

BECKHOFF New Automation Technology

Manual | EN

TF8810

TwinCAT 3 | AES70 (OCA) Communication

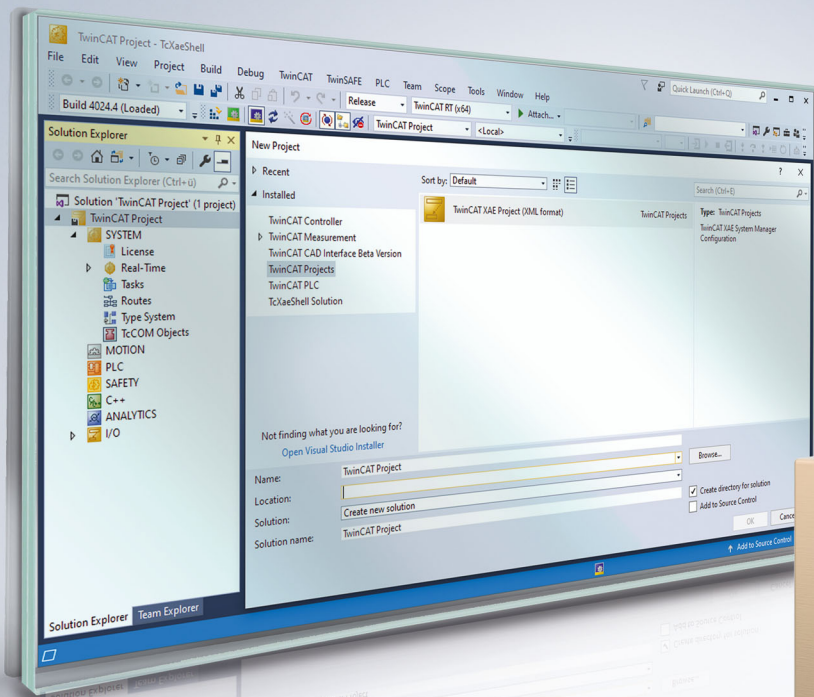


Table of contents

1 Foreword	5
1.1 Notes on the documentation	5
1.2 For your safety	6
1.3 Notes on information security.....	7
2 Overview	8
3 System requirements	9
4 Installation	10
5 Licensing	14
6 Technical introduction	17
7 PLC API	18
7.1 Function blocks	18
7.1.1 FB_OcaDevice	18
7.1.2 FB_OcaRoot	19
7.1.3 Worker function blocks.....	24
7.2 Data types	54
7.2.1 Structures used to represent the properties of OCA objects.....	54
7.2.2 E_OcaStatus	56
7.2.3 E_OcaMuteState.....	57
7.2.4 ST_OcaTemperature	57
7.2.5 ST_OcaDeviceInfo	57
7.2.6 E_OcaMessageType.....	57
8 Examples	58
8.1 Example for using the function block FB_OcaDevice	58
8.2 Example for using the function block FB_OcaRoot.....	58
8.3 Example for using the function block FB_OcaWorker.....	59
8.4 Example for using the function block FB_OcaGain.....	60
9 Support and Service	61

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

© Beckhoff Automation GmbH & Co. KG, Germany.

The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

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>.

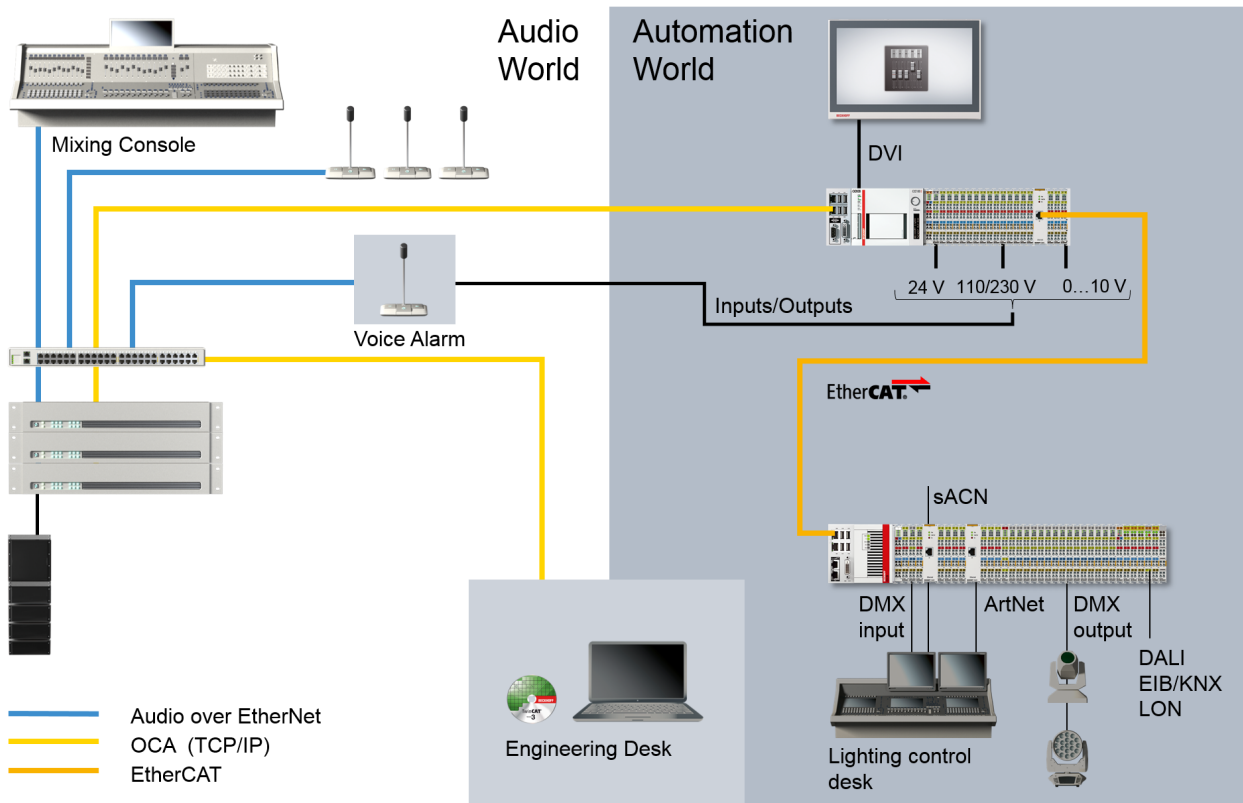
2 Overview

The AES70 standard was published by the Audio Engineering Society, which is based in New York. It defines a scalable control protocol for professional audio devices and describes monitoring and control of such devices, but not the transfer of media data.

The function blocks of the TwinCAT 3 AES70 (OCA) Communication PLC library can be used to establish data exchange between a TwinCAT PLC and a device that supports the AES70 (OCA) standard. The user can query or set properties of different objects in a device (Oca device). Various function blocks such as FB_OcaGain, FB_OcaMute or FB_OcaSwitch are available for this purpose.

The user can thus integrate OCA-compatible audio systems into Beckhoff's PC-based control technology platform and select from a wide range of control panels and various I/Os.

Further information can be found on the Beckhoff website under entertainment industry.



For the OCA-capable amplifiers from d&b audiotechnik, an example of a higher-level function block with some basic functions such as gain, mute or presets is available for download from the d&b audiotechnik website under www.dbaudio.com > Systems > Networks and Integration > Integration

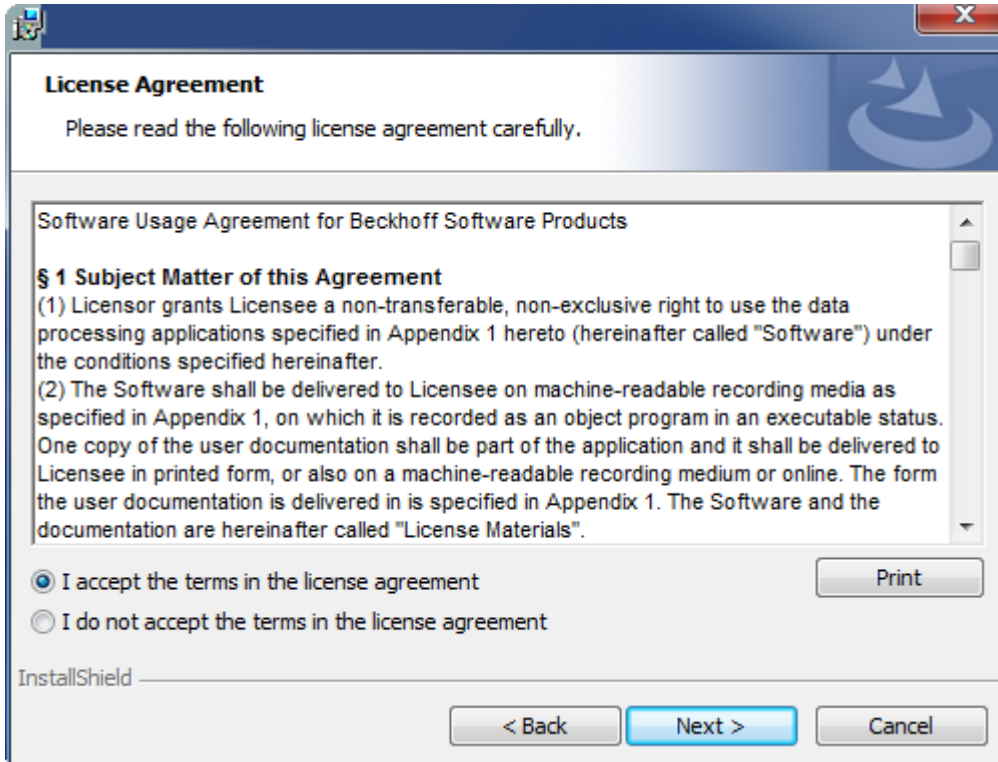
3 System requirements

Technical data	Requirement
Operating system	Windows 7/10, Windows Embedded Standard 7, Windows CE 7
Target platform	PC architecture (x86, x64 or ARM)
TwinCAT version	TwinCAT 3.1 build 4022.2 or higher
Required TwinCAT setup level	TwinCAT 3 XAE, XAR
Required TwinCAT license	TF8810 AES70 (OCA) Communication

