:    //response to Ethernet/IP-driver
   fbAdsRdWrRes(
NETID:= fbADSRDWRTINDEX.NETID ,
PORT:= fbADSRDWRTINDEX.PORT ,
INVOKEID:= fbADSRDWRTINDEX.INVOKEID ,
RESULT:=nAdsResult ,
LEN:=nResponseLen,
DATAADDR:=ADR(Response) ,
RESPOND:=TRUE );
i:=21;
      nAdsResponsesSent:=nAdsResponsesSent+1;
fbADSRDWRTINDEX (CLEAR:=TRUE);
```

```
21: i:=0;
fbAdsRdWrRes(RESPOND:=FALSE);
END_CASE
```

Implementation RS Logix 5000:

1. At the beginning you have to create a new module, either a "Generic Ethernet Module" or an EDS file exported from TwinCAT.
   The advantage of the imported EDS file is that it already contains the size of the process data created in the TwinCAT configuration.

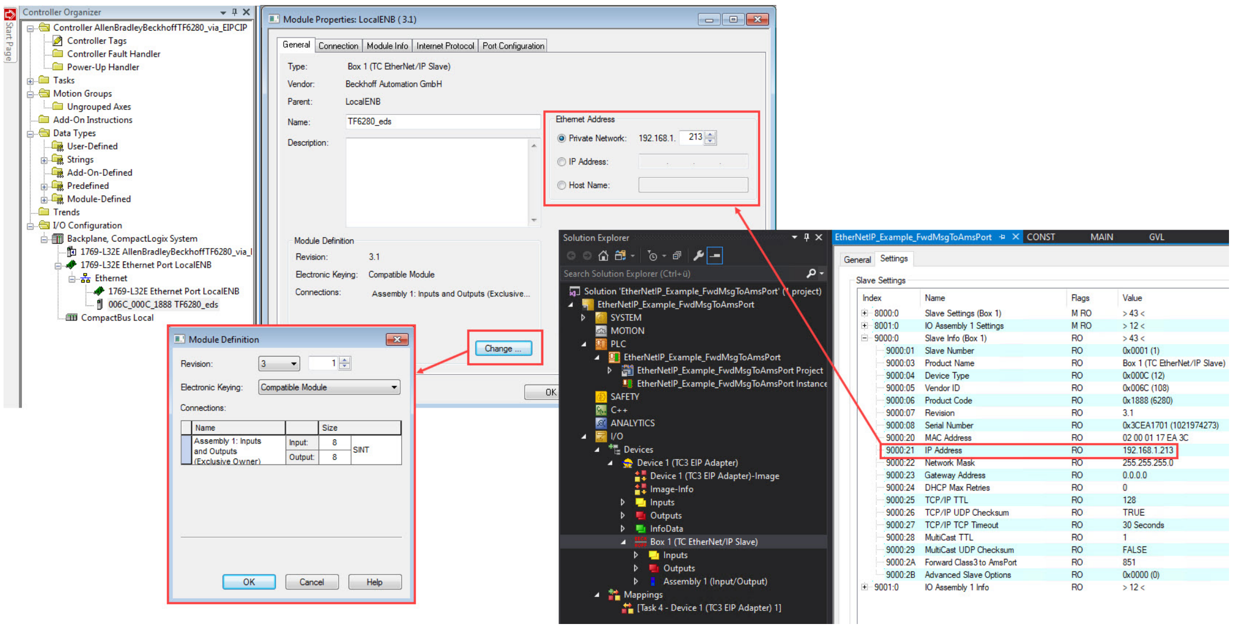


Catalog Number	Description	Vendor	Category
006C_000C_0389	Beckhoff - IL2301-B905	Beckhoff Autom...	Communications Adapter
006C_000C_1888	Box 1 (TC EtherNet/IP Slave)	Beckhoff Autom...	Communications Adapter
006C_000C_235F	Beckhoff - BK9055	Beckhoff Autom...	Communications Adapter
006C_000C_2391	Beckhoff - BK9105	Beckhoff Autom...	Communications Adapter
006C_000C_7FFE	TwinCAT Ethernet/IP Adapter	Beckhoff Autom...	Communications Adapter
EK95XX Series	EK9500	Beckhoff Autom...	Barcode Scanner
EK95XX Series	EK9500	Beckhoff Autom...	Communications Adapter
EK95XX Series	EK9501	Beckhoff Autom...	Communications Adapter

Generic

Catalog Number	Description	Vendor	Category
ETHERNET-BRIDGE	Generic EtherNet/IP CIP Bridge	Allen-Bradley	Communication
ETHERNET-MODULE	Generic Ethernet Module	Allen-Bradley	Communication

