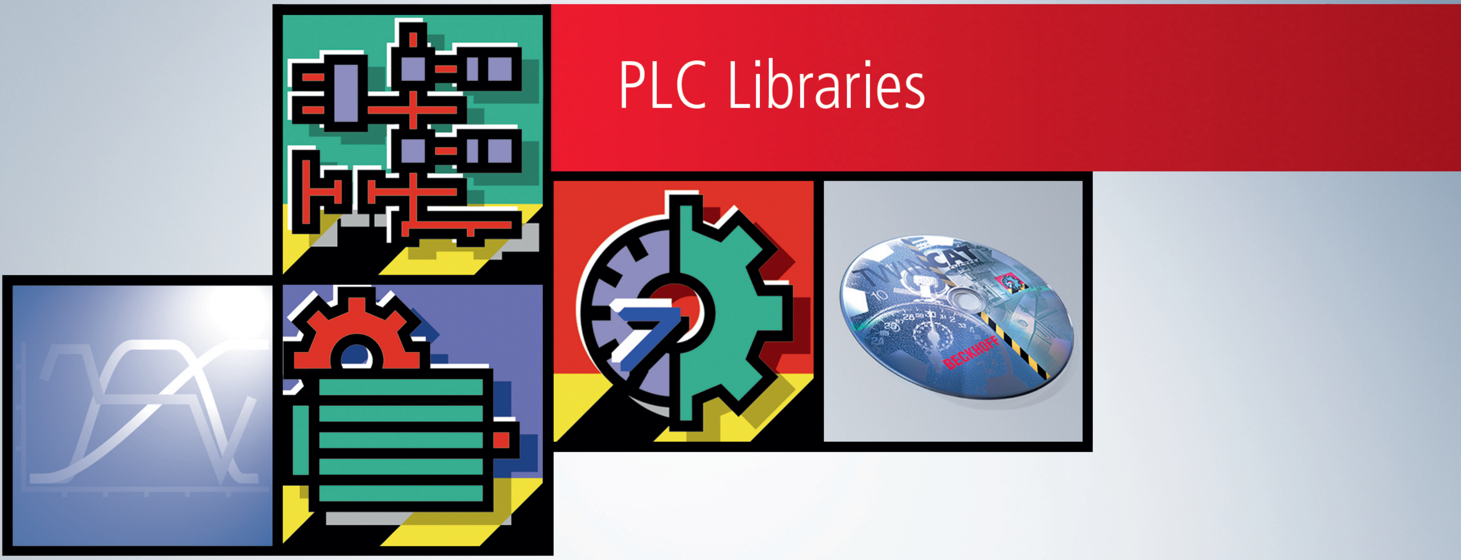
BECKHOFF New Automation Technology

Manual | EN

TX1200

TwinCAT 2 | PLC-Bibliothek: TcDataExchange



2022-10-14 | Version: 1.1

Table of contents

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

2 Introduction ... 8

3 Programming ... 9
3.1 Write/Read Blocks ... 9
3.1.1 FB_ReadAdsSymByName ... 9
3.1.2 FB_WriteAdsSymByName ... 10
3.1.3 FB_WriteBoolOnDelta ... 11
3.1.4 FB_WriteByteOnDelta ... 13
3.1.5 FB_WriteWordOnDelta ... 14
3.1.6 FB_WriteDWordOnDelta ... 15
3.1.7 FB_WriteRealOnDelta ... 16
3.1.8 FB_WriteLRealOnDelta ... 18
3.2 Monitoring Blocks ... 19
3.2.1 FB_CheckWatchdog ... 19
3.2.2 FB_WriteWatchdog ... 20
3.3 Data types ... 21
3.3.1 E_AdsComMode ... 21

4 Appendix ... 22
4.1 Support and Service ... 22

1 Foreword

1.1 Notes on the documentation

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

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

Disclaimer

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

Trademarks

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

Patent Pending

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

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

Copyright

1.2 Safety instructions

Safety regulations

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

Exclusion of liability

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

Personnel qualification

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

Description of symbols

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

⚠ DANGER

Serious risk of injury!

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

⚠ WARNING

Risk of injury!

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

⚠ CAUTION

Personal injuries!

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

NOTE

Damage to the environment or devices

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

Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Notes on information security

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

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

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

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

2 Introduction

The present function blocks simplify data exchange between the TwinCAT PLC runtime system and/or other ADS devices (TwinCAT NC, Bus Terminal Controller, ...).

