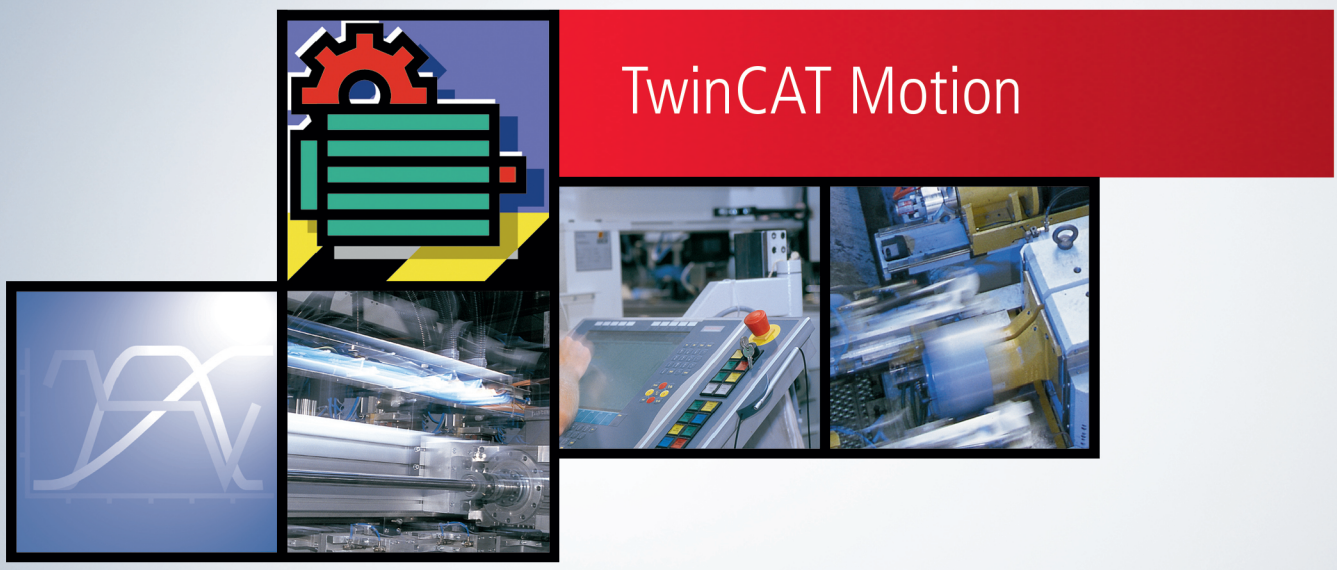
BECKHOFF New Automation Technology

Manual | EN

PLC Library: TcNCDrive

TwinCAT 2 | TX1200, PlcNc, TcNcUtilities



2021-10-08 | Version: 1.0

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 Safety instructions 6
2 POUs of TcNcDrive.lib 7
3 General SoE FB 9
3.1 FB_SoEReset 9
3.2 FB_SoEWritePassword 10
3.3 Command FB 11
3.3.1 FB_SoEExecuteCommand 11
3.3.2 FB_SoEWriteCommandControl 13
3.3.3 FB_SoEReadCommandState 14
3.4 Diagnosis FB 16
3.4.1 FB_SoEReadDiagMessage 16
3.4.2 FB_SoEReadDiagNumber 17
3.4.3 FB_SoEReadDiagNumberList 18
3.4.4 FB_SoEReadClassXDiag 20
3.5 FB for current values 21
3.5.1 FB_SoERead 21
3.5.2 FB_SoEWrite 23
3.5.3 FB_SoEReadAmplifierTemperature 24
3.5.4 FB_SoEReadMotorTemperature 26
3.5.5 FB_SoEReadDcBusCurrent 27
3.5.6 FB_SoEReadDcBusVoltage 28
4 AX5000 specific FB 30
4.1 FB_SoEAX5000ReadActMainVoltage 30
4.2 FB_SoEAX5000SetMotorCtrlWord 31
4.3 FB_SoEAX5000FirmwareUpdate 32
5 F_GetVersionTcNcDrive 36
6 E_FwUpdateState 37

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.

2 POUs of TcNcDrive.lib

This library contains functions and functionblocks for SoE-drives. The access to the drive is done via NC-Axis-Reference (NC_TO_PLC).

The TcNcDrive.lib contains Wrapper-FBs around the FBs of the TcDrive.lib.

There are differences in the usage of the Drive libs in combination with AX5000 and with Bosch Rexroth IndraDriveCS. See samples.

TcNcDrive.lib

The TcNcDrive.lib should be used, if the drive is used with the NC via FBs of the libraries TcNc.lib or TcMc.lib. The FBs of the TcNcDrive.lib use the information of the NC-Axis structure (NC_TO_PLC), that are also used by the FBs of the TcNc.lib or TcMc.lib. The FBs determine via the Nc-AxisID of the NC_TO_PLC structure the access data to the drive (NetID, address and channel number). See samples of the FBs in the documentation of the TcNcDrive.lib.

Hint: The function blocks FB_SoERead and FB_SoEWrite can be used to access any parameter in the drive, that have no special acces FB.

Functions

Name	Description
F_GetVersionTcNcDrive [▶ 36]	This function can be used to read the version information of the PLC library.
F_ConvWordToSTAX5000C1D	See documentation of TcDrive.lib Converts the C1D-Word (S-0-0011) of an AX5000 to a structure ST_AX5000_C1D

Functionblocks

Name	Description
FB_SoEReset [▶ 9]	Execute drive reset (S-0-0099)
FB_SoEWritePassword [▶ 10]	Set drive password (S-0-0267)
FB_SoEReadDiagMessage [▶ 16]	Read diagnostic message (S-0-0095)
FB_SoEReadDiagNumber [▶ 17]	Read diagnostic number (S-0-0390)
FB_SoEReadDiagNumberList [▶ 18]	Read diagnostic number list (up to 30 entries) (S-0-0375)
FB_SoEReadClassXDiag [▶ 20]	Read class 1 diag (S-0-0011) ... class 3 diag (S-0-0013)
FB_SoEExecuteCommand [▶ 11]	Execute command
FB_SoEWriteCommandControl [▶ 13]	Set command control
FB_SoEReadCommandState [▶ 14]	Read command state
FB_SoERead [▶ 21]	Read parameter
FB_SoEWrite [▶ 23]	Write parameter

Name	Description
FB_SoEReadAmplifierTemperature [▶ 24]	Read amplifier temperature (S-0-0384)
FB_SoEReadMotorTemperature [▶ 26]	Read motor temperature (S-0-0383)
FB_SoEReadDcBusCurrent [▶ 27]	Read Dc-Bus-current (S-0-0381)
FB_SoEReadDcBusVoltage [▶ 28]	Read Dc-Bus-Voltage (S-0-0380)
FB_SoEAX5000ReadActMainVoltage [▶ 30]	Read main voltage (P-0-0200)
FB_SoEAX5000SetMotorCtrlWord [▶ 31]	Set motor control word to override brake handling (P-0-0096)
FB_SoEAX5000FirmwareUpdate [▶ 32]	Automatic firmware update of AX5000

Sample project and configuration for AX5000 drive diagnose

See https://infosys.beckhoff.com/content/1033/tcplclibncdrive/Resources/pro/10954810635.pro, https://infosys.beckhoff.com/content/1033/tcplclibncdrive/Resources/tsm/10954813579.tsm

Sample project and configuration for IndraDriveCS drive diagnose

See https://infosys.beckhoff.com/content/1033/tcplclibncdrive/Resources/pro/10954814987.pro, https://infosys.beckhoff.com/content/1033/tcplclibncdrive/Resources/tsm/10954816395.tsm (TcNcDrive.lib v0.0.25 or higher)

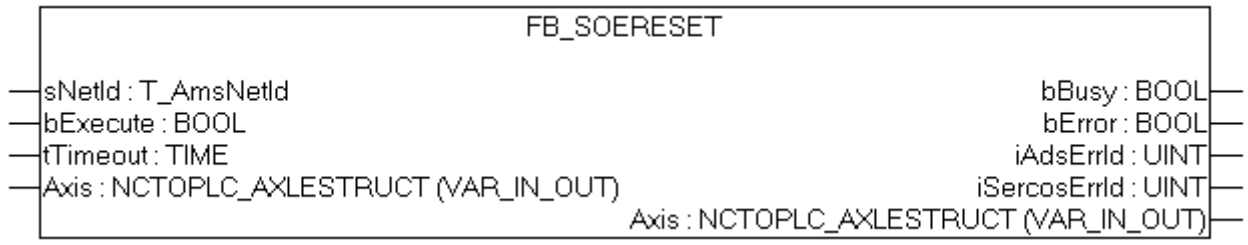
Requirements

Requirements

Component	Version
TwinCAT on the development PC	2.10 Build 1335 or higher
TwinCAT on the Windows CE-Image	2.10 Build 1333 or higher
TwinCAT on the Windows XP-Image	2.10 Build 1333 or higher