2. In the settings of the attached module you may have to adjust the IP and the process data settings.



Module Properties: LocalENB (3.1)

Type: Box 1 (TC EtherNet/IP Slave)
Vendor: Beckhoff Automation GmbH
Parent: LocalENB
Name: TF6280_eds
Description:

Ethernet Address
Private Network: 192.168.1. 213

Module Definition
Revision: 3.1
Electronic Keying: Compatible Module
Connections: Assembly 1: Inputs and Outputs (Exclusive...

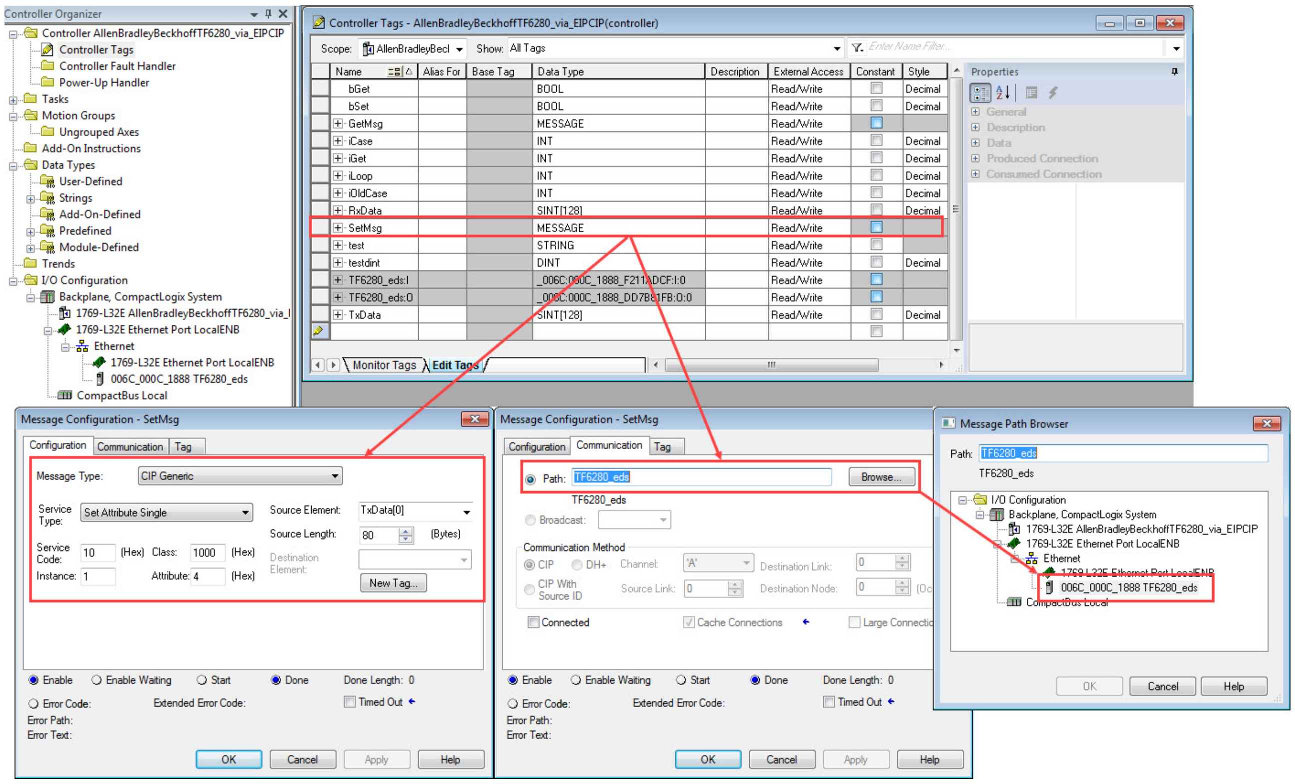
Module Definition
Revision: 3 1
Electronic Keying: Compatible Module

Name		Size	
Assembly 1: Inputs and Outputs (Exclusive Owner)	Input:	8	SINT
	Output:	8	

EtherNetIP_Example_FwdMsgToAmsPort — Settings — Slave Settings

Index	Name	Flags	Value
8000:0	Slave Settings (Box 1)	M RO	> 43 <
8001:0	IO Assembly 1 Settings	M RO	> 12 <
9000:0	Slave Info (Box 1)	RO	> 43 <
9000:01	Slave Number	RO	0x0001 (1)
9000:03	Product Name	RO	Box 1 (TC EtherNet/IP Slave)
9000:04	Device Type	RO	0x000C (12)
9000:05	Vendor ID	RO	0x006C (108)
9000:06	Product Code	RO	0x1888 (6280)
9000:07	Revision	RO	3.1
9000:08	Serial Number	RO	0x3CEA1701 (1021974273)
9000:20	MAC Address	RO	02 00 01 17 EA 3C
9000:21	IP Address	RO	192.168.1.213
9000:22	Network Mask	RO	255.255.255.0
9000:23	Gateway Address	RO	0.0.0.0
9000:24	DHCP Max Retries	RO	0
9000:25	TCP/IP TTL	RO	128
9000:26	TCP/IP UDP Checksum	RO	TRUE
9000:27	TCP/IP TCP Timeout	RO	30 Seconds
9000:28	MultiCast TTL	RO	1
9000:29	MultiCast UDP Checksum	RO	FALSE
9000:2A	Forward Class3 to AmsPort	RO	851
9000:2B	Advanced Slave Options	RO	0x0000 (0)
9001:0	IO Assembly 1 Info	RO	> 12 <

3. To be able to send and receive messages acyclically, structures of the type "Messages" are necessary. In the sample, one structure is used for sending and one for receiving. You must configure both structures accordingly for sending and for receiving.