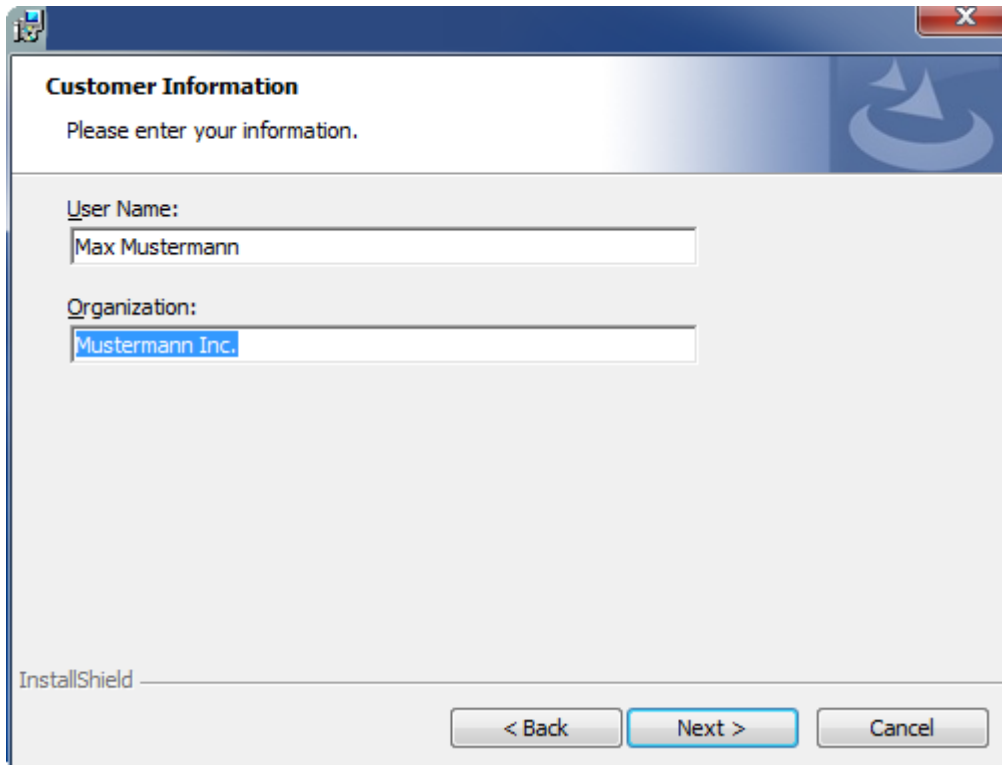
4 Installation

The following section describes how to install the TwinCAT 3 Function for Windows-based operating systems.

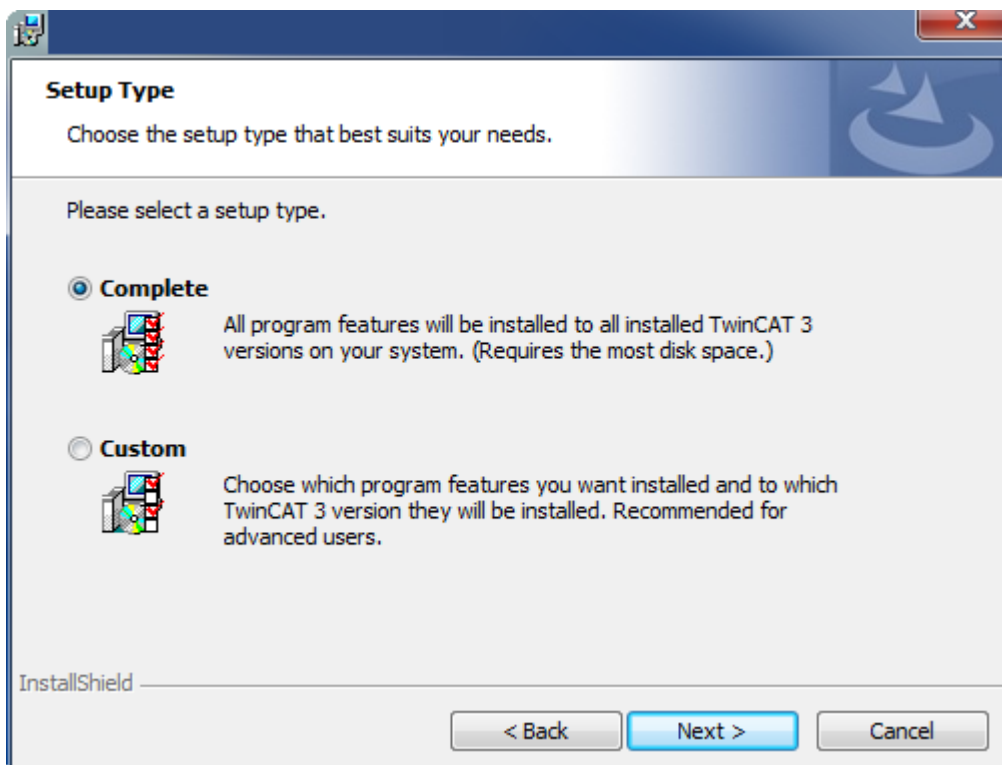
- ✓ The TwinCAT 3 Function setup file was downloaded from the Beckhoff website.
- 1. Run the setup file as administrator. To do this, select the command **Run as administrator** in the context menu of the file.
 - ⇒ The installation dialog opens.
- 2. Accept the end user licensing agreement and click **Next**.



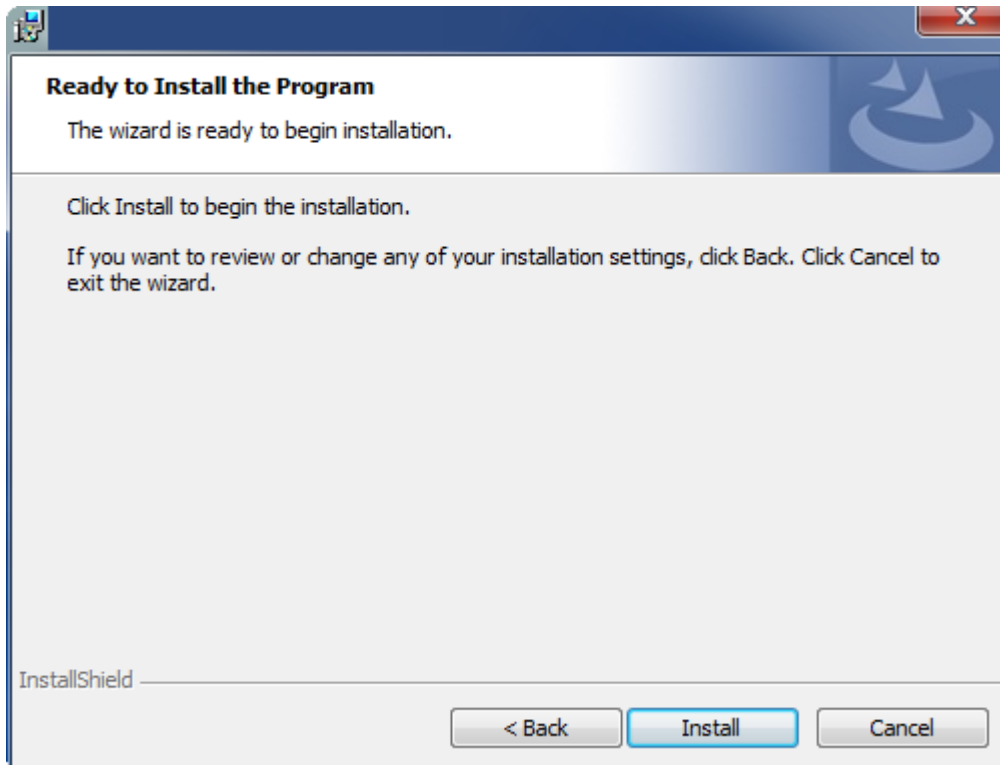
3. Enter your user data.



4. If you want to install the full version of the TwinCAT 3 Function, select **Complete** as installation type. If you want to install the TwinCAT 3 Function components separately, select **Custom**.

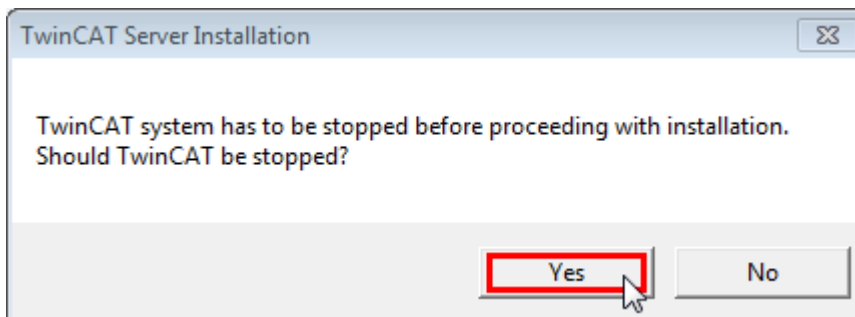


5. Select **Next**, then **Install** to start the installation.

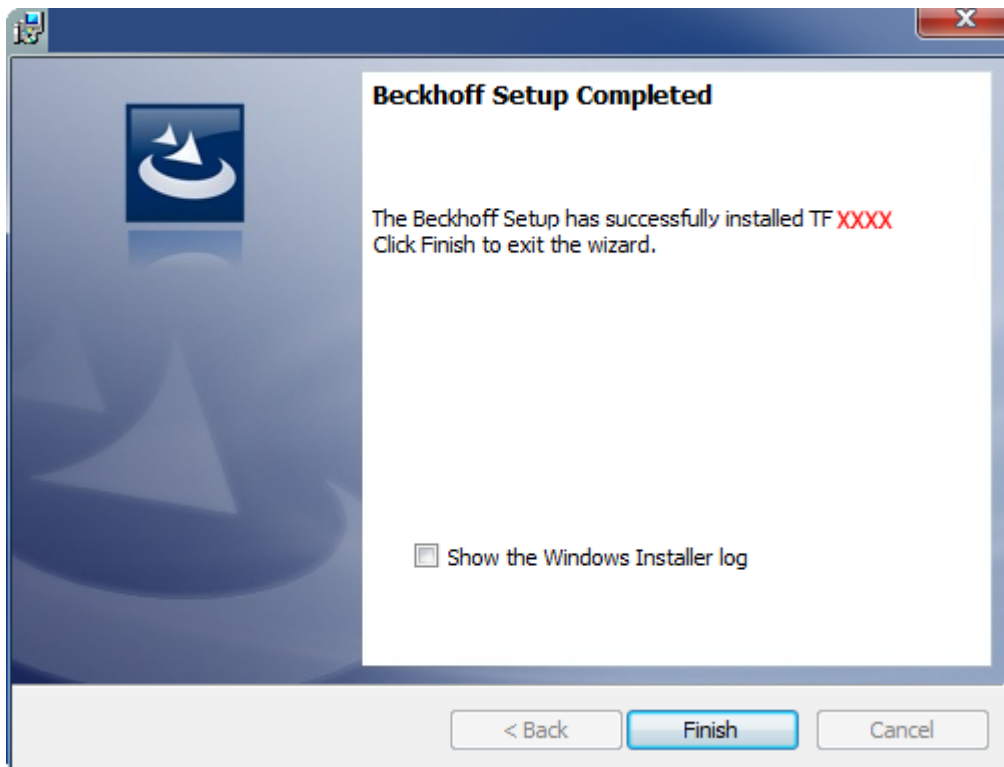


⇒ A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

6. Confirm the dialog with **Yes**.



7. Select **Finish** to exit the setup.



⇒ The TwinCAT 3 Function has been successfully installed and can be licensed (see [Licensing](#) [▶ 14]).