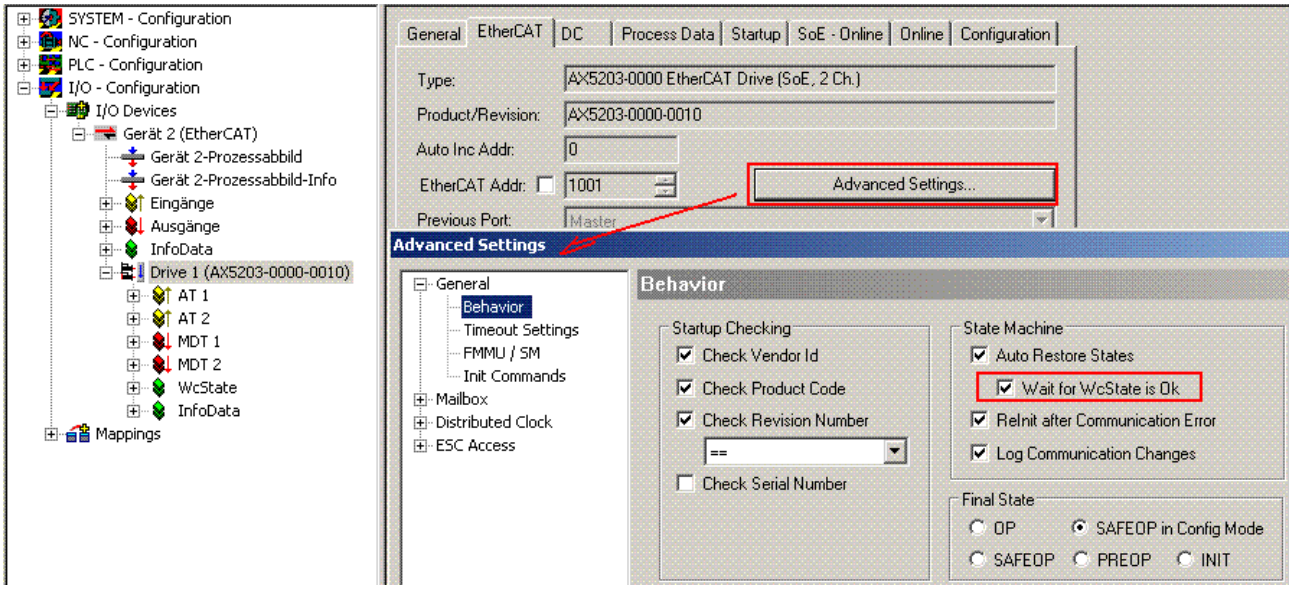
3 General SoE FB

3.1 FB_SoEReset



The functionblock FB_SoEReset can be used to execute a drive reset (S-0-0099). Drives with more than on channel may require a reset on all channels. The timeout time must be 10s, because the reset can take up to 10s.

For the AX5000 the flag "Wait For WcState is OK" in the Advanced EtherCAT Settings has to be active.



An NC-Reset is not executed.

If an NC-Reset is necessary it can be executed via the MC_Reset block from the TcMc.lib

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

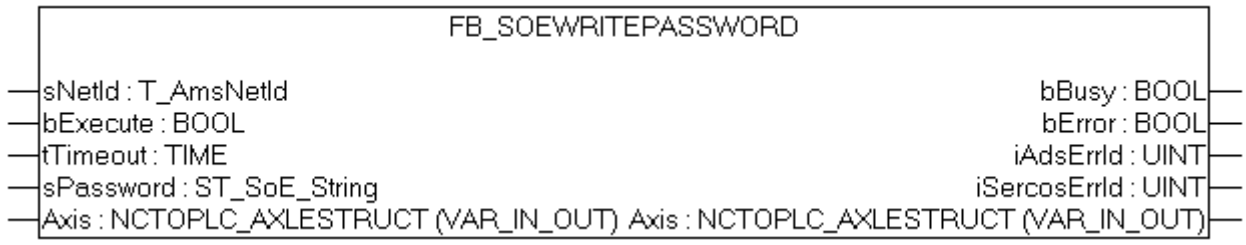
iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbSoEReset : FB_SoEReset_ByDriveRef;
bSoEReset : BOOL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bSoEReset THEN
    fbSoEReset(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
    );
    IF NOT fbSoEReset.bBusy THEN
        fbSoEReset(Axis := stNcToPlc, bExecute := FALSE);
        bSoEReset := FALSE;
    END_IF
END_IF
```

3.2 FB_SoEWritePassword



The functionblock FB_SoEWritePassword can be used to set the drive password (S-0-0267).

VAR_INPUT

```
VAR_INPUT
    sNetId     : T_AmsNetId := '';
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
    sPassword  : ST_SoE_String;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

sPassword: Contains password as Sercos string

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

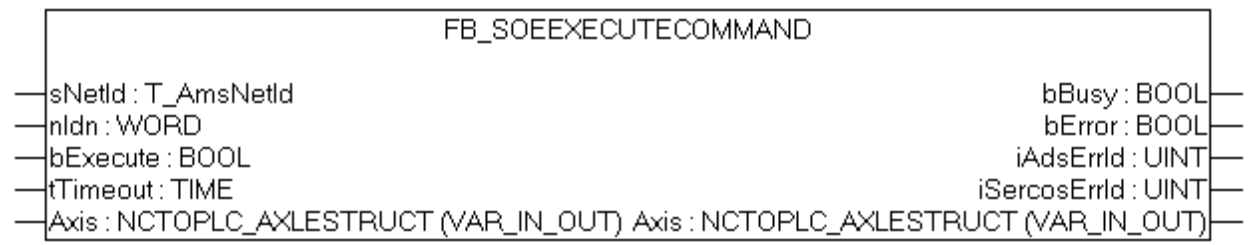
iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbWritePassword : FB_SoEWritePassword;
bWritePassword  : BOOL;
sPassword       :ST_SoE_String;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bWritePassword THEN
    fbWritePassword(
        Axis      := stNcToPlc,
        bExecute  := TRUE,
        tTimeout  := DEFAULT_ADS_TIMEOUT,
        sPassword := sPassword
    );
    IF NOT fbWritePassword.bBusy THEN
        fbWritePassword(Axis := stNcToPlc, bExecute := FALSE);
        bWritePassword := FALSE;
    END_IF
END_IF
```

3.3 Command FB

3.3.1 FB_SoEExecuteCommand



The functionblock FB_SoEExecuteCommand can be used to execute a command.

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    nIdn     : WORD;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

nIdn: Parameter number for the command, i.e. "P_0_IDN + 160" for executing a P-0-0160 command

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

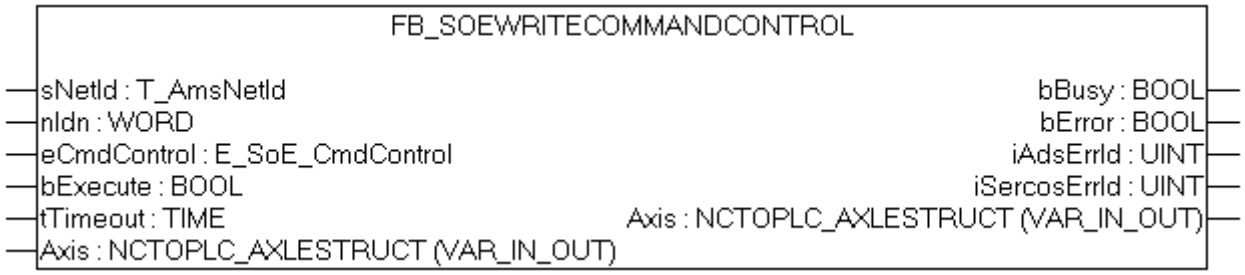
iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbExecuteCommand : FB_SoEExecuteCommand;
bExecuteCommand  : BOOL;
nIdn             : WORD;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bExecuteCommand THEN
    nIdn := P_0_IDN + 160;
    fbExecuteCommand(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        nIdn     := nIdn,
    );
    IF NOT fbExecuteCommand.bBusy THEN
        fbExecuteCommand(Axis := stNcToPlc, bExecute := FALSE);
        bExecuteCommand := FALSE;
    END_IF
END_IF
```

3.3.2 FB_SoEWriteCommandControl



The functionblock FB_SoEWriteCommandControl can be used to prepare, start or cancel a command.