4. Right-click on the tag **SetMsg-Configure SetMsg** to open the configuration settings. These are to be taken over as indicated in the screenshot.
   The specifications **Class**, **Instance** and **Attribute** are freely selectable. At **Service Type** set **Set Attributes Single**. At **Source Element**, create an array whose contents are to be sent. Select the **Source Length** so that it does not exceed the length of the target variable created in TwinCAT.



Controller Tags - AllenBradleyBeckhoffTF6280_via_EIPCIP(controller)

Scope: AllenBradleyBecl Show: All Tags

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
bGet			BOOL		Read/Write	☐	Decimal
bSet			BOOL		Read/Write	☐	Decimal
GetMsg			MESSAGE		Read/Write	☐	
iCase			INT		Read/Write	☐	Decimal
iGet			INT		Read/Write	☐	Decimal
iLoop			INT		Read/Write	☐	Decimal
iOldCase			INT		Read/Write	☐	Decimal
RxData			SINT[128]		Read/Write	☐	Decimal
SetMsg			MESSAGE		Read/Write	☐	
test			STRING		Read/Write	☐	
testdint			DINT		Read/Write	☐	Decimal
TF6280_eds:I			_006C:000C_1888_F211ADCF:I:0		Read/Write	☐	
TF6280_eds:O			_006C:000C_1888_DD7B81FB:O:0		Read/Write	☐	
TxData			SINT[128]		Read/Write	☐	Decimal

Message Configuration - SetMsg (Configuration)

Message Type: CIP Generic
Service Type: Set Attribute Single
Service Code: 10 (Hex) Class: 1000 (Hex)
Instance: 1 Attribute: 4 (Hex)
Source Element: TxData[0]
Source Length: 80 (Bytes)
Destination Element:
Enable Enable Waiting Start Done Done Length: 0
Error Code: Extended Error Code: Timed Out
Error Path:
Error Text:

Message Configuration - SetMsg (Communication)

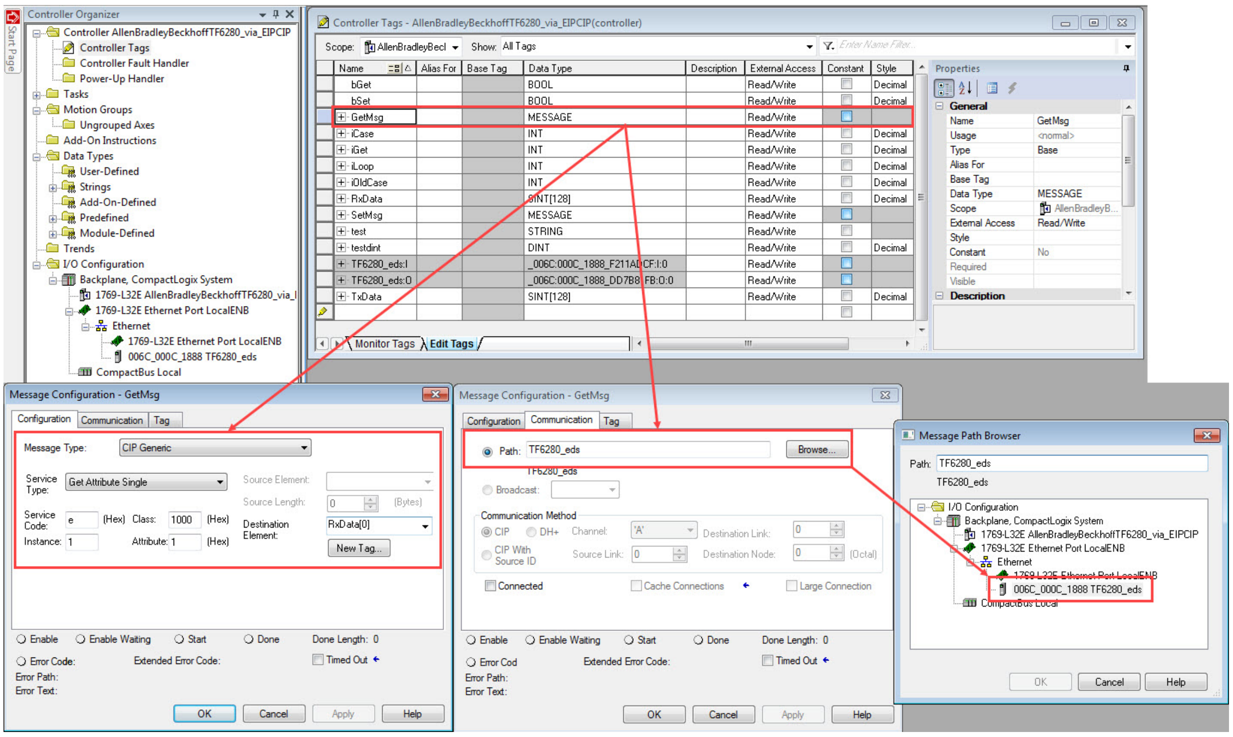
Path: TF6280_eds
TF6280_eds
Broadcast:
Communication Method: CIP DH+ Channel: 'A' Destination Link: 0
CIP With Source ID Source Link: 0 Destination Node: 0
Connected Cache Connections ✓ Large Connection

Message Path Browser

Path: TF6280_eds
TF6280_eds
I/O Configuration
Backplane, CompactLogix System