5 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

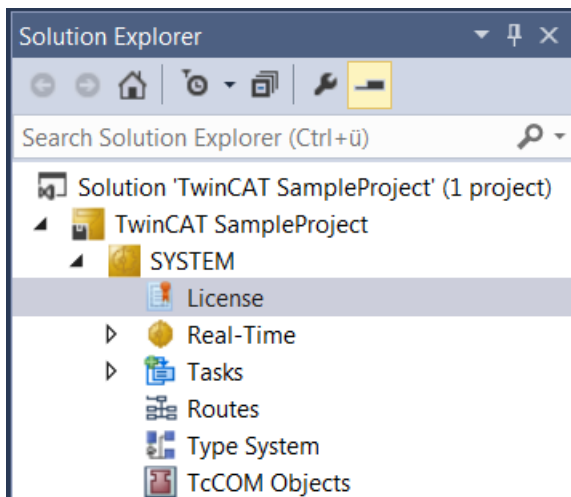
A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "[TwinCAT 3 Licensing](#)".

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.

- 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	<input type="checkbox"/> cpu license
TF3650	TC3 Power Monitoring	<input type="checkbox"/> cpu license
TF3680	TC3 Filter	<input type="checkbox"/> cpu license
TF3800	TC3 Machine Learning Inference Engine	<input type="checkbox"/> cpu license
TF3810	TC3 Neural Network Inference Engine	<input type="checkbox"/> cpu license
TF3900	TC3 Solar-Position-Algorithm	<input type="checkbox"/> cpu license
TF4100	TC3 Controller Toolbox	<input checked="" type="checkbox"/> cpu license
TF4110	TC3 Temperature-Controller	<input type="checkbox"/> cpu license
TF4500	TC3 Speech	<input type="checkbox"/> cpu license

- Open the **Order Information (Runtime)** tab.
 - ⇒ In the tabular overview of licenses, the previously selected license is displayed with the status "missing".
- 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) [Add...]

System Id: 2DB25408-B4CD-81DF-5488-6A3D9B49EF19 | Platform: other (91)

License Request

Provider: Beckhoff Automation [Generate File...]

License Id: [] | Customer Id: []

Comment: []

License Activation

7 Days Trial License... | License Response File...

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

Enter Security Code [X]

Please type the following 5 characters: **OK**

Kg8T4

[] | []

Cancel

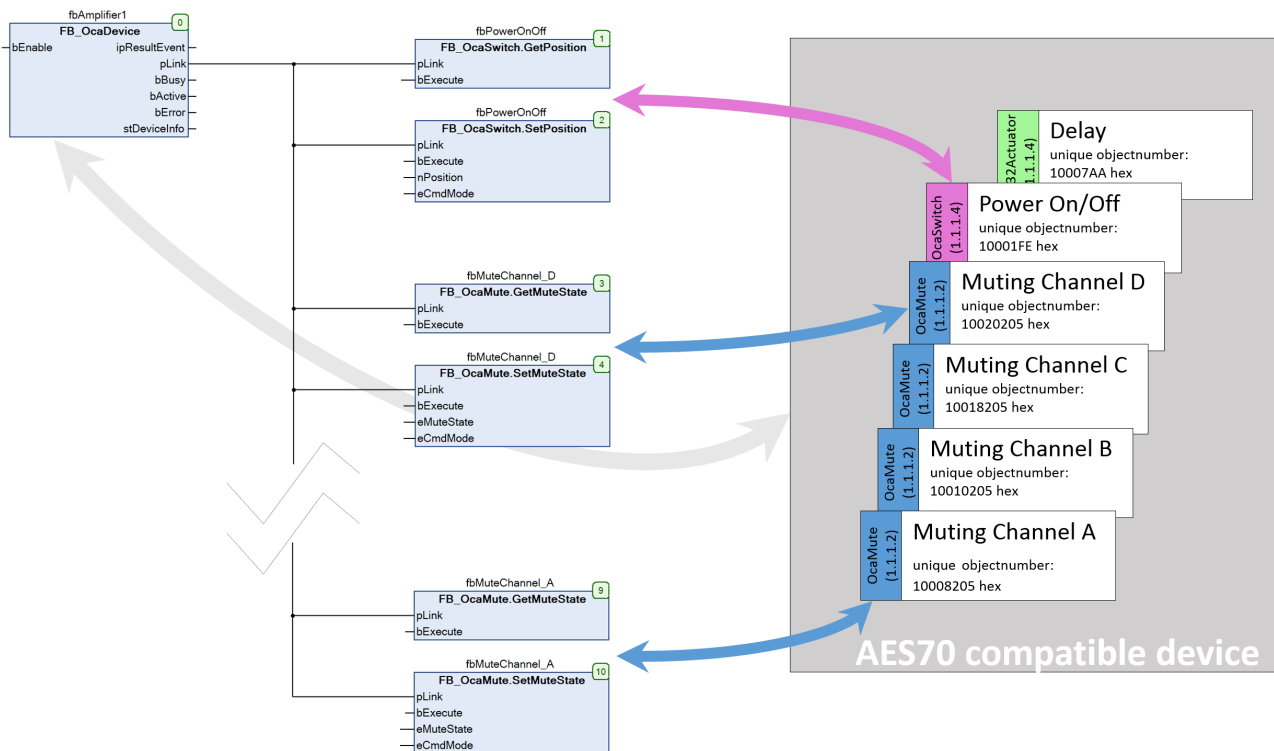
- Enter the code exactly as it is displayed and confirm the entry.
- Confirm the subsequent dialog, which indicates the successful activation.
 - ⇒ In the tabular overview of licenses, the license status now indicates the expiry date of the license.

10. Restart the TwinCAT system.

⇒ The 7-day trial version is enabled.

6 Technical introduction

The AES70 specification describes objects representing functions and device states. These objects are handled with TwinCAT using function block methods. Each object has a unique object number that is defined by the device manufacturer.



For each AES70-compatible device, an instance of the function block **FB_OcaDevice** [► 18] has to be called cyclically. The function block establishes the connection via TCP/IP. The parameterization of the function block (IP address, port etc.) is explained in the [Example for using the function block FB_OcaDevice](#) [► 58]. After calling the function block instance, you must call the methods of those function blocks that represent objects in the AES70-compatible device, for example so-called [worker objects](#) [► 24] such as **OcaMute**, **OcaSwitch**, and so on.

7 PLC API

7.1 Function blocks

7.1.1 FB_OcaDevice



The function block FB_OcaDevice establishes the connection via TcpIp to devices that support the AES70 standard.

● **Cyclically single call**

i The instance of the function block FB_OcaDevice must be called cyclically once at the start of the PLC program.

Syntax

```

VAR_INPUT
  bEnable      : BOOL;
END_VAR
VAR_OUTPUT
  ipTcResultEvent : Tc3_EventLogger.I_TcResultEvent;
  pLink          : POINTER TO ST_Link;
  bBusy         : BOOL;
  bActive       : BOOL;
  bError        : BOOL;
  stDeviceInfo   : ST_OcaDeviceInfo;
END_VAR
    
```

🔧 Inputs

Name	Type	Description
bEnable	BOOL	Enables/disables execution of the function block and starts the "keep alive" mechanism. A rising edge at this input clears any pending errors (indicated by bError = TRUE). The property sDeviceName should be set before the activation. Parameters such as IP address, port, etc. should also be predefined in the global variable list "GVL_AES70".

 **Outputs**

Name	Type	Description
ipTcResultEvent	Tc3_EventLogger.I_TcResultEvent	Result interface with detailed information on the return value
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bBusy	BOOL	TRUE as long as the function block is called with bEnable = TRUE.
bActive	BOOL	Indicates that the device is ready to operate. This output is generally used to activate OCA objects.
bError	BOOL	TRUE, if an error occurs.
stDeviceInfo	ST_OcaDeviceInfo [► 57]	This structure provides information such as the time of the last sent or received message and the number of messages.

 **Properties**

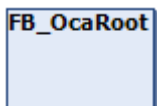
Name	Type	Access	Definition location	Description
AmsNetId	T_AmsNetID	Set	Local	AMS network ID of the device For the local computer (default) an empty string may be specified.
bLocalOcaDevice	BOOL		Local	Currently unused (intended for future extension)
sDeviceName	STRING	Set		Name of the OCA device to be used Parameters such as IP address and port are defined in the array aOcaDevices, which can be found in the global variable list GVL_AES70.
tAdsTimeout	TIME	Set		Maximum time allowed for the execution of the function block This input is internally preset to DEFAULT_ADS_TIMEOUT and does not have to be explicitly assigned.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

See also: [Example for using the function block FB_OcaDevice \[► 58\]](#)

7.1.2 FB_OcaRoot



The function block FB_OcaRoot provides basic OCA functionality. It is the function block from which all other function blocks inherit.

● No explicit call



Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Methods

Name	Definition location	Description
GetClassIdentification [▶ 21]	Local	This method can be used to query the ClassId and ClassVersion of the OCA object. If the query was successful, the result is output in the properties stClassID and stClassVersion.
GetLockable [▶ 21]	Local	This method is used to query whether the OCA object can be locked. If the query was successful, the result is output in the property stLockable.
LockUnlock [▶ 22]	Local	Method for unlocking or locking an OCA object If the method was executed successfully, the result is output in the property stObjectLocked.
GetRole [▶ 22]	Local	OCA objects can have a text label to make their meaning easier to recognize. If the method call was successful, the result is output in the property stRole.
Subscription [▶ 23]	Local	You can use this method to request OCA objects, generate notifications for value changes, or set up sending. If the method was executed successfully, the result is output in the property stSubscribed.

Properties

Name	Type	Access	Definition location	Description
sClassId	STRING	Get	Local	Unique class name
stClassId	ST_ClassIdProperty [▶ 54]	Get	Local	Unique class name
stClassVersion	ST_UINT16Property [▶ 54]	Get	Local	OCA ClassVersion
stLockable	ST_BooleanProperty [▶ 54]	Get	Local	Indicates whether the object can be locked
stSubscribed	ST_BooleanProperty [▶ 54]	Get	Local	Shows whether the object was requested to generate notifications when value changes occur
stRole	ST_StringProperty [▶ 54]	Get	Local	Task of the object in the device (e.g. Config_InputEnable)
nONo	UDINT	Get	Local	Unique number with which the instantiated object is uniquely defined
sOcaObjectDescription	STRING(32)	Set	Local	Arbitrary object name. This is used to assign further object properties to the function block that were defined in the array aOcaDevices (located in the global variable list GVL_AES70).
stObjectLocked	ST_BooleanProperty [▶ 54]	Get	Local	Indicates whether the object is locked.

i AES70 standard

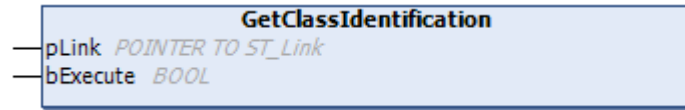
The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.ocaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

See also: [Example for using the function block FB_OcaRoot \[► 58\]](#)

7.1.2.1 GetClassIdentification



The method GetClassIdentification can be used to query the ClassId and ClassVersion of the OCA object. If the query was successful, the result is output in the properties stClassID and stClassVersion.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