VAR_INPUT

```
VAR_INPUT
    sNetId       : T_AmsNetId := '';
    nIdn         : WORD;
    eCmdControl  : E_SoE_CmdControl;
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

nIdn: Parameter number for the command, i.e. "P_0_IDN + 160" for executing a P-0-0160 command

eCmdControl: prepare a command with eSoE_CmdControl_Set := 1, execute a command with eSoE_CmdControl_SetAndEnable := 3 or abort a command with eSoE_CmdControl_Cancel := 0

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bbBusy        : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

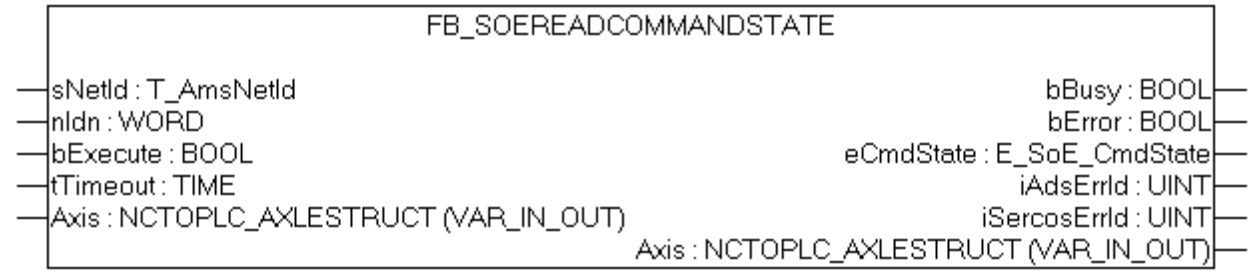
iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbWriteCommandControl : FB_SoEWriteCommandControl;
bWriteCommandControl  : BOOL;
nIdn                  : WORD;
eCmdControl           : E_SoE_CmdControl;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bWriteCommandControl THEN
    nIdn := P_0_IDN + 160;
    fbWriteCommandControl(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        nIdn := nIdn,
        eCmdControl := eCmdControl
    );
    IF NOT fbWriteCommandControl.bBusy THEN
        fbWriteCommandControl(Axis := stNcToPlc, bExecute := FALSE);
        bWriteCommandControl := FALSE;
    END_IF
END_IF
```

3.3.3 FB_SoEReadCommandState



The function block FB_SoEReadCommandState can be used to read the state of a command.

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    nIdn     : WORD;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

nIdn: Parameter number for the command, i.e. "P_0_IDN + 160" for executing a P-0-0160 command

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    eCmdState     : E_SoE_CmdState;
```

```
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

dwAttribute: Supplies the Sercos parameter attribute.

eCmdState: Supplies the command state

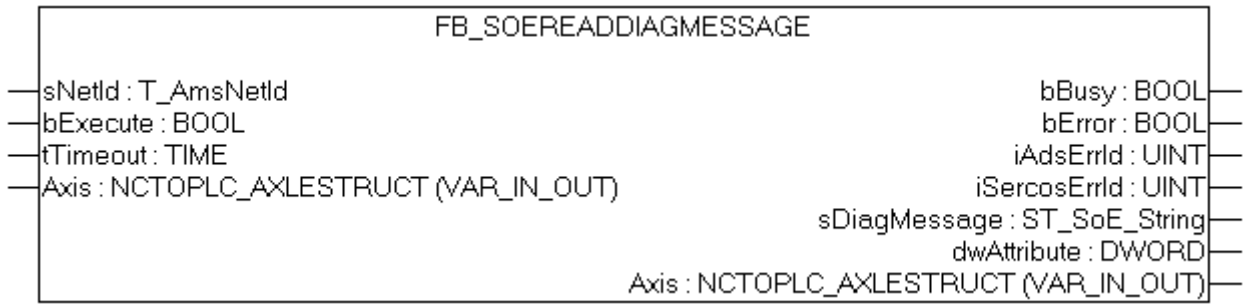
```
    eSoE_CmdState_NotSet =
0
- no command active

    eSoE_CmdState_Set =
1
- command set (prepared) but (not yet) executed

    eSoE_CmdState_Executed =
2
- command was executed

    eSoE_CmdState_SetEnabledExecuted =
3    - command set (prepared) and executed

    eSoE_CmdState_SetAndInterrupted =
5     - command was set but interupted

    eSoE_CmdState_SetEnabledNotExecuted = 7 -
command execution is stll active

    eSoE_CmdState_Error =
15
- error during command execution, switched to error state
```

Sample

```
fbReadCommandState : FB_SoEReadCommandState;
bReadCommandState  : BOOL;
nIdn               : WORD;
eCmdState          :E_SoE_CmdState;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadCommandState THEN
    nIdn := P_0_IDN + 160;
    fbReadCommandState(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        nIdn     := nIdn,
        eCmdState => eCmdState
    );
    IF NOT fbReadCommandState.bBusy THEN
        fbReadCommandState(Axis := stNcToPlc, bExecute := FALSE);
        bReadCommandState := FALSE;
    END_IF
END_IF
```

3.4 Diagnosis FB

3.4.1 FB_SoEReadDiagMessage



The functionblock FB_SoEReadDiagMessage can be used to read the diagnose message as a Sercos-String (S-0-0095).

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
    sDiagMessage  : ST_SoE_String;
    dwAttribute   : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

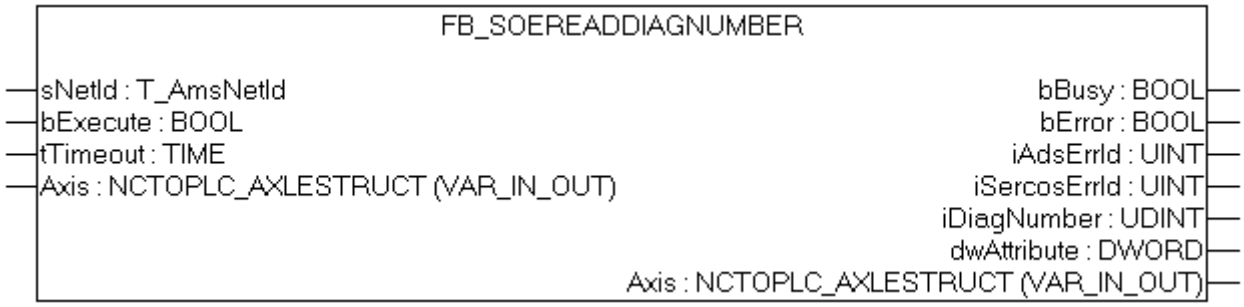
dwAttribute: Supplies the Sercos parameter attribute.

sDiagMessage: Supplies the diagnostic message.

Sample

```
fbDiagMessage : FB_SoEReadDiagMessage;
bDiagMessage  : BOOL;
sDiagMessage  : ST_SoE_String;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bDiagMessage THEN
    fbDiagMessage(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        sDiagMessage=> sDiagMessage
    );
    IF NOT fbDiagMessage.bBusy THEN
        fbDiagMessage(Axis := stNcToPlc, bExecute := FALSE);
        bDiagMessage := FALSE;
    END_IF
END_IF
```

3.4.2 FB_SoEReadDiagNumber



The functionblock FB_SoEReadDiagNumber can be used to read the actual diagnose number as UDINT (S-0-0390).

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
```

```
    iDiagNumber    : UDINT;
    dwAttribute    : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

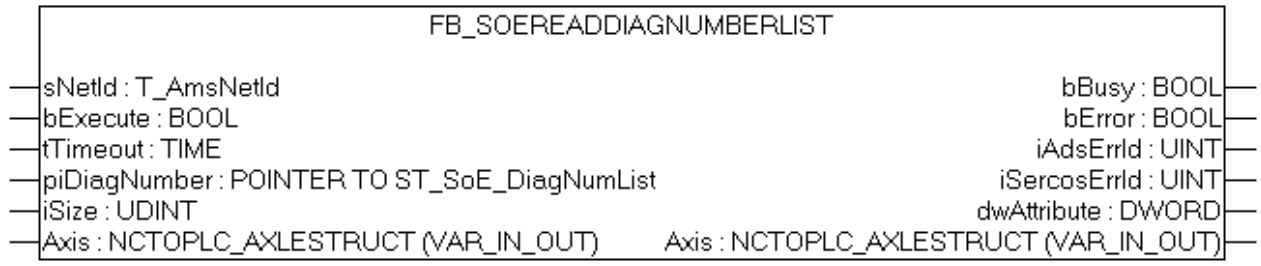
dwAttribute: Supplies the Sercos parameter attribute.

iDiagNumber:Supplies the diagnostic number.