1769-L32E AllenBradleyBeckhoffTF6280_via_EIPCIP
1769-L32E Ethernet Port LocalENB
Ethernet
1769-L32E Ethernet Port LocalENB
006C_000C_1888 TF6280_eds
CompactBus Local

5. Right-click on the tag **GetMsg - Configure GetMsg** to open the configuration settings. These are to be taken over as indicated in the screenshot.
   The specifications **Class**, **Instance** and **Attribute** are freely selectable. At **Service Type** set **Get**

Attribute Single. Create an array at **Destination Element** that receives the acyclic messages. The size of the array is to be chosen according to the receiving messages.



Controller Tags - AllenBradleyBeckhoffTF6280_via_EIPCIP(controller)

Scope: AllenBradleyBecl Show: All Tags

Name	Alias For	Base Tag	Data Type	Description	External Access	Constant	Style
bGet			BOOL		Read/Write	☐	Decimal
bSet			BOOL		Read/Write	☐	Decimal
GetMsg			MESSAGE		Read/Write	☐	
iCase			INT		Read/Write	☐	Decimal
iGet			INT		Read/Write	☐	Decimal
iLoop			INT		Read/Write	☐	Decimal
iOldCase			INT		Read/Write	☐	Decimal
RxData			SINT[128]		Read/Write	☐	Decimal
SetMsg			MESSAGE		Read/Write	☐	
test			STRING		Read/Write	☐	
testdint			DINT		Read/Write	☐	Decimal
TF6280_eds:I			_006C:000C_1888_F211ADCF:I:0		Read/Write	☐	
TF6280_eds:O			_006C:000C_1888_DD7B8 FB:O:0		Read/Write	☐	
TxData			SINT[128]		Read/Write	☐	Decimal

Message Configuration - GetMsg (Configuration)

Message Type: CIP Generic
Service Type: Get Attribute Single
Service Code: e (Hex) Class: 1000 (Hex)
Instance: 1 Attribute: 1 (Hex)
Source Element:
Source Length: 0 (Bytes)
Destination Element: RxData[0]

Message Configuration - GetMsg (Communication)

Path: TF6280_eds
Communication Method: CIP

Message Path Browser

Path: TF6280_eds

⇨ The following sample code sends requests to the Ethernet/IP driver of the TF6280, which forwards them to the TwinCAT PLC for further processing.