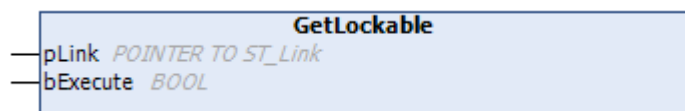
Syntax

```
VAR_INPUT
  pLink : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

7.1.2.2 GetLockable



The method GetLockable is used to query whether the OCA object can be locked. If the query was successful, the result is output in the property stLockable.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

7.1.2.3 LockUnlock



The LockUnlock method can be used to unlock or lock an OCA object. If the method was executed successfully, the result is output in the property stObjectLocked.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

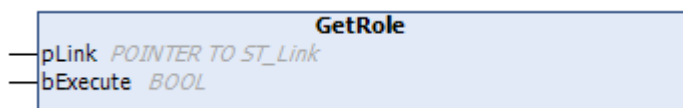
```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  bLock      : BOOL; //If the Method is executed this Boolean Input decides whether the object
should be locked (TRUE) or unlocked (FALSE)
  eCmdMode   : E_OcaMessageTypeCmdbLock;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
bLock	BOOL	Variable that determines whether to lock (TRUE) or unlock (FALSE)
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

7.1.2.4 GetRole



OCA objects can have a text label to make their meaning easier to recognize. If the method call was successful, the result is output in the property stRole.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

7.1.2.5 Subscription



You can use the Subscription method to request OCA objects, generate notifications for value changes, or set up sending of notifications. If the method was executed successfully, the result is output in the property stSubscribed.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  eMode      : E_SubscriptionMode;
  nSubscriptionNo : UDINT;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
eMode	E_SubscriptionMode	Depending on whether a notification request is to be created or deleted, the input of the method can be set to E_SubscriptionMode.ADD_Subscription or E_SubscriptionMode.DELETE_Subscription.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

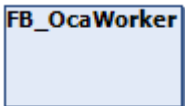
7.1.3 Worker function blocks

Worker function blocks are used to control OCA worker objects.

● AES70 standard

i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.ocaalliance.com.

7.1.3.1 FB_OcaWorker



The function block FB_OcaWorker extends the function block FB_OcaRoot with properties and methods for handling the OCA objects.

● No explicit call

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

```

FB_OcaRoot
  |
  +-- FB_OcaWorker
    
```

Methods

Name	Definition location	Description
GetEnabled / SetEnabled [▶ 25]	Local	Method for querying or setting the Enabled property of the OCA object. If the query was successful, the result is output in the property stEnabled.
GetLabel /SetLabel / RNtfLabel [▶ 26]	Local	This method is used to query whether the OCA object can be locked. If the query was successful, the result is output in the property stLockable.
LockUnlock [▶ 22]	Local	Method for unlocking or locking an OCA object. If the method was executed successfully, the result is output in the property stObjectLocked.
GetRole [▶ 22]	Local	OCA objects can have a text label to make their meaning easier to recognize. If the method call was successful, the result is output in the property stRole.
Subscription [▶ 23]	Local	You can use this method to request OCA objects, generate notifications for value changes, or set up sending. If the method was executed successfully, the result is output in the property stSubscribed.

 Properties

Name	Type	Access	Definition location	Description
stEnabled	ST_BooleanProperty [▶ 54]	Get	Local	Indicates whether the OCA object in the corresponding OCA device is enabled.
stLabel	ST_StringProperty [▶ 54]	Get	Local	OCA objects can have a description text that is stored in this property to facilitate recognition.
stLatency	ST_FLOAT32Property [▶ 54]	Get	Local	Processing latency of the OCA object
stOwner	ST_UINT32Property [▶ 54]	Get	Local	OCA object number of the higher-level object

● AES70 standard

i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

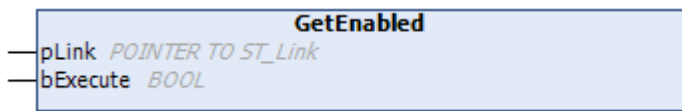
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

See also: [Example for using the function block FB_OcaWorker \[▶ 59\]](#)

7.1.3.1.1 GetEnabled / SetEnabled

GetEnabled



The GetEnabled method is used to query the Enabled property of the OCA object. If the query was successful, the result is output in the property stEnabled.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

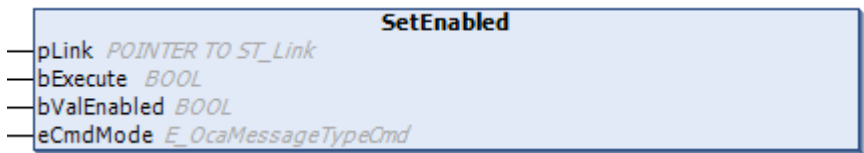
Syntax

```
VAR_INPUT
  pLink    : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
```

 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetEnabled



The SetEnabled method can be used to enable or disable an OCA object. If the method was executed successfully, the result is output in the property stEnabled.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  bValEnabled : BOOL;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
    
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
bValEnabled	BOOL	Depending on whether the OCA object is to be enabled or disabled, this input must be assigned TRUE or FALSE.
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

7.1.3.1.2 GetLabel / SetLabel / RNtfLabel

GetLabel



OCA objects can be labeled to facilitate recognition. The GetLabel method is used to query this property of the OCA object. If the query was successful, the result is output in the property stLabel.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetLabel



The SetLabel method can be used to set the labeling of an OCA object. If the method was executed successfully, the result is output in the property stLabel.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  sLabel     : STRING;
  eCmdMode  : E_OcaMessageTypeCmd;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
sLabel	STRING	Label for the OCA object
eCmdMode	E_ OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_ OcaMessageTypeCmd.OcaCmdRrq or E_ OcaMessageTypeCmd.OcaCmd.

RNtfLabel



If the OCA object is prompted to notify changes (using the [Subscription](#) |▶ 23| method), incoming notifications are read using the RNtfLabel method.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable    : BOOL;
  nSubscriptionNo : UDINT;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.1.3 GetLatency / SetLatency / RNtfLatency

GetLatency



The GetLatency method is used to query the processing latency of the OCA object. If the query was successful, the result is output in the property stLatency.



Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

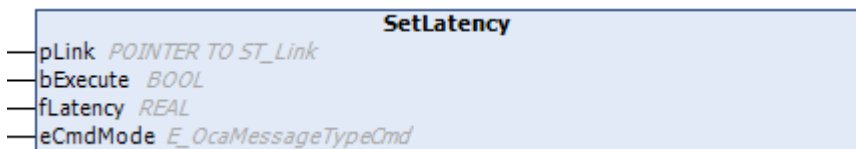
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetLatency



The SetLatency method can be used to set the processing latency of an OCA object. If the method was executed successfully, the result is output in the property stLatency.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  fLatency   : REAL;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
fLatency	REAL	Value to be assigned to this property of the OCA object
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfLatency



If the OCA object was prompted to notify changes (using the [Subscription \[► 23\]](#) method), incoming notifications are read using the RNtfLatency method.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

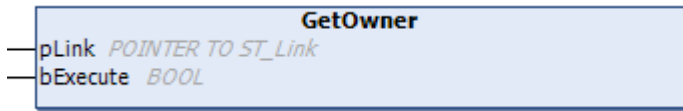
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable     : BOOL;
  nSubscriptionNo : UDINT;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.1.4 GetOwner



The GetOwner method is used to query the higher-level element. If the query was successful, the result is output in the property stOwner.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
    pLink    : POINTER TO ST_Link;
    bExecute : BOOL;
END_VAR
```

📌 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

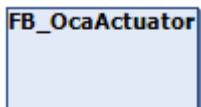
7.1.3.2 Actuator function blocks

Actuator objects are worker objects whose purpose is the control of electronic functions.

● AES70 standard

i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

7.1.3.2.1 FB_OcaActuator



The function block FB_OcaActuator extends the function block FB_OcaWorker. It is the function block from which all function blocks that are assigned to the Actuator category inherit. FB_OcaActuator has neither properties nor methods.

● No explicit call

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

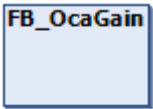
Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaActuator

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.2 FB_OcaGain



The function block FB_OcaGain provides properties and methods for handling OCA objects for gain adjustment.

No explicit call
 Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaActuator
 - FB_OcaGain

Methods

Name	Definition location	Description
GetGain [▶ 32]	Local	Used to query the properties Gain, GainMAX and GainMIN of the OCA object. If the query was successful, the result is output in the properties stGain, stGainMAX and stGainMIN.
SetGain [▶ 32]	Local	Use this method to set the gain of an OCA object in dB. If the method was executed successfully, the result is output in the property stGain.
RNtfGain [▶ 32]	Local	If the OCA object was prompted to notify changes (using the Subscription [▶ 23] method), this method is used to read incoming notifications.

Properties

Name	Type	Access	Definition location	Description
stGain	ST_FLOAT32Property [▶ 54]	Get	Local	Gain in dB
stGainMAX	ST_FLOAT32Property [▶ 54]	Get	Local	Maximum gain value in dB
stGainMIN	ST_FLOAT32Property [▶ 54]	Get	Local	Minimum gain value in dB



AES70 standard

The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

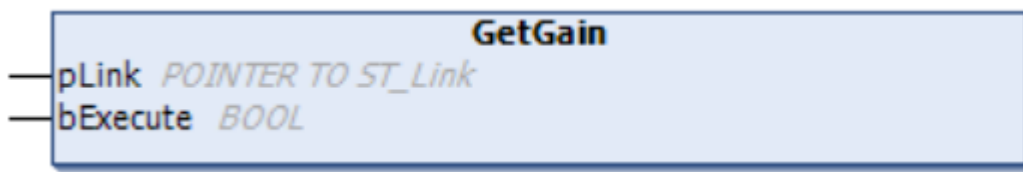
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

See also: [Example for using the function block FB_OcaGain](#) [▶ 60]

7.1.3.2.2.1 GetGain / SetGain / RNtfGain

GetGain



The GetGain method is used to query the properties Gain, GainMAX and GainMIN of the OCA object. If the query was successful, the result is output in the properties stGain, stGainMAX and stGainMIN.



Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

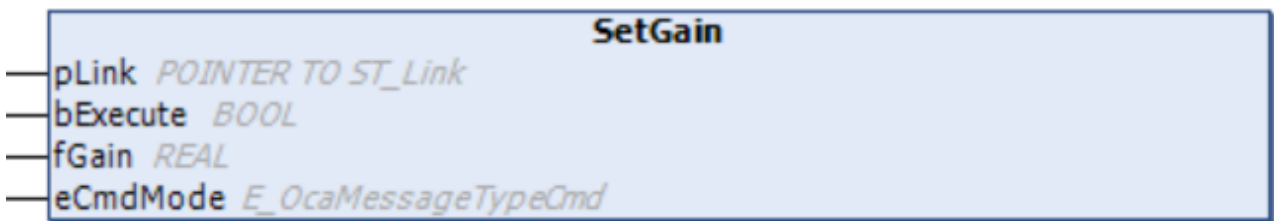
Syntax

```
VAR_INPUT
  pLink    : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetGain



Use the SetGain method to set the gain of an OCA object in dB. If the method was executed successfully, the result is output in the property stGain.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

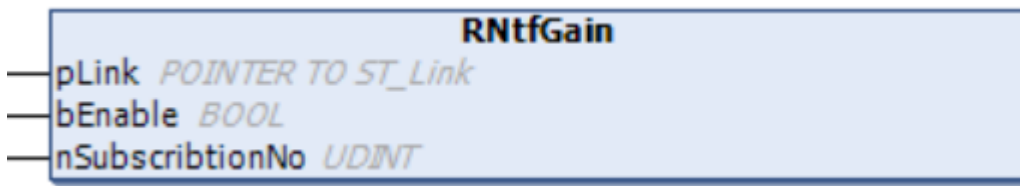
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  fGain      : REAL;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
fGain	REAL	Gain value in dB to which the OCA object is to be set.
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfGain



If the OCA object was prompted to notify changes (using the [Subscription \[▶ 23\]](#) method), incoming notifications are read using the RNtfGain method.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

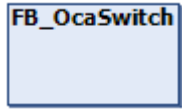
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable    : BOOL;
  nSubscriptionNo : UDINT;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.2.3 FB_OcaSwitch



The function block FB_OcaSwitch extends the function block FB_OcaActuator with properties and methods and for handling OCA switch objects. These objects can have multiple switch positions. Individual positions can be enabled or disabled and given names.