Sample

```
fbDiagNumber : FB_SoEReadDiagNumber;
bDiagNumber  : BOOL;
iDiagNumber  : UDINT;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bDiagNumber THEN
    fbDiagNumber(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        iDiagNumber => iDiagNumber
    );
    IF NOT fbDiagNumber.bBusy THEN
        fbDiagNumber(Axis := stNcToPlc, bExecute := FALSE);
        bDiagNumber := FALSE;
    END_IF
END_IF
```

3.4.3 FB_SoEReadDiagNumberList



The functionblock FB_SoEReadDiagNumberList can be used to read the history of the diagnose numbers as a list (S-0-0375).

VAR_INPUT

```
VAR_INPUT
    sNetId        : T_AmsNetId := '';
    bExecute      : BOOL;
    tTimeout      : TIME := DEFAULT_ADS_TIMEOUT;
    piDiagNumber  : POINTER TO ST_SoE_DiagNumList;
    iSize         : UDINT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

piDiagNumber: Pointer to the list of the last max. 30 error numbers. The list consists of the actual and maximum number of bytes in the list, and of the last 30 list entries

iSize: Size of the list in bytes (as Sizeof())

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
    dwAttribute   : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

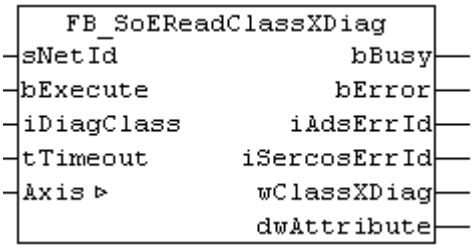
iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

dwAttribute: Supplies the Sercos parameter attribute.

Sample

```
fbDiagNumberList     : FB_SoEReadDiagNumberList;
bDiagNumberList      : BOOL;
stDiagNumberList     : ST_SoE_DiagNumList;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bDiagNumberList  THEN
    fbDiagNumberList(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        piDiagNumber:= ADR(stDiagNumberList),
        iSize := SIZEOF(stDiagNumberList),
    );
    IF NOT fbDiagNumberList.bBusy THEN
        fbDiagNumberList(Axis := stNcToPlc, bExecute := FALSE);
        bDiagNumberList := FALSE;
    END_IF
END_IF
```

3.4.4 FB_SoEReadClassXDiag



The function block FB_SoEReadClassXDiag can be used to read the actual Class 1 Diagnose (S-0-0011) ... Class 3 Diagnose (S-0-0013) as a WORD. There is a conversion function F_ConvWordToSTAX5000C1D for evaluation of the Class 1 Diagnose as a structure ST_AX5000_C1D. See Documentation of TcDrive.lib.

VAR_INPUT

```
VAR_INPUT
    sNetId     : T_AmsNetId := '';
    bExecute   : BOOL;
    iDiagClass : USINT:= 1; (* 1: C1D (S-0-0011) is default, 2: C2D (S-0-0012), 3: C3D (S-0-0013) *)
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

iDiagClass: Selects which diagnose should be read. The diagnose info can be different with different drive vendors. Not always all diagnose parameters (C1D ... C3D) or all bits in these parameters are implementiert.
1: Error: Class 1 Diag (S-0-0011)
2: Warning: Class 2 Diag (S-0-0012)
3: Informationen: Class 3 Diag (S-0-0013)

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
    wClassXDiag   : WORD;
    dwAttribute   : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

wClassXDiag: Supplies the class X diagnose.

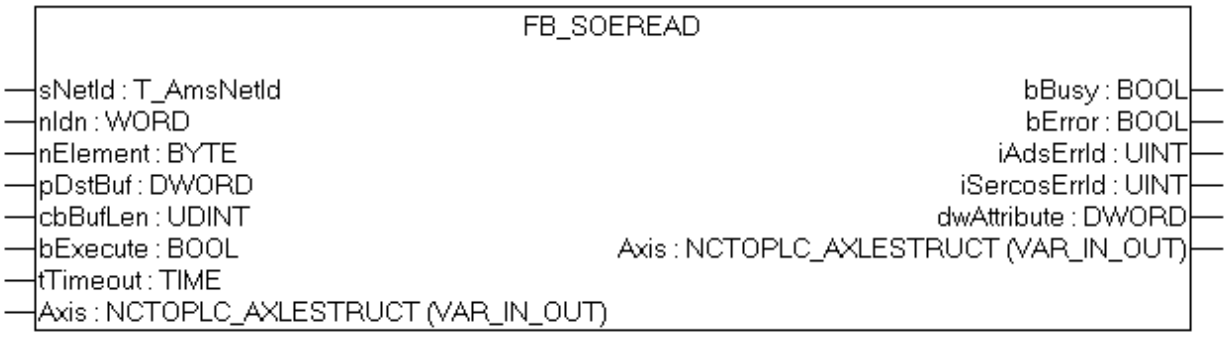
dwAttribute: Supplies the Sercos parameter attribute.

Sample

```
fbClassXDiag : FB_SoEReadClassXDiag;
bClassXDiag  : BOOL;
iDiagClass   : USINT := 1;
wClass1Diag  : WORD;
stAX5000C1D  : ST_AX5000_C1D;
wClass2Diag  : WORD;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bClassXDiag THEN
    fbClassXDiag(
        Axis       := stNcToPlc,
        bExecute   := TRUE,
        iDiagClass := iDiagClass,
        tTimeout   := DEFAULT_ADS_TIMEOUT
    );
    IF NOT fbClassXDiag.bBusy THEN
        fbClassXDiag(Axis := stNcToPlc, bExecute := FALSE);
        bClassXDiag := FALSE;

        CASE
fbClassXDiag.iDiagClass OF
        1:

           wClass1Diag := fbClassXDiag.wClassXDiag;
           stAX5000C1D := F_ConvWordToSTAX5000C1D(wClass1Diag);

        2:
           wClass2Diag := fbClassXDiag.wClassXDiag;
        END_CASE
    END_IF
END_IF
```

3.5 FB for current values

3.5.1 FB_SoERead



The functionblock FB_SoERead can be used to read a parameter.

VAR_INPUT

```
VAR_INPUT
    sNetId     : T_AmsNetId := '';
    nIdn       : WORD;
    nElement   : BYTE;
    pDstBuf    : DWORD;
    cbBufLen   : UDINT;
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
    sPassword  : ST_SoE_String;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

nIdn: Parameter number for the command, i.e. "S_0_IDN + 33" for executing a S-0-0033 command

nElement: Shows on which parameter part is to be read, e.g. 16#40 is the value of the parameter.

```
    EC_SOE_ELEMENT_DATASTATE :BYTE :=16#01;
    EC_SOE_ELEMENT_NAME      :BYTE :=16#02;
    EC_SOE_ELEMENT_ATTRIBUTE :BYTE :=16#04;
    EC_SOE_ELEMENT_UNIT      :BYTE :=16#08;
    EC_SOE_ELEMENT_MIN       :BYTE :=16#10;
    EC_SOE_ELEMENT_MAX       :BYTE :=16#20;
    EC_SOE_ELEMENT_VALUE     :BYTE :=16#40;
    EC_SOE_ELEMENT_DEFAULT   :BYTE :=16#80;
```

pSrcBuf: Contains the address of the buffer containing the data to be read.

cbBufLen: The maximum available buffer size for the data to be read, in bytes.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
    dwAttribute   : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

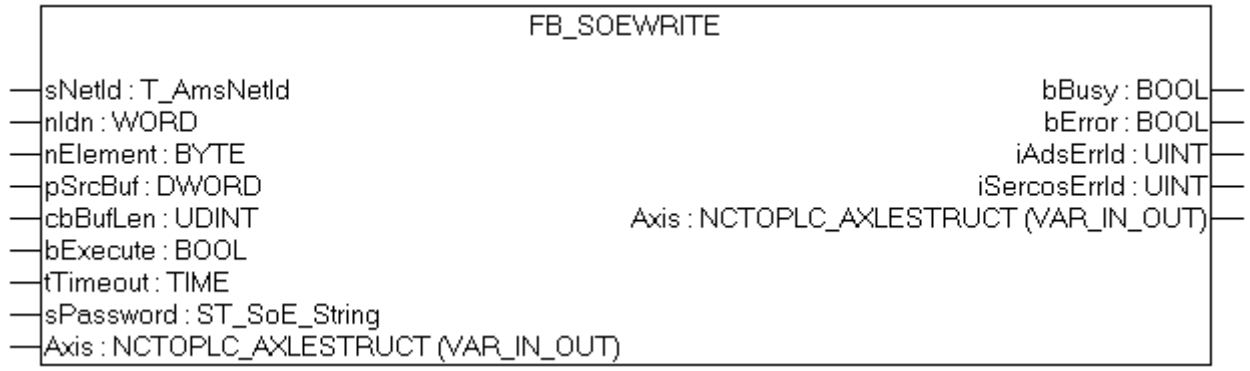
dwAttribute: Supplies the Sercos parameter attribute.

Sample