The *FB_WriteXXXOnDelta()* function blocks implement a write procedure when the input signal rises above or falls below a specified limit value. The frequency with which the input signal is examined can be set. Event-driven data writing minimises the loading on the fieldbus. If an error occurs during transmission, the process is repeated until the connection is established once more. All data types supported in the TwinCAT PLC are permitted as source and destination variables. Symbol names are also supported.

The FB_XXXAdsSymByName() function blocks execute a procedure to write or read the name of the PLC variable.

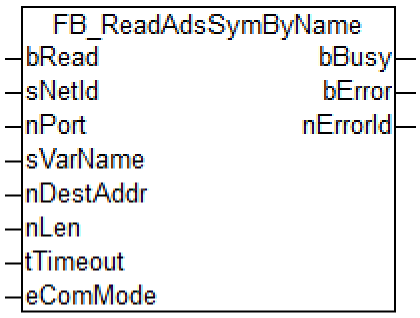
Watchdog blocks are available to monitor individual communication partners. The device that is to be monitored cyclically transmits an incrementing counter. A check is made at the receiver to see that the counter state changes within a specific time.

3 Programming

3.1 Write/Read Blocks

Name	Description
FB_ReadAdsSymByName [▶ 9]	Reads a variable of any desired data type by variable name
FB_WriteAdsSymByName [▶ 10]	Writes a variable of any desired data type by variable name
FB_WriteBoolOnDelta [▶ 11]	Writes a variable of type BOOLEAN in response to an event.
FB_WriteByteOnDelta [▶ 13]	Writes a variable of type BYTE in response to an event.
FB_WriteWordOnDelta [▶ 14]	Writes a variable of type WORD in response to an event.
FB_WriteDWordOnDelta [▶ 15]	Writes a variable of type DWORD in response to an event.
FB_WriteRealOnDelta [▶ 16]	Writes a variable of type REAL in response to an event.
FB_WriteLRealOnDelta [▶ 18]	Writes a variable of type LREAL in response to an event.

3.1.1 FB_ReadAdsSymByName



The block enables the reading of any value from another controller with the aid of the symbol name.

On a rising edge at the *bRead* input the block reads the value of the variable *sVarName* from the selected ADS device (e.g. PLC). The ADS device is indicated by the AMS-NetId (*sNetId*) and the AMS Port number (*nPort*). The value is written into the variable to which *nDestAddr* points.

The internal mode of operation of the block can be changed with the aid of the *eComMode* input:

eComMode := eAdsComModeSecureCom: Following each read procedure the handle of the PLC variable is released again. This mode should be used when values are exchanged very slowly.

eComMode := eAdsComModeFastCom: As long as the *sVarName*, *sNetID* and *nPort* inputs do not change, the handle of the PLC variable will not be released after each read procedure. This mode should be used when values are exchanged very frequently.

VAR_INPUT

```
bRead         : BOOL;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort  := AMSPORT_R0_PLC_RTS1;
sVarName      : STRING;
nDestAddr     : DWORD;
nLen          : UDINT;
tTimeout      : TIME  := DEFAULT_ADS_TIMEOUT;
eComMode      : E_AdsComMode  := eAdsComModeSecureCom;
```

bRead: This block reads the contents of the *sVarName*	**sNetId:** AMS NetId of the ADS device from	**nPort:** AMS Port number of the ADS device from	**sVarName:** Symbol name of the variable to	**nDestAddr:** Address of the variable into which	**nLen:** Length of the variable to be read in bytes.	**tTimeout:** Time until processing is aborted.	**eComMode**: Enum used to specify whether the handle of

variable of the selected ADS device and writes it into the variable to which pointer *nDestAddr* points.	which the value is to be read.	which the value is to be read.	be read on the selected ADS device.	the read value is written.			the PLC variable is released again after each read procedure.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
nErrorId      : UDINT;
```

bBusy: Transmission is active.	**bError:** An error occurred during the transmission.	**nErrorId:** ADS error number if an error has occurred.

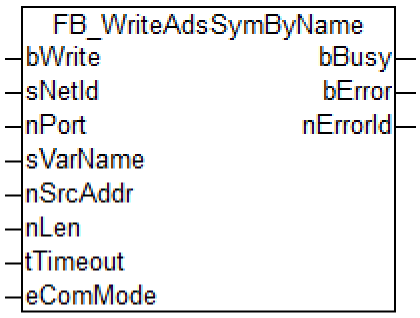
Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from Build 2247	PC/CX	TcDataExchange library from V1.1.0

Also see about this

- E_AdsComMode [▸ 21]

3.1.2 FB_WriteAdsSymByName



Writing any desired value to another controller with the aid of the symbol name.

On a rising edge at the *bWrite* input the block writes the value to which the pointer *nSrcAddr* points into the variable *sVarName* of the selected ADS device (e.g. PLC). The ADS device is indicated by the AMS-NetId (*sNetId*) and the AMS Port number (*nPort*).

The internal mode of operation of the block can be changed with the aid of the *eComMode* input:

eComMode := eAdsComModeSecureCom: Following each write procedure the handle of the PLC variable is released again. This mode should be used when values are exchanged very slowly.

eComMode := eAdsComModeFastCom: As long as the *sVarName*, *sNetID* and *nPort* inputs do not change, the handle of the PLC variable will not be released after each write procedure. This mode should be used when values are exchanged very frequently.

VAR_INPUT