No explicit call

Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

FB_OcaRoot

 FB_OcaWorker

 FB_OcaActuator

 FB_OcaSwitch

Methods

Name	Definition location	Description
GetPosition [▶ 35]	Local	This method is used to query the Position, PositionMAX, and PositionMIN properties of the OCA object.
SetPosition [▶ 35]	Local	Use this method to set the switch position of an OCA object.
RNtfPosition [▶ 35]	Local	This method reads incoming notifications
GetPositionEnabled [▶ 37]	Local	Method for querying a switch position
SetPositionEnabled [▶ 37]	Local	Method for enabling or disabling a switch position
GetPositionName	Local	Method for querying the name of a switch position
SetPositionName	Local	Use this method to assign a name to a switch position.

 **Properties**

Name	Type	Access	Definition location	Description
stPosition	ST_UINT16Property [▶ 54]	Get	Local	Indicates the switch position of the OCA object.
stPosition MAX	ST_UINT16Property [▶ 54]	Get	Local	Indicates the lowest switch position of the OCA object.
stPosition MIN	ST_UINT16Property [▶ 54]	Get	Local	Indicates the highest switch position of the OCA object.
stPosition Enabled	ST_PositionEnabledProperty [▶ 54]	Get	Local	Indicates whether a particular switch position is enabled or disabled.
stPosition Name	ST_PositionNameProperty [▶ 54]	Get	Local	Indicates the position name of a particular switch position.

● AES70 standard

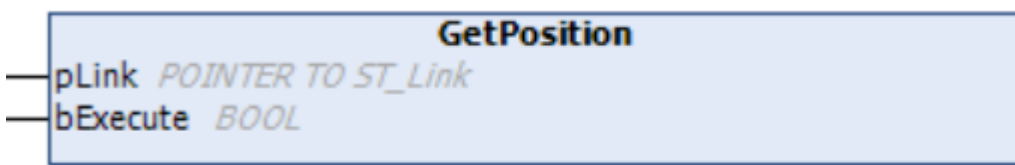
i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.ocaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.3.1 GetPosition / SetPosition / RNtfPosition

GetPosition



The GetPosition method is used to query the Position, PositionMAX, and PositionMIN properties of the OCA object. If the query was successful, the result is output in the properties stPosition, stPositionMAX and stPositionMIN.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
    pLink    : POINTER TO ST_Link; //Pointer to address of the structure which links the OCA objects
            to the OCA device
    bExecute : BOOL; //The Method is triggered by a rising edge at this input.
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetPosition



The SetPosition method can be used to set the switch position of an OCA object. If the method was executed successfully, the result is output in the property stPosition.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

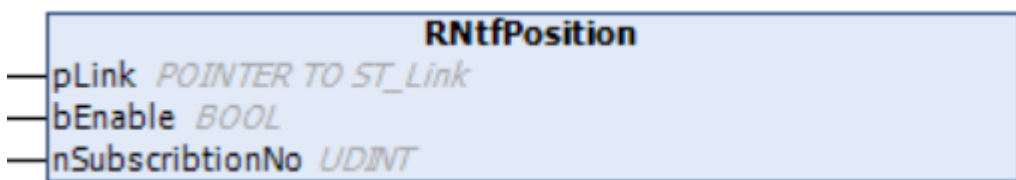
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link; //Pointer to address of the structure which links the OCA objects
to the OCA device
  bExecute   : BOOL; //The Method is triggered by a rising edge at this input.
  nPosition  : UINT;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nPosition	UINT	Switch position for the OCA object.
eCmdMode	E_OcaMessageTypeCmd; CmdbLock	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfPosition



If the OCA object was prompted to notify changes (using the [Subscription](#) [► 23] method), incoming notifications are read using the RNtfPosition method.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

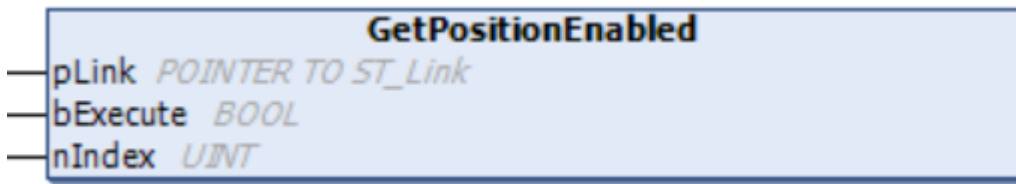
```
VAR_INPUT
    pLink      : POINTER TO ST_Link;
    bEnable    : BOOL;
    nSubscriptionNo : UDINT;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.2.3.2 GetPositionEnabled / SetPositionEnabled

GetPositionEnabled



The GetPositionEnabled method is used to query whether the respective switch position determined by nIndex is enabled or disabled. If the query was successful, the result is output in the property stPositionEnabled.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

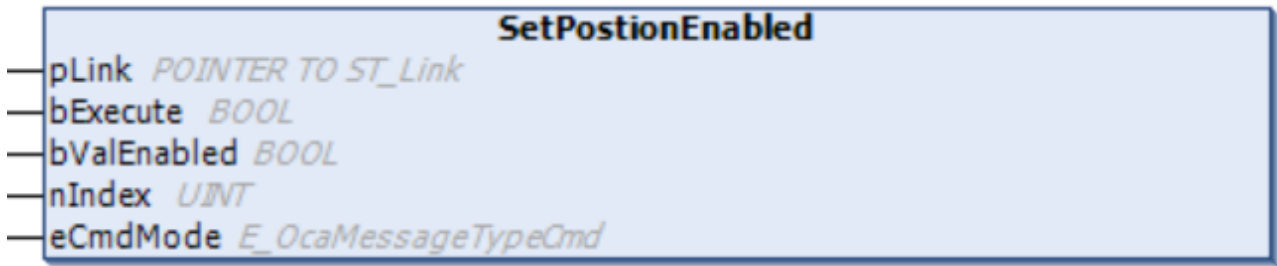
Syntax

```
VAR_INPUT
    pLink      : POINTER TO ST_Link; //Pointer to address of the structure which links the OCA objects
to the OCA device
    bExecute   : BOOL; //The Method is triggered by a rising edge at this input.
    nIndex    : UINT; //The Index of the queried Position
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nIndex	UINT	Index of the switch position to be queried

SetPositionEnabled



The SetPositionEnabled method can be used to enable or disable a specific switch position specified by nIndex. If the query was successful, the result is output in the property stPositionEnabled.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

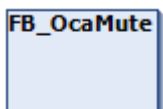
Syntax

```
VAR_INPUT
    pLink      : POINTER TO ST_Link; //Pointer to address of the structure which links the OCA
objects to the OCA device
    bExecute   : BOOL; //The Method is triggered by a rising edge at this input.
    bValEnabled : BOOL; //Set this Input to TRUE to enable the Position specified by nIndex or FALSE
to disable this Position
    nIndex     : UINT; //The Index of the Position which should be modified
    eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

🔧 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
bValEnabled	BOOL	If this input is set to TRUE, the switch position selected by nIndex is enabled when this method is executed. If the input is set to FALSE, the switch position is disabled.
nPosition	UINT	Switch position for the OCA object.
eCmdMode	E_OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

7.1.3.2.4 FB_OcaMute



The function block FB_OcaMute provides properties and methods for handling OCA objects for muting.

● No explicit call

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaActuator
 - FB_OcaMute

Methods

Name	Definition location	Description
GetMuteState [▶ 39]	Local	This method is used to query the Position, PositionMAX, and PositionMIN properties of the OCA object.
SetMuteState [▶ 39]	Local	Use this method to set the switch position of an OCA object.
RNtfMuteState [▶ 39]	Local	This method reads incoming notifications.

Properties

Name	Type	Access	Definition location	Description
stMuteState	ST_MuteProperty [▶ 54]	Get	Local	Indicates whether the OCA object is muted or not.

AES70 standard

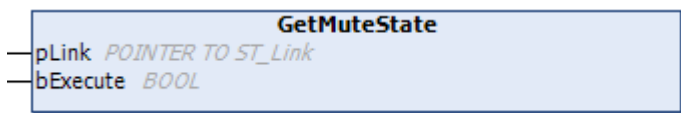
The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.ocaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.4.1 GetMuteState / SetMuteState / RNtfMuteState

GetMuteState



The GetPosition method is used to query the MuteState property of the OCA object. If the query was successful, the result is output in the property stMuteState.

Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

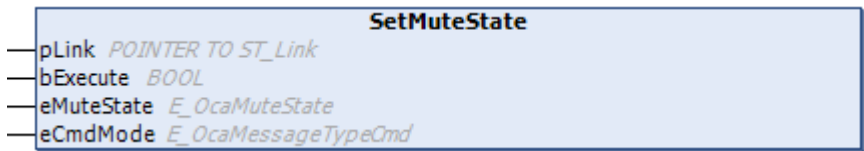
Syntax

```
VAR_INPUT
    pLink : POINTER TO ST_Link;
    bExecute : BOOL;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetMuteState



The SetMuteState method can be used to set the muting feature of an OCA object. If the method was executed successfully, the result is output in the property stMuteState.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