```
fbRead : FB_SoERead;
bRead  : BOOL;
iReadValue : UINT;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bRead THEN
    nIdn := S_0_IDN + 33;
    fbRead(
        Axis     := stNcToPlc,
        nIdn     := nIdn,
        nElement := 16#40,
        pDstBuf  := ADR(iReadValue),
        cbBufLen := SIZEOF(iReadValue),
```

```
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
    );
    IF NOT fbRead.bBusy THEN
        fbRead(Axis := stNcToPlc, bExecute := FALSE);
        bRead := FALSE;
    END_IF
END_IF
```

3.5.2 FB_SoEWrite



The functionblock FB_SoEWrite can be used to write a parameter.

VAR_INPUT

```
VAR_INPUT
    sNetId      : T_AmsNetId := '';
    nIdn        : WORD;
    nElement    : BYTE;
    pSrcBuf     : DWORD;
    cbBufLen    : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
    sPassword   : ST_SoE_String;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

nIdn: Parameter number for the command, i.e. "S_0_IDN + 47" for executing a S-0-0047 command

nElement: Shows on which parameter part is to be read, e.g. 16#40 is the value of the parameter. Most times only write access on this value is possible, other parameter parts are write-protected.

```
    EC_SOE_ELEMENT_DATASTATE :BYTE :=16#01;
    EC_SOE_ELEMENT_NAME      :BYTE :=16#02;
    EC_SOE_ELEMENT_ATTRIBUTE :BYTE :=16#04;
    EC_SOE_ELEMENT_UNIT      :BYTE :=16#08;
    EC_SOE_ELEMENT_MIN       :BYTE :=16#10;
    EC_SOE_ELEMENT_MAX       :BYTE :=16#20;
    EC_SOE_ELEMENT_VALUE     :BYTE :=16#40;
    EC_SOE_ELEMENT_DEFAULT   :BYTE :=16#80;
```

pSrcBuf: Contains the address of the buffer containing the data to be send.

cbBufLen: The maximum available buffer size for the data to be written, in bytes.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

sPassword: contains the password as Sercos-String. Not yet used. The password has to be written with FB_SoEWritePassword [▶ 10].

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    iAdsErrId     : UINT;
    iSercosErrId  : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

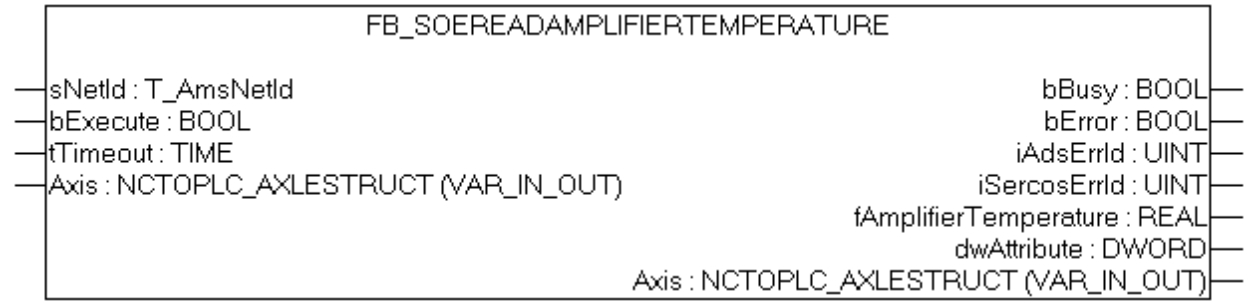
iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbWrite : FB_SoEWrite;
bWrite  : BOOL;
iWriteValue : UINT;
sPassword   : ST_SoE_String;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bWrite THEN
    nIdn := S_0_IDN + 33;
    fbWrite(
        Axis      := stNcToPlc,
        nIdn      := nIdn,
        nElement  := 16#40,
        pSrcBuf   := ADR(iWriteValue),
        cbBufLen  := SIZEOF(iWriteValue),
        sPassword := sPassword,
        bExecute  := TRUE,
        tTimeout  := DEFAULT_ADS_TIMEOUT,
    );
    IF NOT fbWrite.bBusy THEN
        fbWrite(Axis := stNcToPlc, bExecute := FALSE);
        bWrite := FALSE;
    END_IF
END_IF
```

3.5.3 FB_SoEReadAmplifierTemperature



The functionblock FB_SoEReadAmplifierTemperature can be used to read the amplifier temperature (S-0-0384).

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy                : BOOL;
    bError               : BOOL;
    iAdsErrId            : UINT;
    iSercosErrId         : UINT;
    fAmplifierTemperature  : REAL;
    dwAttribute          : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

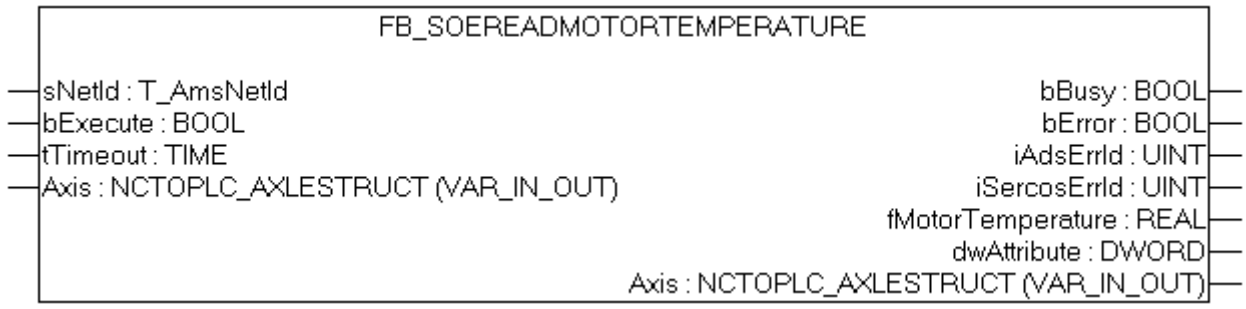
dwAttribute: Supplies the Sercos parameter attribute.

fAmplifierTemperature: Supplies the amplifier temperature (e.g. 26.2 means to 26.2°C).

Sample