```
bWrite        : BOOL;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
```

```
sVarName      : STRING;
nSrcAddr      : DWORD;
nLen          : UDINT;
tTimeout      : TIME := DEFAULT_ADS_TIMEOUT;
eComMode      : E_AdsComMode := eAdsComModeSecureCom;
```

bWrite: This block writes the contents of the variable to which pointer *nSrcAddr* points into the variable *sVarName* of the selected ADS device.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS Port number of the ADS device to which the value is to be transmitted.	**sVarName:** Symbol name of the variable to be written on the selected ADS device.	**nSrcAddr:** Address of the variable in which the value to written is located.	**nLen:** Length in bytes of the variable to be written.	**tTimeout:** Time until processing is aborted.	**eComMode**: Enum used to specify whether the handle of the PLC variable is released again after each write procedure.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
nErrorId      : UDINT;
```

bBusy: Transmission is active.	**bError:** An error occurred during the transmission.	**nErrorId:** ADS error number if an error has occurred.

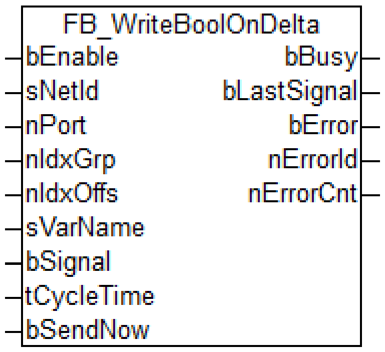
Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64 from Build 2247	PC/CX	TcDataExchange library from V1.1.0

Also see about this

E_AdsComMode [21]

3.1.3 FB_WriteBoolOnDelta



Event-driven writing of a variable of type BOOLEAN.

The *FB_WriteBoolOnDelta()* function block checks cyclically whether the value at the *bSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If a change is detected, the value of the signal is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable       : BOOL := FALSE;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp       : UDINT;
nIdxOffs      : UDINT;
sVarName      : STRING;
bSignal       : BOOL;
tCycleTime    : TIME := t#0s;
bSendNow      : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS- port number of the ADS device to which the value is to be transmitted.	**nIdxGrp:** Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs:** Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**bSignal:** Variable whose value is to be transmitted.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

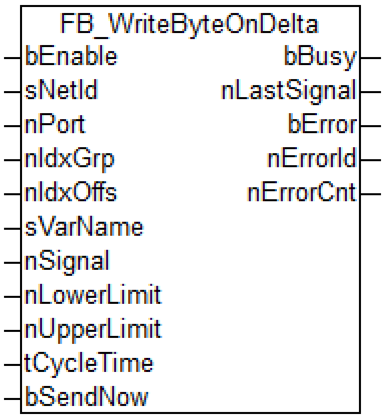
```
bBusy         : BOOL;
bLastSignal   : BOOL;
bError        : BOOL;
nErrorId      : UDINT;
nErrorCnt     : UDINT;
```

bBusy: Transmission is active.	**bLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.1.4 FB_WriteByteOnDelta



Event-driven writing of a variable of type BYTE.