A single attribute value is read from the TwinCAT PLC with a positive edge at "bGet". In this sample the values "TestReadOnlyAttribute1, TestReadOnlyAttribute2 and TestReadOnlyAttribute3" can be read.
A single attribute value is written to the TwinCAT PLC with a positive edge at "bSet". In this sample the fourth attribute in the TwinCAT PLC can be described with the content "123Beckhoff567" and "HelloBeckhoff".

```
//GetAttribute
IF bGet THEN
    bGet:=0;
    iCase:=20+iGet;
END_IF;

//SetAttribute
IF bSet AND iOldCase=5 THEN
    bSet:=0;
    iCase:=6;
ELSIF bSet AND iOldCase=6 THEN
    bSet:=0;
    iCase:=5;
END_IF;

CASE iCase OF
5:    //HelloBeckhoff --> (ASCII)
    iOldCase:=iCase;

    TxData[0]:=72;              //H
    TxData[1]:=101;              //e
    TxData[2]:=108;              //l
    TxData[3]:=108;              //l
    TxData[4]:=111;              //o
    TxData[5]:=66;              //B
    TxData[6]:=101;              //e
    TxData[7]:=99;              //c
    TxData[8]:=107;              //k
    TxData[9]:=104;              //h
    TxData[10]:=111;          //o
    TxData[11]:=102;          //f
    TxData[12]:=102;          //f
    iCase:=10;

6:    //123Beckhoff567 --> (ASCII)
    iOldCase:=iCase;
    TxData[0]:=49;             //1
    TxData[1]:=50;             //2
    TxData[2]:=51;             //3
    TxData[3]:=66;             //B
    TxData[4]:=101;             //e
    TxData[5]:=99;             //c
    TxData[6]:=107;             //k
    TxData[7]:=104;             //h
    TxData[8]:=111;             //o
    TxData[9]:=102;             //f
    TxData[10]:=102;         //f
    TxData[11]:=52;             //4
    TxData[12]:=53;             //5
    TxData[13]:=54;             //6
    iCase:=10;

10:    //SetAttribute
    msg(SetMsg);
    IF SetMsg.DN OR SetMsg.ER THEN
        FOR iLoop:=0 TO 80 DO
            TxData[iLoop]:=0;
        end_FOR;
        iCase:=0;
    END_IF;

20:    //TestReadOnlyAttribute1
    GetMsg.Class:=16#1000;
    GetMsg.Instance:=16#01;
    GetMsg.Attribute:=16#01;
    iCase:=30;


21:    //TestReadOnlyAttribute2
    GetMsg.Class:=16#1000;
    GetMsg.Instance:=16#01;
    GetMsg.Attribute:=16#02;
    iCase:=30;


22:    //TestReadOnlyAttribute3
    GetMsg.Class:=16#1000;
    GetMsg.Instance:=16#01;
    GetMsg.Attribute:=16#03;
    iCase:=30;

30:    //GetAttribue
    msg(GetMsg);
    IF GetMsg.DN OR GetMsg.ER then
        iGet:=iGet+1;
        IF iGet >= 3 THEN
            iGet:=0;
        END_IF;
        iCase:=0;
    END_IF;

END_CASE;
```

You can find the documented example as a TwinCAT project here: https://infosys.beckhoff.com/content/1033/TF6280_Tc3_EthernetIPSlave/Resources/14758092427/.zip.

6 Properties

6.1 Virtual slave

Using the TF6280, up to eight slaves can be parameterized with a physical interface. In this case a virtual MAC address is formed for each virtual slave device, so that up to eight EtherNet/IP slaves can be operated on a PC via an Ethernet interface.

The advantage is that this option enables convenient connection of eight EtherNet/IP controllers and limitations in the bus communication with the slave can be bypassed without using additional hardware.

This feature can be used, for sample, for exchanging large data quantities with an EtherNet/IP master or for connecting with several EtherNet/IP masters in different subnets.

Create an additional box in the TwinCAT system configuration and proceed in the same way as for the configuration of a real slave.

Unique MAC address

If the virtual MAC address is assigned manually, ensure that it is truly unique in your network.