```
fbReadAmplifierTemp   : FB_SoEReadAmplifierTemperature;
bReadAmplifierTemp    : BOOL;
fAmplifierTemperature : REAL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadAmplifierTemp THEN
    fbReadAmplifierTemp(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        fAmplifierTemperature=>fAmplifierTemperature
    );
    IF NOT fbReadAmplifierTemp.bBusy THEN
        fbReadAmplifierTemp(Axis := stNcToPlc, bExecute := FALSE);
        bReadAmplifierTemp := FALSE;
    END_IF
END_IF
```

3.5.4 FB_SoEReadMotorTemperature



The functionblock FB_SoEReadMotorTemperature can be used to read the temperature of the motor (S-0-0383). If the motor does not contain a temperature sensor then the FB reports 0.0, means 0.0°C.

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy             : BOOL;
    bError            : BOOL;
    iAdsErrId         : UINT;
    iSercosErrId      : UINT;
    fMotorTemperature : REAL;
    dwAttribute       : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

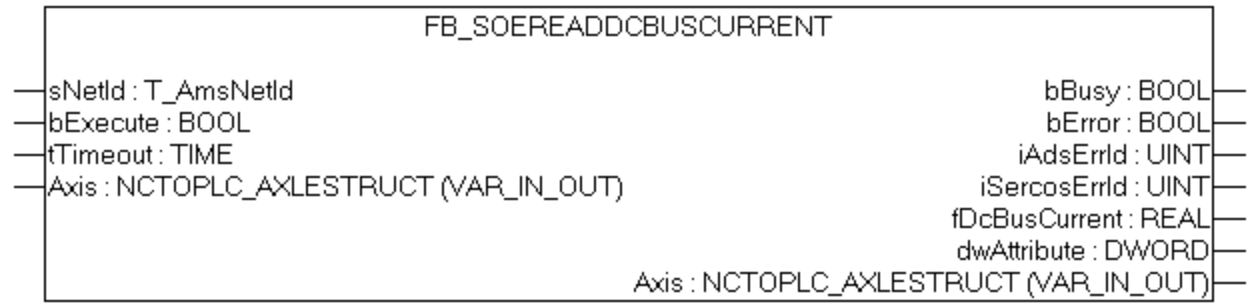
dwAttribute: Supplies the Sercos parameter attribute.

fMotorTemperature: Supplies the motor temperature (i.e. 30.5 means 30.5°C). If the motor does not contain a temperature sensor thenthe value is 0.0, means 0.0°C.

Sample

```
fbReadMotorTemp   :FB_SoEReadMotorTemperature;
bReadMotorTemp    : BOOL;
fMotorTemperature : REAL;


(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadMotorTemp AND NOT bInit THEN
    fbReadMotorTemp(
       Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        fMotorTemperature=>fMotorTemperature
    );
    IF NOT fbReadMotorTemp.bBusy THEN
        fbReadMotorTemp(Axis := stNcToPlc, bExecute := FALSE);
        bReadMotorTemp := FALSE;
    END_IF
END_IF
```

3.5.5 FB_SoEReadDcBusCurrent



The functionblock FB_SoEAX5000ReadDcBusCurrent_ByDriveRef can be used to read the DC-Bus-Current (S-0-0381).

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    iAdsErrId      : UINT;
    iSercosErrId   : UINT;
    fDcBusCurrent  : REAL;
    dwAttribute    : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

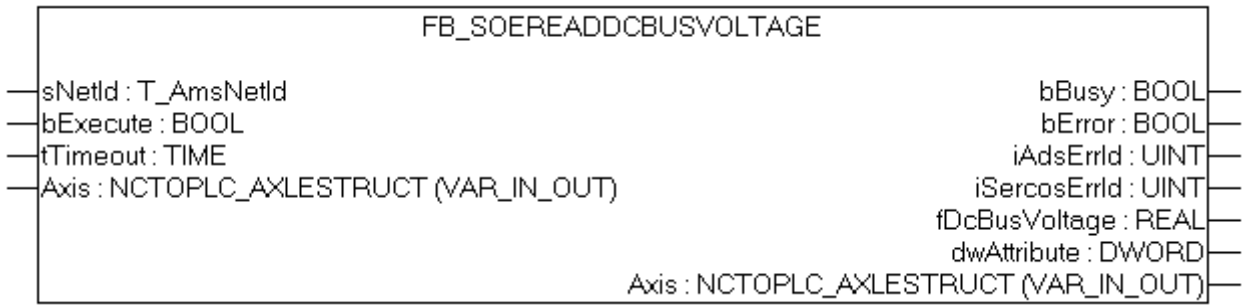
dwAttribute: Supplies the Sercos parameter attribute.

fDcBusCurrent: Supplies the DC-Bus-Current (i.e. 2.040 means 2.040A).

Sample

```
fbReadDcBusCurrent : FB_SoEReadDcBusCurrent_ByDriveRef;
bReadDcBusCurrent  : BOOL;
fDcBusCurrent      : REAL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadDcBusCurrent THEN
    fbReadDcBusCurrent(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        fDcBusCurrent=>fDcBusCurrent
    );
    IF NOT fbReadDcBusCurrent.bBusy THEN
        fbReadDcBusCurrent(Axis := stNcToPlc, bExecute := FALSE);
        bReadDcBusCurrent := FALSE;
    END_IF
END_IF
```

3.5.6 FB_SoEReadDcBusVoltage



The functionblock FB_SoEReadDcBusVoltage can be used to read the DC-Bus-Voltage of the amplifier (S-0-0380).

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    iAdsErrId      : UINT;
    iSercosErrId   : UINT;
    fDcBusVoltage  : REAL;
    dwAttribute    : DWORD;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

dwAttribute: Supplies the Sercos parameter attribute.

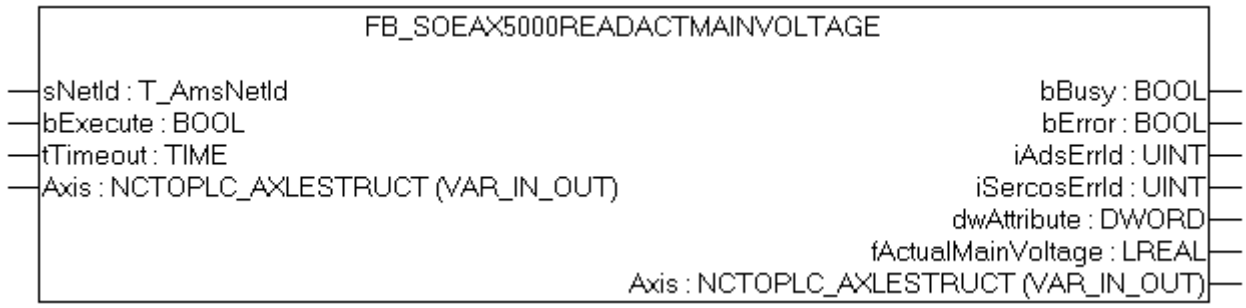
fDcBusVoltage: Supplies the DC-Bus-Voltage (i.e. 294.0 means 294.0V).

Sample

```
fbReadDcBusVoltage : FB_SoEReadDcBusVoltage;
bReadDcBusVoltage  : BOOL;
fDcBusVoltage      : REAL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadDcBusVoltage THEN
    fbReadDcBusVoltage(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        fDcBusVoltage=>fDcBusVoltage
    );
    IF NOT fbReadDcBusVoltage.bBusy THEN
        fbReadDcBusVoltage(Axis := stNcToPlc, bExecute := FALSE);
        bReadDcBusVoltage := FALSE;
    END_IF
END_IF
```

4 AX5000 specific FB

4.1 FB_SoEAX5000ReadActMainVoltage



The functionblock FB_SoEAX5000ReadActMainVoltage can be used to read the main voltage (P-0-0200) of the AX5000.

VAR_INPUT

```
VAR_INPUT
    sNetId  : T_AmsNetId := '';
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy              : BOOL;
    bError             : BOOL;
    iAdsErrId          : UINT;
    iSercosErrId       : UINT;
    dwAttribute        : DWORD;
    fActualMainVoltage : LREAL;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

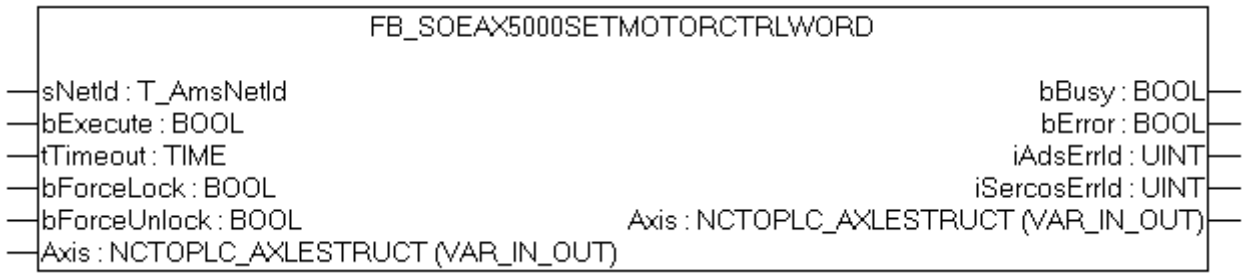
dwAttribute: Supplies the Sercos parameter attribute.

fActualMainVoltage: Supplies the main voltage of the AX5000 (i.e. 303.0 means 303.0V).