The *FB_WriteByteOnDelta()* function block checks cyclically whether the value at the *nSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If the comparison establishes that the current value is greater than *nUpperLimit* or lower than the value *nLowerLimit*, then the value is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable      : BOOL := FALSE;
sNetId       : T_AmsNetId;
nPort        : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp      : UDINT;
nIdxOffs     : UDINT;
sVarName     : STRING;
nSignal      : BYTE;
nLowerLimit  : BYTE;
nUpperLimit  : BYTE;
tCycleTime   : TIME := t#0s;
bSendNow     : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS-port number of the ADS device to which the value is to be transmitted.	**nIdxGrp**: Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs**: Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**nSignal:** Variable whose value is to be transmitted.	**nLowerLimit:** Lower limit value.	**nUpperLimit:** Upper limit value.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

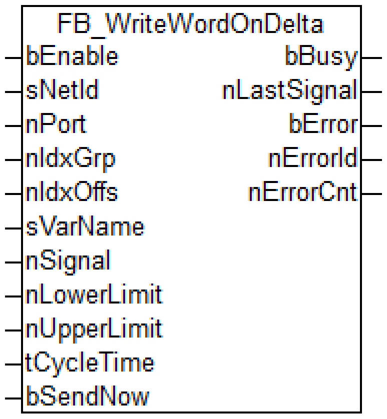
```
bBusy        : BOOL;
nLastSignal  : BYTE;
bError       : BOOL;
nErrorId     : UDINT;
nErrorCnt    : UDINT;
```

bBusy: Transmission is active.	**bLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.1.5 FB_WriteWordOnDelta



Event-driven writing of a variable of type WORD.

The *FB_WriteWordOnDelta()* function block checks cyclically whether the value at the *nSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If the comparison establishes that the current value is greater than *nUpperLimit* or lower than the value *nLowerLimit*, then the value is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable       : BOOL := FALSE;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp       : UDINT;
nIdxOffs      : UDINT;
sVarName      : STRING;
nSignal       : BYTE;
nLowerLimit   : BYTE;
nUpperLimit   : BYTE;
tCycleTime    : TIME := t#0s;
bSendNow      : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS-port number of the ADS device to which the value is to be transmitted.	**nIdxGrp:** Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs:** Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**nSignal:** Variable whose value is to be transmitted.	n**Lower Limit:** Lower limit value.	**nUpper Limit:** Upper limit value.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

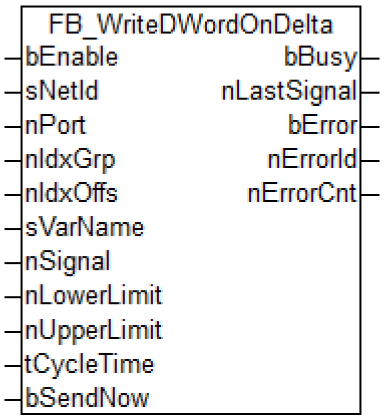
```
bBusy         : BOOL;
nLastSignal   : BYTE;
bError        : BOOL;
nErrorId      : UDINT;
nErrorCnt     : UDINT;
```

bBusy: Transmission is active.	**bLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.1.6 FB_WriteDWordOnDelta



Event-driven writing of a variable of type DWORD.

The *FB_WriteDWordOnDelta()* function block checks cyclically whether the value at the *nSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If the comparison establishes that the current value is greater than *nUpperLimit* or lower than the value *nLowerLimit*, then the value is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable       : BOOL := FALSE;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp       : UDINT;
nIdxOffs      : UDINT;
sVarName      : STRING;
nSignal       : BYTE;
nLowerLimit       : BYTE;
nUpperLimit       : BYTE;
tCycleTime        : TIME := t#0s;
bSendNow      : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS-port number of the ADS device to which the value is to be transmitted.	**nIdxGrp**: Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs**: Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**nSignal:** Variable whose value is to be transmitted.	**nLower Limit:** Lower limit value.	**nUpper Limit:** Upper limit value.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

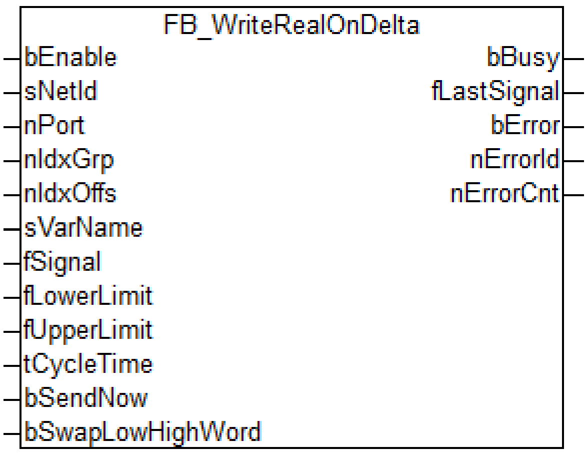
```
bBusy         : BOOL;
nLastSignal   : BYTE;
bError        : BOOL;
nErrorId      : UDINT;
nErrorCnt     : UDINT;
```

bBusy: Transmission is active.	**bLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.1.7 FB_WriteRealOnDelta



Event-driven writing of a variable of type REAL.