6.2 TF6280 - Configuration parameters

6.2.1 Index 0x8000 Slave Settings

Index	Name	Meaning
8000:0	Adapter Settings	
8000:1	Adapter Number	Adapter Box ID
8000:3	Product Name	Name of the device
8000:4	Device Type	Device type
8000:5	Vendor ID	Vendor number
8000:6	Product Code	Product code
8000:7	Revision	Version
8000:8	Serial Number	Serial number (see object 0x9000)
8000:20	MAC Address	MAC address (see object 0x9000)
8000:21	IP Address	IP address • `0.0.0.0`: Will be assigned dynamically by the DHCP service • `255.255.255.255`: The operating system address is used Otherwise: statically assigned IP address
8000:22	Network Mask	Subnet mask • `0.0.0.0`: Will be assigned dynamically by the DHCP service Otherwise: statically assigned subnet mask
8000:23	Gateway address	Gateway address • `0.0.0.0`: Will be assigned dynamically by the DHCP service Otherwise: statically assigned gateway address
8000:24	DHCP Max Retries	0: Continuous repetition of the DHCP addressing attempts. (Currently only this mode is implemented, as of: 10-2016)
8000:25	TCP/IP TTL	“Time to live” – value for unicast TCP/UDP communication
8000:26	TCP/IP UDP Checksum	Checksum function (Unicast): • 0: UDP checksum disabled. • 1: UDP checksum enabled
8000:27	TCP/IP TCP Timeout	Time switch for inactive TCP connection in seconds • 0: Time switch disabled
8000:28	Multicast TTL	“Time to live” value for multicast UDP communication
8000:29	Multicast UDP checksum	Checksum function (Multicast): • 0: UDP checksum disabled • 1: UDP checksum enabled
8000:2A	Forward Class3 to PLC	Message forwarding to the PLC (Currently not implemented, as of: 10-2016)
8000:2B	Advanced adapter options	“Store Category” parameter • Bit9=Cat2, • Bit8=Cat1 see Writing the IP address from the PLC [▶ 19]

6.2.2 Index 0x8001 IO Assembly Settings

Index	Name	Meaning
8001:0	IO Assembly Settings	
8001:1	Assembly Number	Assembly Id
8001:1	Configuration Instance	Configuration instance
8001:3	Configuration Size	Configuration size (always 0)
8001:4	Input Instance (T->O)	Link point for input values (T->O: Target->Originator)
8001:5	Input Size (T->O)	Size of the input values (in bytes)
8001:6	Output Instance (O->T)	Link point for output values (O->T, Originator->Target)
8001:7	Output Size (O-T)	Size of the output values (in bytes)
8001:8	Heartbeat Instance (Listen Only)*	Heartbeat link point (only for monitoring connections)
8001:9	Heartbeat Size (Listen Only)*	always 0
8001:A	Heartbeat Instance (Input Only)**	Heartbeat link point (only for input connections)
8001:B	Heartbeat Size (Input Only)**	always 0
8001:C	Advanced Assembly Options	Bit 14: `0x4000` hex • 0 = default • 1 = disables the link between “ConnCtrl“ and ”ConnState“ for the EtherNet/IP IO connection The other bits are always set to 0 (reserved)

* Heartbeat Instance (Listen Only): Enables monitoring of the input data (output data for TF6280) if a connection exists. The "Listen Only" connection is also terminated when the normal connection is terminated.

** Heartbeat Instance (Input Only): Enables reading of the input data (output data for TF6280). This connection is independent of the actual communication.

The heartbeat is necessary for the monitoring of both connection types (Listen Only and Input Only).

6.2.3 Index 0x9000 Adapter Info

The current valid settings are displayed here; these can differ from the object `0x8000`. The object `0x9000` displays the active parameters.