Sample

```
fbReadActMainVoltage : FB_SoEAX5000ReadActMainVoltage;
bReadActMainVoltage  : BOOL;
fActualMainVoltage   : REAL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bReadActMainVoltage THEN
    fbReadActMainVoltage(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        fActualMainVoltage=>fActualMainVoltage
    );
    IF NOT fbReadActMainVoltage.bBusy THEN
       fbReadActMainVoltage(Axis := stNcToPlc, bExecute := FALSE);
       bReadActMainVoltage := FALSE;
    END_IF
END_IF
```

4.2 FB_SoEAX5000SetMotorCtrlWord



The functionblock FB_SoEAX5000SetMotorCtrlWord can be used to set the ForceLock-Bit (Bit 0) or the ForceUnlock-Bit in the motor control word (P-0-0096), in order to set or release the brake. The brake is set and released automatically via the enable of the drive.

TheForceLock-Bit can be used to set the brake independent of the enable, the ForceUnlock-Bit can be used to release the brake independent of the enable. If ForceLock and ForceUnlock are set simultaneously then the ForceLock (brake locked) has the higher priority.

VAR_INPUT

```
VAR_INPUT
    sNetId       : T_AmsNetId := '';
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
    bForceLock   : BOOL;
    bForceUnlock : BOOL;
END_VAR
```

sNetId: A string containing the AMS network identifier of the PC.

bExecute: The block is activated by a rising edge at this input.

tTimeout: Maximum time allowed for the execution of the function block.

bForceLock: Lock the brake independent of the enable.

bForceUnlock: Release (unlock) the brake independent of the enable.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy        : BOOL;
    bError       : BOOL;
    iAdsErrId    : UINT;
    iSercosErrId : UINT;
END_VAR
```

bBusy: This output is set when the function block is activated, and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

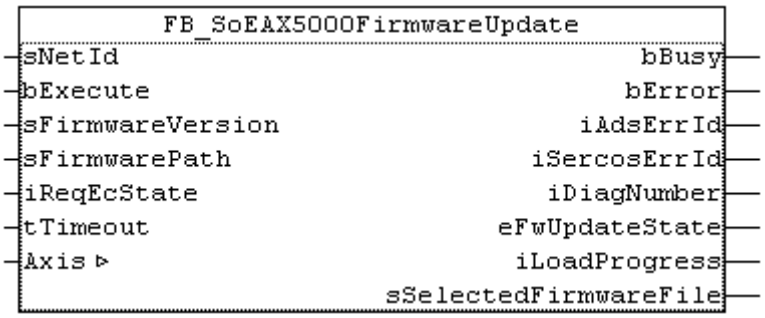
iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

Sample

```
fbSetMotorCtrlWord : FB_SoEAX5000SetMotorCtrlWord;
bSetMotorCtrlWord  : BOOL;
bForceLock         : BOOL;
bForceUnlock       : BOOL;

(* NcAxis *)
stNcToPlc AT %I* : NCTOPLC_AXLESTRUCT;
IF bSetMotorCtrlWord THEN
    fbSetMotorCtrlWord(
        Axis     := stNcToPlc,
        bExecute := TRUE,
        tTimeout := DEFAULT_ADS_TIMEOUT,
        bForceLock := bForceLock,
        bForceUnlock:= bForceUnlock
    );

    IF NOT fbSetMotorCtrlWord.bBusy THEN
        fbSetMotorCtrlWord(Axis := stNcToPlc, bExecute := FALSE);
        bSetMotorCtrlWord := FALSE;
    END_IF
END_IF
```

4.3 FB_SoEAX5000FirmwareUpdate



The functionblock FB_SoEAX5000FirmwareUpdate can be used to check and update the firmware of the AX5000 to a requested version (revision and build) or to the newest build of the configured revision.

In order to update the following sequence is executed:
- get configured slave type, i.e. AX5103-0000-0010
- get scanned slave type for the slave address, i.e. AX5103-0000-0009
- get current slave firmware, i.e. v1.05_b0009
- compare configured and scanned slave (number of channels, current, revision, firmware)
- create firmware files name and search for the file

- update firmware (if required)
- rescan slave
- switch the slave to the requested EtherCAT state

A successful updatefinishes with**eFwUpdateState = eFwU_FwUpdateDone**. If the update is not required, then the state returns **eFwUpdateState = eFwU_NoFwUpdateRequired**. The firmware update is executed via the channel of the drive (A=0 or B=1) set in stDriveRef. With two channel devices, the firmware update can only be executed via one of the channels. The other channel signals**eFwUpdateState = eFwU_UpdateViaOtherChannelActive** or **= eFwU_UpdateViaOtherChannel**.

During the firmware update (**eFwUpdateState = eFwU_FwUpdateInProgress**) the update progress is reported via**iLoadProgress in percent**.

Attention: During the update the PLC and TwinCAT have to stay in RUN mode, the EtherCAT connection must be maintained and the AX5000 must stay powered up!

VAR_INPUT

```
VAR_INPUT
   sNetId           : T_AmsNetId;
   bExecute         : BOOL;
   sFirmwareVersion : STRING(20);  (* version string vx_yy_bnnnn, e.g. "v1.05_b0009" for v1.05 Build
0009 *)
   sFirmwarePath    : T_MaxString; (* drive:\path, e.g. "C:\TwinCAT\Io\TcDriveManager\FirmwarePool"
*)
   iReqEcState      : UINT := EC_DEVICE_STATE_OP;
   tTimeout         : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

bExecute: The block is activated by a rising edge at this input.

sFirmwareVersion: The required firmware version in form of **v**x.yy_**b**nnnn, i.e. "**v**1.05_**b**0009" for version v1.05 build 0009.

Release builds:
"v1.05_b**00**09" for specific build, i.e. v1.05 Build 0009
"v1.05_b**00**??" newstest build of a version, i.e. v1.05
"v1.??_b**00**??" newstest build of a major version, i.e. v1
"v?.??_b**00**??" newstest version and newstest build
"" newstest version and newstest build

Customer specific firmware builds:
"v1.05_b**10**09" for specific build, i.e. v1.05 Build 1009
"v1.05_b**10**??" newstest build of a version, i.e. v1.05
"v1.??_b**10**??" newstest build of a major version, i.e. v1
"v?.??_b**10**??" newstest version and newstest build
...

"v1.05_b**89**09" for specific build, i.e. v1.05 Build 8909
"v1.05_b**89**??" newstest build of a version, i.e. v1.05
"v1.??_b**89**??" newstest build of a major version, i.e. v1
"v?.??_b**89**??" newstest version and newstest build

Debug builds:
"v1.05_b**90**09" for specific build, i.e. v1.05 Build 9009
"v1.05_b**90**??" newstest build of a version, i.e. v1.05
"v1.??_b**90**??" newstest build of a major version, i.e. v1
"v?.??_b**90**??" newstest version and newstest build

sFirmwarePath: The path for the firmware pool, where the firmware files are located, i.e. "C:\TwinCAT\Io\TcDriveManager\FirmwarePool".

sNetIdIPC: AMS-NetID of the controller (IPC).

iReqEcState: Requested EtherCAT state after the update (only if an update is executed). The states are defined in the TcEtherCAT.lib as globale constants.

tTimeout: The firmware update can take a few minutes, the timeout here defines the timeout for internal ADS instances.

VAR_IN_OUT

```
VAR_IN_OUT
    Axis : NCTOPLC_AXLESTRUCT;(* reference to NC axis *)
END_VAR
```

Axis: Axis structure.

VAR_OUTPUT

```
VAR_OUTPUT
   bBusy                : BOOL;
   bError               : BOOL;
   iAdsErrId            : UINT;
   iSercosErrId         : UINT;
   iDiagNumber          : UDINT;
   eFwUpdateState       : E_FwUpdateState;
   iLoadProgress        : INT;
   sSelectedFirmwareFile : STRING(MAX_STRING_LENGTH); (* found firmware file, e.g.
"AX5yxx_xxxx_0010_v1_05_b0009.efw" *)
END_VAR
```

bBusy: This output is set when the function block is activated and remains set until an acknowledgement is received.

bError: This output is set up after the bBusy output has been reset if there has been an error in transmission of the command.

iAdsErrId: Supplies the ADS error code associated with the most recently executed command if the bError output is set.

iSercosErrId: Supplies the Sercos error code associated with the most recently executed command if the bError output is set.

iDiagNumber: Supplies the drive error code associated with the most recently executed firmware update if the bError output is set.

eFwUpdateState: Supplies the status of the firmware update. See E_FwUpdateState [▶ 37].

iLoadProgress: Supplies the progress of the firmware load in percent.