The *FB_WriteRealOnDelta()* function block checks cyclically whether the value at the *fSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If the comparison establishes that the current value is greater than *fUpperLimit* or lower than the value *fLowerLimit*, then the value is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

The internal representation of floating-point numbers differs according to the hardware being used. Whereas Intel use the "little-endian" format, Motorola-based hardware employs the "big-endian" format. The input variable *bSwapLowHighWord* can be used to make the necessary adjustment to be able to exchange floating point numbers. This is necessary, for instance, if floating point numbers need to be exchanged between the TwinCAT PLC running on a PC and a BC9000.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable           : BOOL := FALSE;
sNetId            : T_AmsNetId;
nPort             : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp           : UDINT;
nIdxOffs          : UDINT;
sVarName          : STRING;
fSignal           : REAL;
fLowerLimit       : REAL;
fUpperLimit       : REAL;
tCycleTime        : TIME := t#0s;
bSendNow          : BOOL;
bSwapLowHighWord  : BOOL := FALSE;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS-port number of the ADS device to which the value is to be transmitted.	**nIdxGrp:** Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs:** Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**fSignal:** Variable whose value is to be transmitted.	**fLowerLimit:** Lower limit value.	**fUpperLimit:** Upper limit value.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.	**bSwapLowHighWord:** The least significant WORD and the most significant WORD are swapped.

VAR_OUTPUT

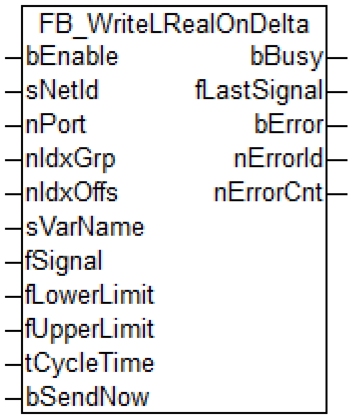
```
bBusy          : BOOL;
fLastSignal    : BYTE;
bError         : BOOL;
nErrorId       : UDINT;
nErrorCnt      : UDINT;
```

bBusy: Transmission is active.	**fLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.1.8 FB_WriteLRealOnDelta



Event-driven writing of a variable of type LREAL.

The *FB_WriteLRealOnDelta()* function block checks cyclically whether the value at the *fSignal* input has changed. The cycle time within which the check is carried out is specified by the parameter *tCycleTime*. If 0 seconds is given for *tCycleTime*, the input signal is examined during every PLC cycle. If the comparison establishes that the current value is greater than *fUpperLimit* or lower than the value *fLowerLimit*, then the value is sent to the ADS device that is to be specified. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

If the *bEnable* input is set to FALSE, no further signal transmission is carried out.

VAR_INPUT

```
bEnable       : BOOL := FALSE;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp       : UDINT;
nIdxOffs      : UDINT;
sVarName      : STRING;
fSignal       : BYTE;
fLowerLimit   : BYTE;
fUpperLimit   : BYTE;
tCycleTime    : TIME := t#0s;
bSendNow      : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS-port number of the ADS device to which the value is to be transmitted.	**nIdxGrp**: Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs**: Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**fSignal:** Variable whose value is to be transmitted.	**fLowerLimit:** Lower limit value.	**fUpperLimit:** Upper limit value.	**tCycleTime:** Cycle period in which the input signal is examined to see that it has changed.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

```
bBusy         : BOOL;
fLastSignal   : BYTE;
bError        : BOOL;
nErrorId      : UDINT;
nErrorCnt     : UDINT;
```

bBusy: Transmission is active.	**fLastSignal:** Most recently transmitted value.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

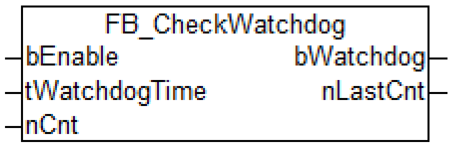
Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX	TcDataExchange-Bibliothek from V1.0.0

3.2 Monitoring Blocks

Name	Description
FB_CheckWatchdog [▶ 19]	Monitors the received watchdog signal
FB_WriteWatchdog [▶ 20]	Writes a watchdog signal (an incrementing counter) cyclically

3.2.1 FB_CheckWatchdog



Monitoring a watchdog signal transmitted by the FB_WriteWatchdog() [▶ 20] block.

The device to be monitored regularly transmits a changing counter state to the device that is to check the transmission. The function block FB_CheckWatchdog() is used there to monitor the state of the counter. If this does not change within a specific period, the *bWatchdog* output is set to TRUE. If a value of 0 seconds is specified for *tWatchdogTime*, the *bWatchdog* signal is set to FALSE. The period specified by *tWachtdogTime* should be a multiple (5-10 times) of the time in which the monitoring signal is transmitted.

VAR_INPUT