6.2.4 Index 0x9001 IO Assembly Info

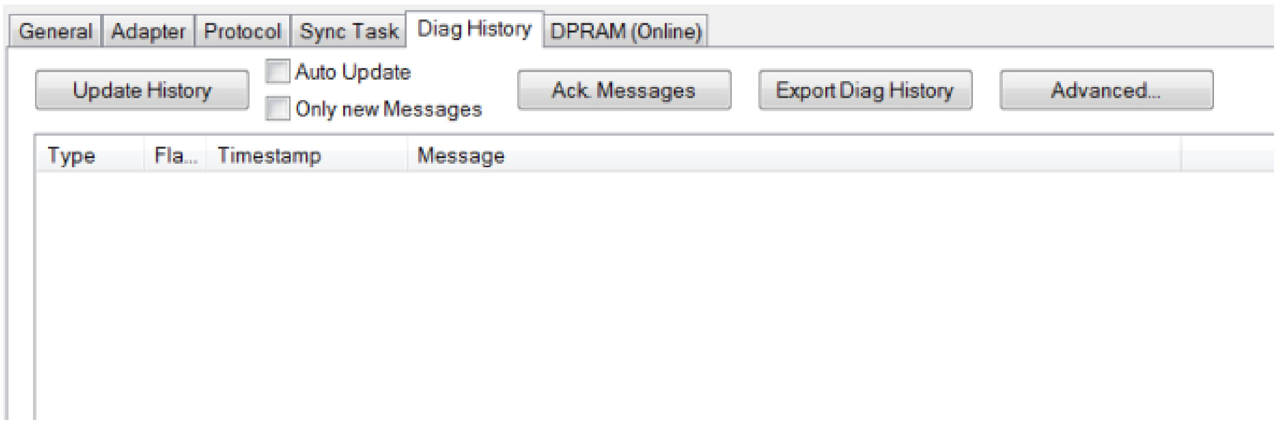
The current valid assembly settings are displayed here; these can differ from the object `0x8001`. The object `0x9001` displays the active parameters.

7 Diagnostic history

The diagnostics history is a tool for monitoring the status of the EtherNet/IP interface and displaying the diagnostic messages with timestamps in plain text.

In addition, information / errors that occurred in the past are logged, in order to enable precise troubleshooting at a later stage. This also applies for errors that only occurred for such a short time that any corresponding messages were not visible.

The diagnostic history is part of the TwinCAT system, where it can be found under Devices, EtherNet/IP in the **Diag History** tab.



7.1 Error codes TF6280

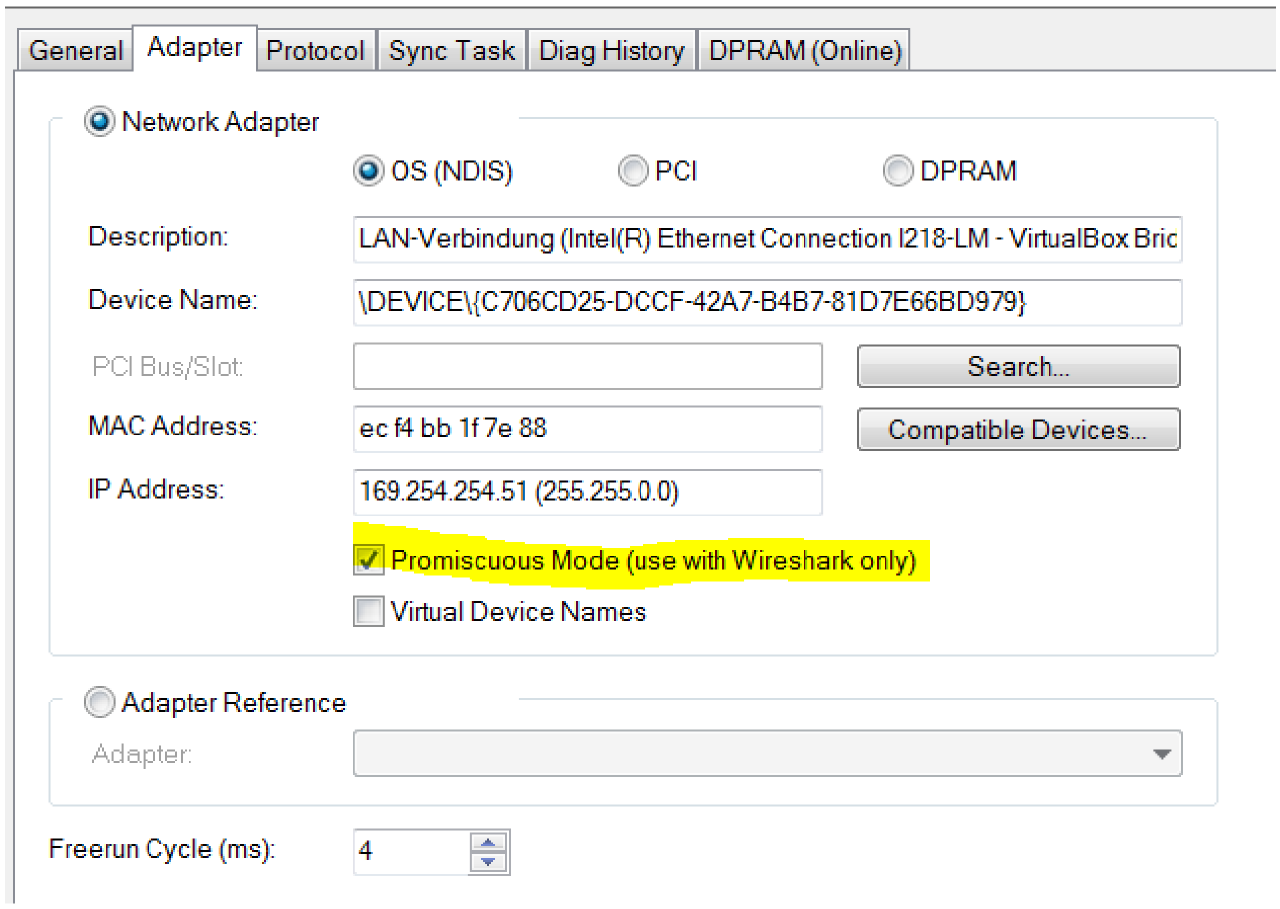
Error	Code hex / (decimal)	Description	Remedy/meaning
CN_ORC_ALREADY_USED	0x100 / (256)	Connection already in use	The connection is already established; use another connection or close this one
CN_ORC_BAD_TRANSPORT	0x103 / (259)	Transport type not supported	The transport type is not supported
CN_ORC_OWNER_CONFLICT	0x106 / (262)	More than one guy configuring	A connection already exists; a further connection cannot be established
CN_ORC_BAD_CONNECTION	0x107 / (263)	Trying to close inactive conn	Faulty connection
CN_ORC_BAD_CONN_TYPE	0x108 / (264)	Unsupported connection type	The Connection type is not supported, check your settings.
CN_ORC_BAD_CONN_SIZE	0x109 / (265)	Connection size mismatch	The connection size does not match, check your settings.
CN_ORC_CONN_UNCONFIGURED	0x110 / (272)	Connection unconfigured	Connection was not configured
CN_ORC_BAD_RPI	0x111 / (273)	Unsupportable RPI	The task time usually doesn't match; make sure that the EL6652 operates internally with 1 ms and that you can adjust this with the Cycle Time Multiplier. Otherwise adjust the task time.
CN_ORC_NO_CM_RESOURCES	0x113 / (275)	Conn Mgr out of connections	No further resources are available