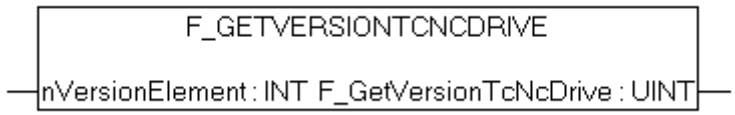
sSelectedFirmwareFile: Supplies the name of the searched firmware file.

Sample

```
VAR CONSTANT
   iNumOfDrives         : INT := 2;
END_VAR

VAR
    fbFirmwareUpdate     : ARRAY[1..iNumOfDrives] OF FB_SoEAX5000FirmwareUpdate;
    stNcToPlc AT %I*     : ARRAY[1..iNumOfDrives] OF NCTOPLC_AXLESTRUCT;
    sFirmwareVersion     : ARRAY[1..iNumOfDrives] OF STRING(20)(* :=2('v1.04_b0002') *);

    eFwUpdateState       : ARRAY[1..iNumOfDrives] OF E_FwUpdateState;
    sSelectedFirmwareFile: ARRAY [1..iNumOfDrives]OF STRING(MAX_STRING_LENGTH);
    iUpdateState         : INT;
    bExecute             : BOOL;
    sNetIdIPC            : T_AmsNetId := '';
    sFirmwarePath        : T_MaxString := 'C:\TwinCAT\Io\TcDriveManager\FirmwarePool';
    I                    : INT;
    bAnyBusy             : BOOL;
    bAnyError            : BOOL;
END_VAR
CASE iUpdateState OF
0:
    IF bExecute THEN
        iUpdateState := 1;
    END_IF

1:
```

```
    FOR I := 1 TO iNumOfDrives DO
       fbFirmwareUpdate[I](
           Axis := stNcToPlc[I],
           bExecute := TRUE,
           tTimeout := T#15s,
           sFirmwareVersion := sFirmwareVersion[I],
           sFirmwarePath := sFirmwarePath,
           sNetId := sNetIdIPC,
           iReqEcState := EC_DEVICE_STATE_OP,
           eFwUpdateState => eFwUpdateState[I],
        );
    END_FOR
    iUpdateState := 2;

2:
    bAnyBusy := FALSE;
    bAnyError:= FALSE;
    FOR I := 1 TO iNumOfDrives DO
       fbFirmwareUpdate[I](
           Axis := stNcToPlc[I],
           eFwUpdateState => eFwUpdateState[I],
           sSelectedFirmwareFile => sSelectedFirmwareFile[I],
        );
        IF NOT fbFirmwareUpdate[I].bBusy THEN
           fbFirmwareUpdate[I](bExecute := FALSE, Axis := stNcToPlc[I]);
           IF fbFirmwareUpdate[I].bError THEN
               bAnyError := TRUE;
           END_IF
        ELSE
           bAnyBusy := TRUE;
        END_IF
    END_FOR

    IF NOT bAnyBusy THEN
        bExecute := FALSE;

        IF NOT bAnyError THEN
           iUpdateState := 0; (* OK *)
        ELSE
           iUpdateState := 3; (* Error *)
        END_IF
    END_IF
3:
    (* Error handling *)
    iUpdateState := 0;

END_CASE
```

5 F_GetVersionTcNcDrive



This function can be used to read PLC library version information.

FUNCTION F_GetVersionTcNcDrive : UINT

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

nVersionElement : Version element to be read. Possible parameters:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

Requirements

Development environment	Target system type	PLC libraries to be linked
TwinCAT v2.10.0 Build >= 1329	PC or CX (x86)	TcDrive.lib, TcEtherCAT.lib, TcUtilities.Lib, TcSystem.lib
TwinCAT v2.10.0 Build >= 1329	CX (ARM)	

6 E_FwUpdateState

E_FwUpdateState describes the state of the firmware update.

```
TYPE E_SoE_CmdState :(
   (* update states *)
   eFwU_NoError := 0,
   eFwU_CheckCfgIdentity,
   eFwU_CheckSlaveCount,
   eFwU_CheckFindSlavePos,
   eFwU_WaitForScan,
   eFwU_ScanningSlaves,
   eFwU_CheckScannedIdentity,
   eFwU_CheckScannedFirmware,
   eFwU_FindFirmwareFile,
   eFwU_WaitForUpdate,
   eFwU_WaitForSlaveState,
   eFwU_StartFwUpdate,
   eFwU_FwUpdateInProgress,
   eFwU_FwUpdateDone,
   eFwU_NoFwUpdateRequired,

   (* not updating via this channel *)
   eFwU_UpdateViaOtherChannelActive,
   eFwU_UpdatedViaOtherChannel,

   (* error states *)
   eFwU_GetSlaveIdentityError     := -1,
   eFwU_GetSlaveCountError        := -2,
   eFwU_GetSlaveAddrError         := -3,
   eFwU_StartScanError        := -4,
   eFwU_ScanStateError        := -5,
   eFwU_ScanIdentityError         := -6,
   eFwU_GetSlaveStateError        := -7,
   eFwU_ScanFirmwareError         := -8,
   eFwU_FindFileError         := -9,
   eFwU_CfgTypeInNoAX5xxx         := -10,
   eFwU_ScannedTypeInNoAX5xxx     := -11,
   eFwU_ChannelMismatch           := -12,
   eFwU_ChannelMismatch_1Cfg_2Scanned := -13,
   eFwU_ChannelMismatch_2Cfg_1Scanned := -14,
   eFwU_CurrentMismatch           := -15,
   eFwU_FwUpdateError         := -16,
   eFwU_ReqSlaveStateError        := -17
);
END_TYPE
```

```
Update Status

eFwU_NoError
: initial state


eFwU_CheckCfgIdentity
: reading of the configured slave type (number of channels,
current, revision)


eFwU_CheckSlaveCount
: get configured amount of slaves


eFwU_CheckFindSlavePos
: search slave address in the master object directory


eFwU_WaitForScan
: wait for online scan


eFwU_ScanningSlaves
: online scan of slaves


eFwU_CheckScannedIdentity
: reading of scanned slave types (number of channels, current,
revision)
```

```
eFwU_CheckScannedFirmware
: get firmware version of the drive


eFwU_FindFirmwareFile
: search for firmware file


eFwU_WaitForUpdate
: wait for updates (short delay before the update)


eFwU_WaitForSlaveState
: get EtherCAT slave state


eFwU_StartFwUpdate
: Start firmware update


eFwU_FwUpdateInProgress
: firmware update active


eFwU_FwUpdateDone
: firmware update succeeded


eFwU_NoFwUpdateRequired
: no firmware update required


   eFwU_UpdateViaOtherChannelActive   : Update
via the other drive channel active


eFwU_UpdatedViaOtherChannel
: Updated via the other drive channel

Update Errors

eFwU_GetSlaveIdentityError
: reading of the configured slave type failed, see iAdsErrId


eFwU_GetSlaveCountError
: get configured amount of slaves failed, see iAdsErrId


eFwU_GetSlaveAddrError
: search slave address in the master object directory failed, see
iAdsErrId


eFwU_StartScanError
: start of online scan of slaves failed, see iAdsErrId


eFwU_ScanStateError
: online scan failed, see iAdsErrId


eFwU_ScanIdentityError
: reading of scanned slave types (number of channels, current,
revision) failed, see iAdsErrId


eFwU_GetSlaveStateError
: get EtherCAT slave state failed, see iAdsErrId


eFwU_ScanFirmwareError
: get firmware version of the drive failed, see iAdsErrId +
iSercosErrId


eFwU_FindFileError
: search for firmware file failed, see iAdsErrId
```

```
eFwU_CfgTypeInNoAX5xxx
: the configured slave is not an AX5000


eFwU_ScannedTypeInNoAX5xxx
: the scanned slave is not an AX5000


eFwU_ChannelMismatch
: The amoutn of configured and scanned channels of the AX5000 do
not match

   eFwU_ChannelMismatch_1Cfg_2Scanned : one channel
device configured but two channel device found

   eFwU_ChannelMismatch_2Cfg_1Scanned : two channel
device configured but one channel device found


eFwU_CurrentMismatch
: current of the AX5000 type does not match, i.e. AX5103 (3A)
configured but AX5106 (6A) found


eFwU_FwUpdateError
: general update error, see iAdsErrId


eFwU_ReqSlaveStateError
: switching ot requested EtherCAT state failed, see iAdsErrId
```

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