```
bEnable          : BOOL := FALSE;
tWatchdogTime    : TIME := t#0s;
nCnt             : UDINT;
```

bEnable: Enable block.	**tWatchdogTime:** Period within which *nCnt* should change.	**nCnt:** Current counter state of the watchdog signal

VAR_OUTPUT

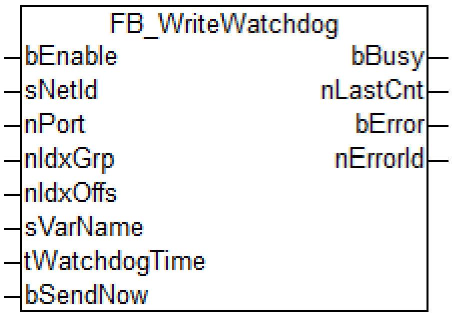
```
bWatchdog     : BOOL;
nLastCnt      : UDINT;
```

bWatchdog: FALSE indicates a valid monitoring signal. The output will become TRUE if no change is detected in *nCnt* during the period specified by *tWatchdogTime*.	**nLastCnt:** Most recent successfully transmitted counter state of the monitoring signal.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.2.2 FB_WriteWatchdog



Writing a watchdog signal to another ADS device (TwinCAT PLC, Bus Terminal Controller, ...).

The FB_WriteWatchdog() function block cyclically writes the contents of a 32-bit counter into another ADS device. The counter is incremented every time the transmission is successful. The FB_CheckWatchdog() [▶ 19] function block can be used at the receiver to evaluate this signal. The receiver is addressed by means of the AMS-NetId and the port number (see also ADS Device Identification). The position within the receiver is specified by the index group/index offset or by the symbol name. Usually this is the input image or the flags area.

The period for *tWachtdogTime* should not be shorter than 1 second, to avoid transmitting the counter state too frequently. If 0 seconds is given for *tWatchdogTime*, the signal is not transmitted. Please also note the description of the FB_CheckWatchdog() [▶ 19] block. If the *bEnable* input is set to FALSE, no further transmission of the watchdog signal is carried out.

VAR_INPUT

```
bEnable       : BOOL := FALSE;
sNetId        : T_AmsNetId;
nPort         : T_AmsPort := AMSPORT_R0_PLC_RTS1;
nIdxGrp       : UDINT;
nIdxOffs      : UDINT;
sVarName      : STRING;
tWatchdogTime : TIME := t#0s;
bSendNow      : BOOL;
```

bEnable: Enable block.	**sNetId:** AMS-NetId of the ADS device to which the value is to be transmitted.	**nPort:** AMS- port number of the ADS device to which the value is to be transmitted.	**nIdxGrp:** Index group within the ADS device into which the value is to be transmitted.	**nIdxOffs:** Index offset within the ADS device into which the value is to be transmitted.	**sVarName:** Symbol name within the ADS device into which the value is to be transmitted.	**tWatchdog Time:** Cycle time with which the watchdog signal is to be transmitted.	**bSendNow:** The value is transmitted immediately in response to a rising edge.

VAR_OUTPUT

```
bBusy        : BOOL;
nLastCnt     : UDINT;
bError       : BOOL;
nErrorId     : UDINT;
```

bBusy: Transmission is active.	**nLastCnt:** Most recently transmitted counter state.	**bError:** An error occurred during the transmission.	**nErrorId:** Error number if an error has occurred.	**nErrorCnt:** Number of transmission attempts that have returned faults.

Requirements

Development environment	Target system	Required libraries
TwinCAT 2.11 R3/x64	PC/CX, BX or BC	TcDataExchange-Bibliothek from V1.0.0

3.3 Data types

3.3.1 E_AdsComMode

```
TYPE E_AdsComMode :
(
  eAdsComModeSecureCom  := 0,
  eAdsComModeFastCom    := 1
);
END_TYPE
```

4 Appendix

4.1 Support and Service

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

Beckhoff's branch offices and representatives

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

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

You will also find further documentation for Beckhoff components there.

Beckhoff Support

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

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

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

Beckhoff Service

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

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

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

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

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

More Information:
www.beckhoff.com/tx1200

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