CN_ORC_BAD_VENDOR_PRODUCT	0x114 / (276)	Mismatch in electronic key	Wrong vendor number
CN_ORC_BAD_DEVICE_TYPE	0x115 / (277)	Mismatch in electronic key	Wrong device type
CN_ORC_BAD_REVISION	0x116 / (278)	Mismatch in electronic key	Wrong revision number
CN_ORC_BAD_CONN_POINT	0x117 / (279)	Non-existent instance number	Wrong connection number
CN_ORC_BAD_CONFIGURATION	0x118 / (280)	Bad config instance number	Faulty configuration
CN_ORC_CONN_REQ_FAILS	0x119 / (281)	No controlling connection open	Connection could not be established
CN_ORC_NO_APP_RESOURCES	0x11A / (282)	App out of connections	No more free connections available.

If you cannot fix this error yourself, Support will require the following information:

- TwinCAT version and build number and a
- Wireshark recording

Prepare Wireshark recording

The Wireshark recording can be created with a network hub, a network switch with port mirroring, e.g. the Beckhoff ET2000, or with the **Promiscuous Mode** of the TwinCAT system.



[✓ Network Adapter]

[✓ OS (NDIS)] PCI DPRAM

Description: LAN-Verbindung (Intel(R) Ethernet Connection I218-LM - VirtualBox Bric

Device Name: \DEVICE\{C706CD25-DCCF-42A7-B4B7-81D7E66BD979}

PCI Bus/Slot:

MAC Address: ec f4 bb 1f 7e 88

IP Address: 169.254.254.51 (255.255.0.0)

✓ Promiscuous Mode (use with Wireshark only)

Virtual Device Names

Adapter Reference

Adapter:

Freerun Cycle (ms): 4

8 Appendix

8.1 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.

Download finder

Our download finder contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

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 our internet page: 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
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
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

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

More Information:

www.beckhoff.com/tf6280

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