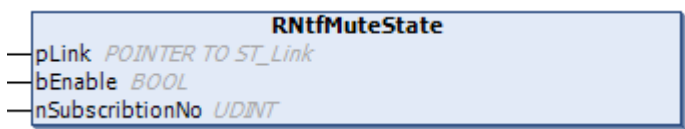
```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  eMuteState : E_OcaMuteState;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
eMuteState	E_OcaMuteState [► 57]	Muting setting (muted/unmuted) to be assigned to the OCA object.
eCmdMode	OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfMuteState



If the OCA object was prompted to notify changes (using the [Subscription \[► 23\]](#) method), incoming notifications are read using the RNtfMuteState method.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

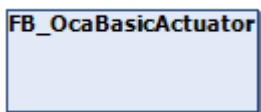
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable    : BOOL;
  nSubscriptionNo : UDINT;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.2.5 FB_OcaBasicActuator



The function block FB_OcaBasicActuator extends the function block FB_OcaActuator. It is the function block from which all function blocks that are assigned to the BasicActuator category inherit. FB_OcaBasicActuator has neither properties nor methods.

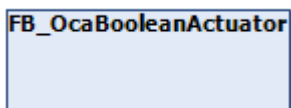
Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaActuator
 - FB_OcaBasicActuator

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.6 FB_OcaBooleanActuator



The function block FB_OcaBooleanActuator extends the function block FB_OcaBasicActuator with properties and methods for handling OcaBooleanActuator objects.

● No explicit call

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker

FB_OcaActuator

FB_OcaBasicActuator

FB_OcaBooleanActuator

Methods

Name	Definition location	Description
GetSetting [▶ 42]	Local	This method is used to query a Boolean property.
SetSetting [▶ 42]	Local	This method can be used to set the value of a Boolean property.
RNtfSetting [▶ 42]	Local	This method reads incoming notifications

Properties

Name	Type	Access	Definition location	Description
stSetting	ST_BooleanProperty [▶ 54]	Get	Local	Indicates the switch position of the OCA object.

AES70 standard

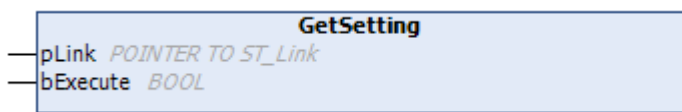
The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.6.1 GetSetting / SetSetting / RNtfSetting

GetSetting



The GetSetting method is used to query the Boolean property of the OCA object. If the query was successful, the result is output in the property stSetting.

Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
  pLink    : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetSetting

```

SetSetting
pLink POINTER TO ST_Link
bExecute BOOL
bSetting BOOL
eCmdMode E_OcaMessageTypeCmd
    
```

The SetSetting method can be used to set the muting feature of an OCA object. If the method was executed successfully, the result is output in the property stMuteState.

i **Cyclic method call**

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  bSetting   : BOOL;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
bSetting	BOOL	Value to be assigned to the property of the OCA object.
eCmdMode	OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfMuteState

```

RNtfSetting
pLink POINTER TO ST_Link
bEnable BOOL
nSubscriptionNo UDINT
    
```

If the OCA object was prompted to notify changes (using the [Subscription](#) [▶ 23] method), incoming notifications are read using the RNtfSetting method.

i **Cyclic method call**

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable    : BOOL;
  nSubscriptionNo : UDINT;
END_VAR

```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.2.7 FB_OcaStringActuator**FB_OcaStringActuator**

The function block FB_OcaStringActuator extends the function block FB_OcaBasicActuator with properties and methods for handling OcaStringActuator objects.

● No explicit call

Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

FB_OcaRoot

FB_OcaWorker

FB_OcaActuator

FB_OcaBasicActuator

FB_OcaStringActuator

Methods

Name	Definition location	Description
GetValue [▶ 45]	Local	This method is used to query a signed property (integer, signed data type, 32 bits).
SetValue [▶ 45]	Local	This method can be used to set the value of the property.
RNtfValue [▶ 45]	Local	This method reads incoming notifications
GetMaxLen [▶ 47]	Local	Use this method to query the maximum length of the string that the OCA object accepts.

 Properties

Name	Type	Access	Definition location	Description
stSetting	ST_STRINGProperty [▶ 54]	Get	Local	Contains the string
stMaxLen	ST_UINT16Property [▶ 54]	Get	Local	Contains the maximum length of the string

● AES70 standard

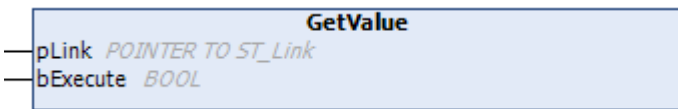
i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.7.1 GetValue / SetValue / RntfValue

GetValue



The GetValue method is used to query the property of the OCA object. If the query was successful, the result is output in the property stSetting.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

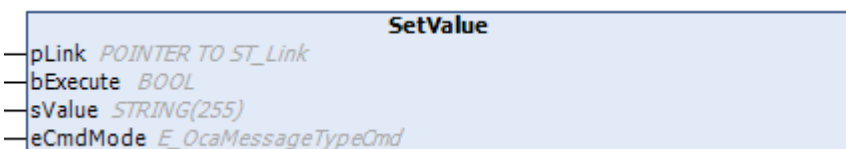
Syntax

```
VAR_INPUT
    pLink    : POINTER TO ST_Link;
    bExecute : BOOL;
END_VAR
```

 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetValue



The SetValue method can be used to set the property of an OCA object. If the method was executed successfully, the result is output in the property stSetting.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

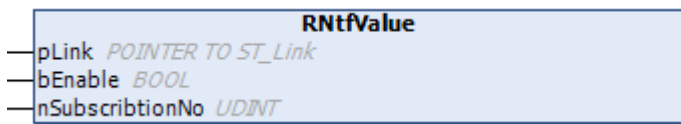
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  sValue     : STRING;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSetting	UDINT	Value to be assigned to the property of the OCA object.
eCmdMode	OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RNtfValue



If the OCA object was prompted to notify changes (using the Subscription [▶ 23] method), incoming notifications are read using the RNtfValue method.

Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

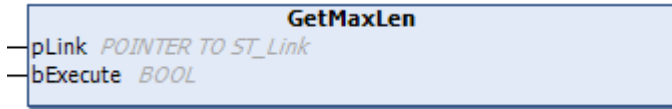
Syntax

```
VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bEnable    : BOOL;
  nSubscriptionNo : UDINT;
END_VAR
```

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.2.7.2 GetMaxLen



The GetMaxLen method is used to query the maximum accepted length of the string of the OCA object (OcaStringActuator). If the query was successful, the result is output in the property stMaxLen.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

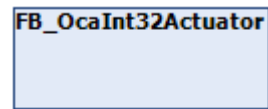
```

VAR_INPUT
  pLink    : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR
    
```

📌 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

7.1.3.2.8 FB_OcaInt32Actuator



The function block FB_OcaInt32Actuator extends the function block FB_OcaBasicActuator with properties and methods for handling OcaInt32Actuator objects.

● No explicit call

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaActuator
 - FB_OcaBasicActuator
 - FB_OcaInt32Actuator

Methods

Name	Definition location	Description
GetValue [▶ 48]	Local	This method is used to query a signed property (integer, signed data type, 32 bits).
SeValue [▶ 48]	Local	This method can be used to set the value of the property.
RNtfValue [▶ 48]	Local	This method reads incoming notifications

Properties

Name	Type	Access	Definition location	Description
stSetting	ST_INT32Property [▶ 54]	Get	Local	Contains the value (integer, signed data type, 32 bit).
stSettingMAX	ST_INT32Property [▶ 54]	Get	Local	Contains the maximum value (integer, signed data type, 32 bit).
stSettingMIN	ST_INT32Property [▶ 54]	Get	Local	Contains the minimum value (integer, signed data type, 32 bit).

● AES70 standard

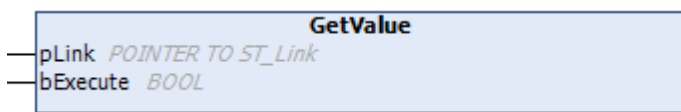
i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.2.8.1 GetValue / SetValue / RNtfValue

GetValue



The GetValue method is used to query the property of the OCA object. If the query was successful, the result is output in the properties tSetting, stSettingMAX and stSettingMIN.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink   : POINTER TO ST_Link;
  bExecute : BOOL;
END_VAR

```


 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

SetValue

```

SetValue
— pLink POINTER TO ST_Link
— bExecute BOOL
— nSetting DINT
— eCmdMode E_OcaMessageTypeCmd
    
```

The SetValue method can be used to set the property of an OCA object. If the method was executed successfully, the result is output in the property stSetting.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```

VAR_INPUT
  pLink      : POINTER TO ST_Link;
  bExecute   : BOOL;
  nSetting   : UDINT;
  eCmdMode   : E_OcaMessageTypeCmd;
END_VAR
    
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSetting	UDINT	Value to be assigned to the property of the OCA object.
eCmdMode	OcaMessageTypeCmd	Depending on whether confirmation is required or not, this input variable is set to E_OcaMessageTypeCmd.OcaCmdRrq or E_OcaMessageTypeCmd.OcaCmd.

RntfValue

```

RntfValue
— pLink POINTER TO ST_Link
— bEnable BOOL
— nSubscriptionNo UDINT
    
```

If the OCA object is prompted to notify changes (using the [Subscription](#) [▶ 23] method), incoming notifications are read using the RntfValue method.

● Cyclic method call

i Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Syntax

```
VAR_INPUT
    pLink      : POINTER TO ST_Link;
    bEnable    : BOOL;
    nSubscriptionNo : UDINT;
END_VAR
```

 **Inputs**

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

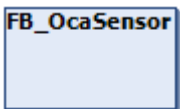
7.1.3.3 Sensor function blocks

Sensors enable querying of different parameters.

 **AES70 standard**

i The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

7.1.3.3.1 FB_OcaSensor

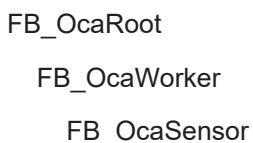


The function block FB_OcaSensor extends the function block FB_OcaWorker. It is the function block from which all function blocks that are assigned to the Sensor category inherit.

 **No explicit call**

i Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy



 **Methods**

Name	Definition location	Description
<u>GetReadingState</u> ▶ 51	Local	You can use this method to query whether the value that was read is valid or not. If the query was successful, the result is output in the property stReadingState.

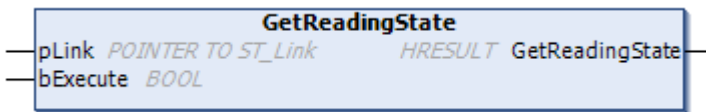
 Properties

Name	Type	Access	Definition location	Description
stReadingState	ST_SensorReadingState [► 54]	Get	Local	Indicates whether the queried value is valid or not.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.3.1 GetReadingState



The GetReadingState method is used to query the ReadingState property of the OCA object. If the query was successful, the result is output in the property stReadingState.

- Cyclic method call**
Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

 Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

7.1.3.3.2 FB_OcaTemperatureSensor



The function block FB_OcaTemperatureSensor provides properties and methods for querying OCA objects that map a temperature value.

- No explicit call**
Since there is no code in the body of the function block, it should not be called explicitly. Instead, use the corresponding methods of the function block.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaSensor
 - FB_OcaTemperatureSensor

Methods

Name	Definition location	Description
GetReading	Local	Used to query the properties Reading, minReading and maxReading of the OCA object. If the query was successful, the result is output in the properties stReading, stReadingMAX and stReadingMIN.
RNtfGain	Local	If the OCA object was prompted to notify changes (using the Subscription [▶ 23] method), this method is used to read incoming notifications.

AES70 standard

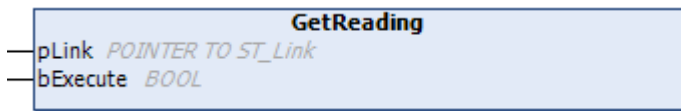
The names of variables and function blocks have been adapted to the AES70 standard where possible. Information about this communication protocol can be found at www.aes.org and www.oaalliance.com.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.3.2.1 GetReading / RntfReading

GetReading



The GetReading method is used to query the property of the OCA object. If the query was successful, the result is output in the properties stReading, stReadingMAX and stReadingMIN

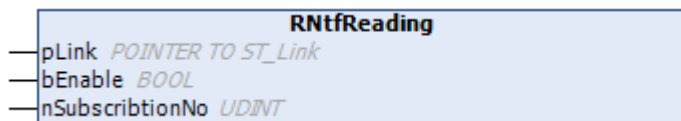
Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.

RntfReading



If the OCA object was prompted to notify changes (using the [Subscription \[▶ 23\]](#) method), incoming notifications are read using the RntfReading method.

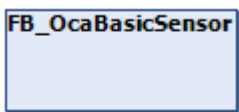
Cyclic method call

Since several PLC cycles may pass between sending and the response from the device, this method should be executed cyclically.

Inputs

Name	Type	Description
pLink	POINTER TO ST_Link	Pointer for establishing a connection between OCA objects and the OCA device.
bExecute	BOOL	The method is executed with a positive edge at the bExecute input.
nSubscriptionNo	UDINT	Unique number for localizing the respective subscription.

7.1.3.3 FB_OcaBasicSensor



The function block FB_OcaBasicSensor extends the function block [FB_OcaSensor](#) [► 50]. It is the function block from which all function blocks that are assigned to the BasicSensor category inherit. FB_OcaBasicSensor has neither properties nor methods.

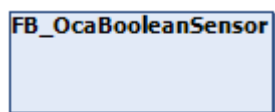
Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaSensor
 - FB_OcaBasicSensor

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4022.2 or higher	PC or CX (x64, x86, ARM)	Tc3_AES70

7.1.3.3.4 FB_OcaBooleanSensor

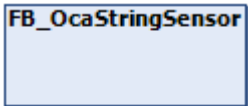


In terms of functionality, the function block FB_OcaBooleanSensor is similar to the function block [FB_OcaBooleanActuator](#) [► 41]. However, since sensors can only be read, there is no corresponding set method.

Inheritance hierarchy

- FB_OcaRoot
 - FB_OcaWorker
 - FB_OcaSensor
 - FB_OcaBasicSensor
 - FB_OcaBooleanSensor

7.1.3.3.5 FB_OcaStringSensor



In terms of functionality, the function block FB_OcaBooleanSensor is similar to the function block [FB_OcaStringActuator \[► 44\]](#). However, since sensors can only be read, there is no corresponding set method.

Inheritance hierarchy

FB_OcaRoot

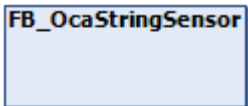
FB_OcaWorker

FB_OcaSensor

FB_OcaBasicSensor

FB_OcaStringSensor

7.1.3.3.6 FB_OcaInt32Sensor



In terms of functionality, the function block FB_OcaBooleanSensor is similar to the function block [FB_OcaInt32Actuator \[► 47\]](#). However, since sensors can only be read, there is no corresponding set method.

Inheritance hierarchy

FB_OcaRoot

FB_OcaWorker

FB_OcaSensor

FB_OcaBasicSensor

FB_OcaInt32Sensor

7.2 Data types

7.2.1 Structures used to represent the properties of OCA objects

ST_OcaProperty

All structures listed below inherit from this structure.

```

TYPE ST_OcaProperty :
STRUCT
    eState: E_OcaStatus := E_OcaStatus.Undefined;
    sPropString: STRING(5) := '00p00';
END_STRUCT
END_TYPE

```

See also: [E_OcaStatus \[► 56\]](#)

ST_BooleanProperty

```
TYPE ST_BooleanProperty EXTENDS ST_OcaProperty :
STRUCT
    bVal: BOOL := FALSE;
END_STRUCT
END_TYPE
```

ST_ClassIdProperty

```
TYPE ST_ClassIdProperty EXTENDS ST_OcaProperty :
STRUCT
    nVal: ST_OcaClassId;
END_STRUCT
END_TYPE
```

ST_FLOAT32Property

```
TYPE ST_FLOAT32Property EXTENDS ST_OcaProperty:
STRUCT
    fVal: REAL;
END_STRUCT
END_TYPE
```

ST_INT16Property

```
TYPE ST_INT16Property EXTENDS ST_OcaProperty :
STRUCT
    nVal: INT;
END_STRUCT
END_TYPE
```

ST_INT32Property

```
TYPE ST_INT32Property EXTENDS ST_OcaProperty :
STRUCT
    nVal: DINT;
END_STRUCT
END_TYPE
```

ST_INT8Property

```
TYPE ST_INT8Property EXTENDS ST_OcaProperty :
STRUCT
    nVal: BYTE;
END_STRUCT
END_TYPE
```

ST_MuteStateProperty

```
TYPE ST_MuteStateProperty EXTENDS ST_OcaProperty :
STRUCT
    eVal: E_OcaMuteState := E_OcaMuteState.Unmuted ;
END_STRUCT
END_TYPE
```

See also: [E_OcaMuteState](#) [[▶ 57](#)]

ST_PolarityStateProperty

```
TYPE ST_PolarityStateProperty EXTENDS ST_OcaProperty :
STRUCT
    eVal: E_OcaPolarityState := E_OcaPolarityState.NonInverted;
END_STRUCT
END_TYPE
```

ST_PositionEnabledProperty

```
TYPE ST_PositionEnabledProperty EXTENDS ST_BooleanProperty:
STRUCT
    nIndex: UINT;
END_STRUCT
END_TYPE
```

ST_PositionNameProperty

```

TYPE ST_PositionNameProperty EXTENDS ST_StringProperty :
STRUCT
  nIndex: UINT;
END_STRUCT
END_TYPE

```

ST_SensorReadingState

```

TYPE ST_SensorReadingState EXTENDS ST_OcaProperty :
STRUCT
  eVal: E_OcaSensorReadingState := E_OcaSensorReadingState.eUnknown;
END_STRUCT
END_TYPE

```

ST_StringProperty

```

TYPE ST_StringProperty EXTENDS ST_OcaProperty :
STRUCT
  stVal: ST_OcaString;
END_STRUCT
END_TYPE

```

ST_SubscriptionManagerState

```

TYPE ST_SubscriptionManagerState EXTENDS ST_OcaProperty :
STRUCT
  eVal: E_OcaSubscriptionManagerState;
END_STRUCT
END_TYPE

```

ST_TemperatureProperty

```

TYPE ST_TemperatureProperty EXTENDS ST_OcaProperty:STRUCT
  stVal: ST_OcaTemperature;
END_STRUCT
END_TYPE

```

See also: [ST_OcaTemperature \[► 57\]](#)

ST_UINT16Property

```

TYPE ST_UINT16Property EXTENDS ST_OcaProperty :
STRUCT
  nVal: UINT;
END_STRUCT
END_TYPE

```

ST_UDINT32Property

```

TYPE ST_UDINT32Property EXTENDS ST_OcaProperty :
STRUCT
  nVal: UDINT;
END_STRUCT
END_TYPE

```

ST_UINT8Property

```

TYPE ST_UINT8Property EXTENDS ST_OcaProperty :
STRUCT
  nVal: SINT;
END_STRUCT
END_TYPE

```

7.2.2 E_OcaStatus

The status code that identifies the result of the method invocation the response belongs to.

E_OcaStatus has a size of 1 byte.

```

(*Status codes returned from method calls*)
{attribute 'qualified_only'}
TYPE E_OcaStatus :
(
  Ok := 0,
  ProtocolVersionError := 1,

```



```

DeviceError := 2,
Locked := 3,
BadFormat := 4,
BaGOno := 5,
ParameterError := 6,
ParameterOutOfRange := 7,
NotImplemented := 8,
InvalidRequest := 9,
ProcessingFailed := 10,
BadMethod := 11,
PartiallySucceeded := 12,
Timeout := 13,
BufferOverflow := 14,
DecodingError := 20,
Undefined := 21
) BYTE;
END_TYPE

```

7.2.3 E_OcaMuteState

```

{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_OcaMuteState:
(
    Muted:= 1,
    Unmuted:=2
);
END_TYPE

```

7.2.4 ST_OcaTemperature

```

TYPE ST_OcaTemperature :
STRUCT
    fDegreesCelsius: REAL; //Value in Degrees Celsius
END_STRUCT
END_TYPE

```

7.2.5 ST_OcaDeviceInfo

Structure used to show informations about OCA devices.

```

TYPE ST_OcaDeviceInfo:
STRUCT
    tDeviceEnabledSince:    DATE_AND_TIME;
    tClientConnectedSince: DATE_AND_TIME; //yet not used - for further extensions
    tLastSentMsg:           DATE_AND_TIME;
    tLastReceivedMsg:      DATE_AND_TIME;
    aSentMessages:         ARRAY[E_OcaMessageType.OcaCmd..E_OcaMessageType.OcaKeepAlive] OF UDINT;
    aReceivedMessages:     ARRAY[E_OcaMessageType.OcaCmd..E_OcaMessageType.OcaKeepAlive] OF UDINT;
END_STRUCT
END_TYPE

```

7.2.6 E_OcaMessageType

Indicates the type of the message

```

{attribute 'qualified_only'}
TYPE E_OcaMessageType:
(
    OcaCmd= 0,           // Command - no Response Required
    OcaCmdRrq= 1,       // Command - Response Required
    OcaNtf= 2,          // Notification
    OcaRsp= 3,          // Response (to a command or notification)
    OcaKeepAlive= 4,    // Keep-alive message used for device supervision.
    Idle:=7
) BYTE;
END_TYPE

```

8 Examples

8.1 Example for using the function block FB_OcaDevice

The example shows how to handle and parameterize the function block FB_OcaDevice. The function block FB_OcaDevice forms the basis for the use of further function blocks that can be used to read and modify OCA objects of an OCA device.

The screenshot displays the TwinCAT development environment with three main windows:

- Top Window (Main.A_Operation [Online]):** Shows a ladder logic diagram for the `fbOcaDevice` function block. The `bEnable` input is set to `True`. Other inputs include `ipResultEvent` (set to `0`), `pLink` (set to `0xFFFFFFFFA80233C4B8`), `bBusy` (set to `True`), `bActive` (set to `True`), `bError` (set to `False`), and `stDeviceInfo`.
- Middle Window (Main.A_Init [Online]):** Shows the initialization code for the function block:


```

1 IF NOT bInit TRUE THEN
2   bInit TRUE := TRUE;
3
4   (* ----- Init devices ----- *)
5   InitDeviceList(); //prepare device
6   fbOcaDevice.sDeviceName := 'Amplifier_01'; //name of the device
7 END_IF
8 RETURN
      
```
- Bottom Window (Main.InitDeviceList [Online]):** Shows the `InitDeviceList` function code:


```

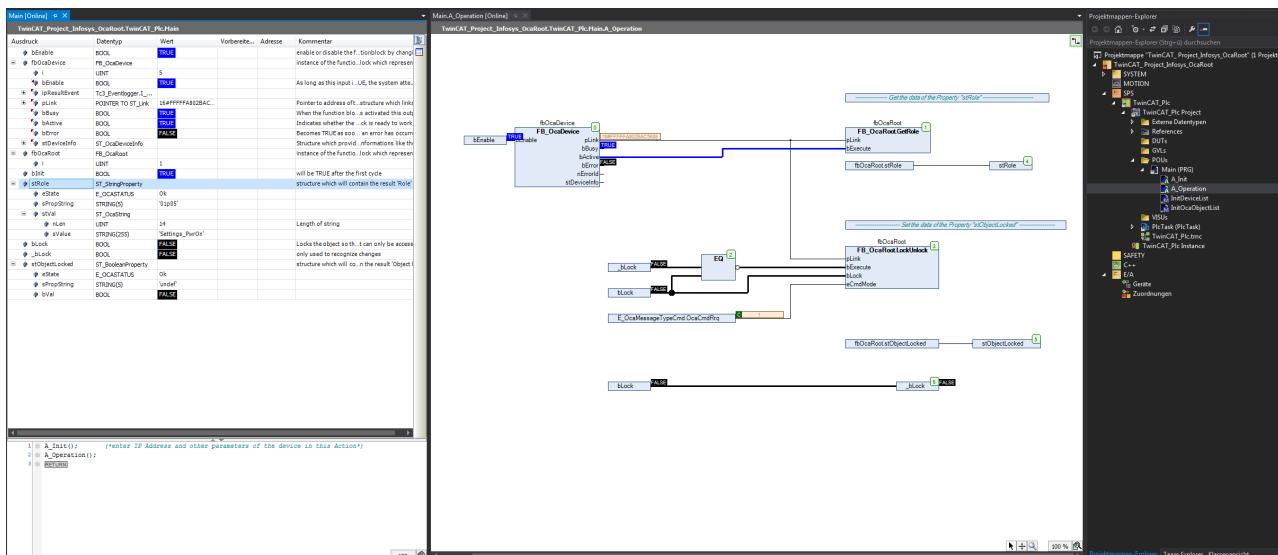
1 GVL_AES70.aOcaDevices[0].ipAddr := '169.254.1.17'; // the ip address of the device
2 GVL_AES70.aOcaDevices[0].ipPort := 30016; // the port of the device
3 GVL_AES70.aOcaDevices[0].sDeviceName := 'Amplifier_1'; // the unique name of the device
4
5
6 GVL_AES70.aOcaDevices[1].ipAddr := '169.254.1.7'; // the ip address of the device
7 GVL_AES70.aOcaDevices[1].ipPort := 30013; // the port of the device
8 GVL_AES70.aOcaDevices[1].sDeviceName := 'Amplifier_01'; // the unique name of the device
9
10 //add your devices here
      
```
- Right Window (Projektmappen-Explorer):** Shows the project structure for `TwinCAT_Project_Infosys_OcaDevice`, including folders for `SYSTEM`, `MOTION`, `SPS`, `POUs`, and `MAIN`.

This example assumes that a device that supports the AES70 standard is connected.

Download: https://infosys.beckhoff.com/content/1033/tf8810_tc3_aes70/Resources/4223793163/.zip

8.2 Example for using the function block FB_OcaRoot

The example shows how the function block FB_OcaRoot can be used. The function block FB_OcaRoot provides basic functionalities and passes these on to all function blocks of the library that are used to read and modify OCA objects such as FB_OcaMute, FB_OcaSwitch etc.

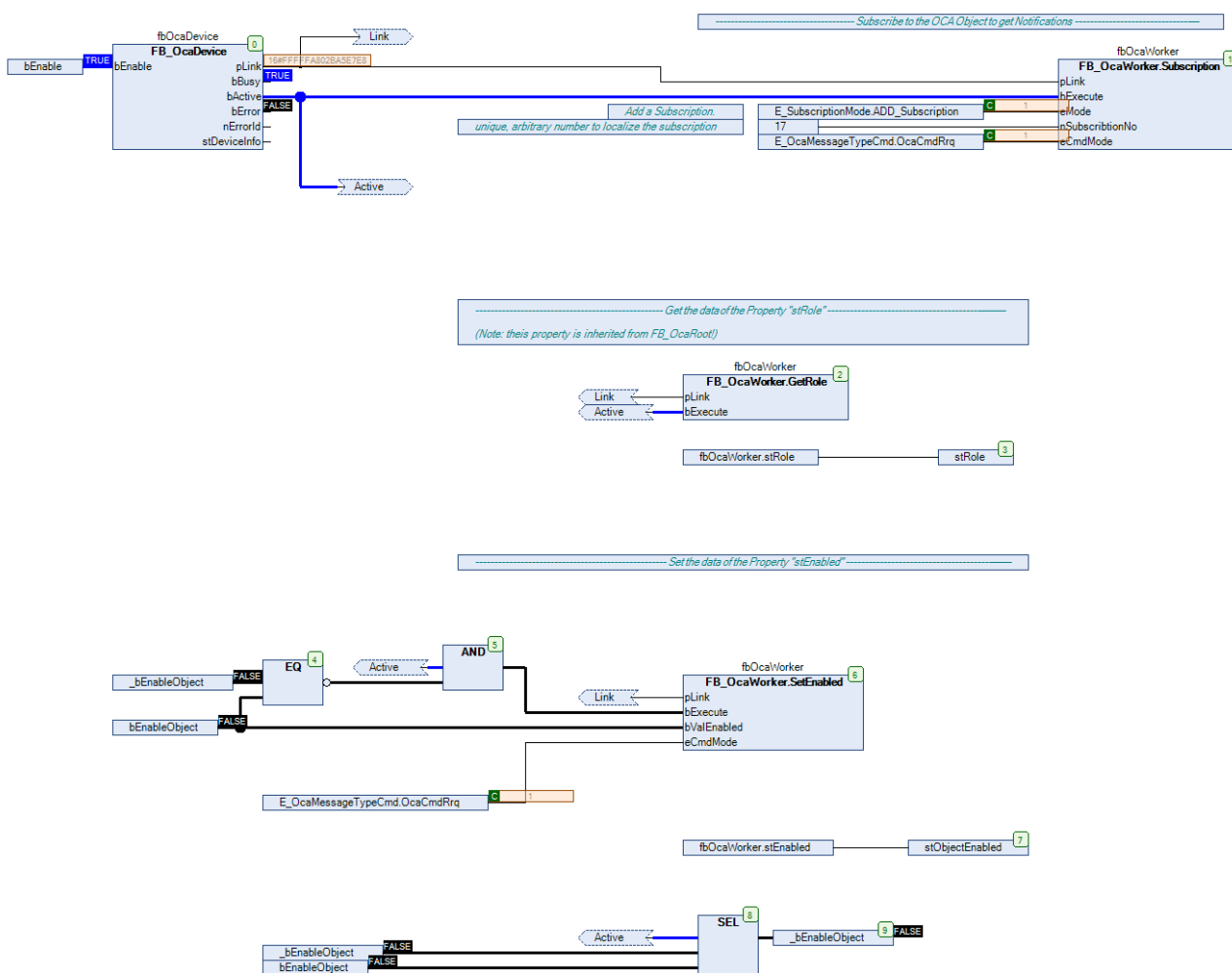


This example assumes that a device that supports the AES70 standard is connected.

Download: https://infosys.beckhoff.com/content/1033/tf8810_tc3_aes70/Resources/4223020171.zip

8.3 Example for using the function block FB_OcaWorker

The example shows how the function block FB_OcaWorker can be used.

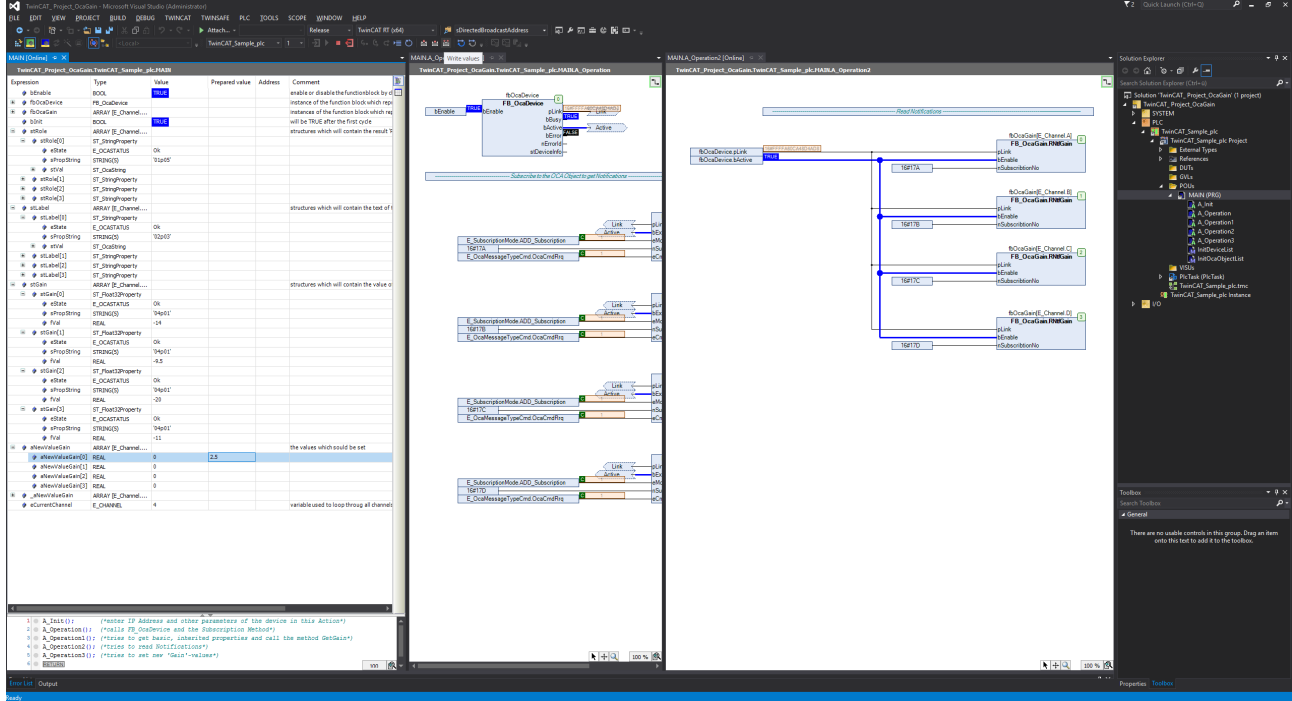


This example assumes that a device that supports the AES70 standard is connected.

Download: https://infosys.beckhoff.com/content/1033/tf8810_tc3_aes70/Resources/4223021835/.zip

8.4 Example for using the function block FB_OcaGain

The example shows how to handle and parameterize the function block FB_OcaGain. Methods and properties that are inherited from FB_OcaWorker and therefore also from FB_OcaRoot are also used.



This example assumes that a device that supports the AES70 standard is connected.

Download: https://infosys.beckhoff.com/content/1033/tf8810_tc3_aes70/Resources/4223791499/.zip

9 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/tf8810

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

