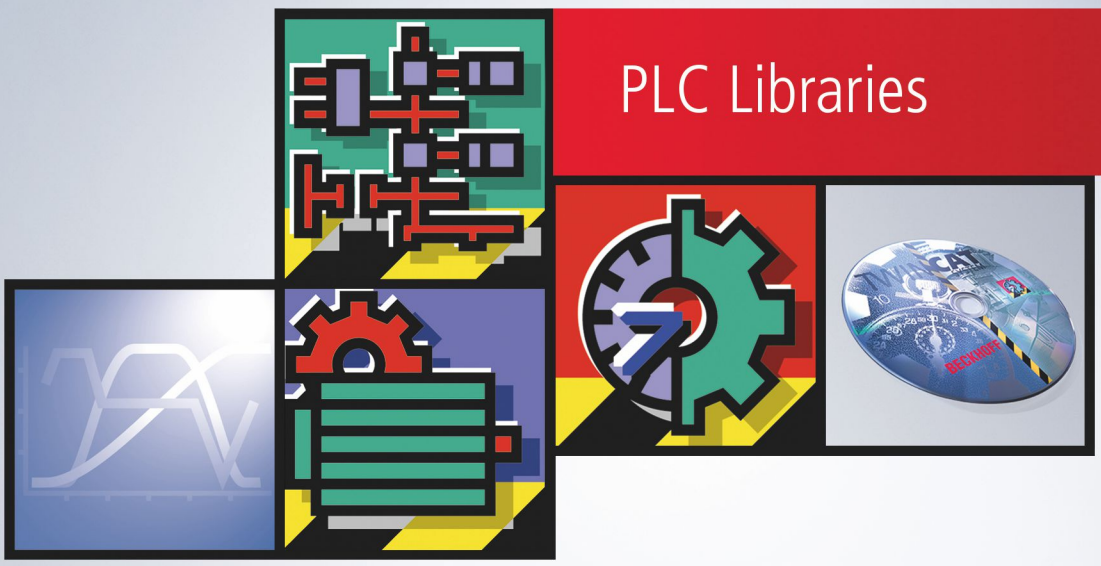
BECKHOFF New Automation Technology

Handbuch | DE

PLC-Bibliothek: TcNCDrive

TwinCAT 2 | TX1200, PlcNc, TcNcUtilities



07.09.2023 | Version: 1.1

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

⚠ GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

⚠ WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

⚠ VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt

Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

2 POUs der TcNcDrive.lib

In dieser Bibliothek sind Funktionen und Funktionsbausteine für SoE-Antriebe enthalten, die per NC-Achsstruktur auf den Antrieb zugreifen.

Die TcNcDrive.lib enthält Wrapper-Bausteine um die Bausteine der TcDrive.lib.

Unterschiede bei der Verwendung der Drive-Libs in Verbindung mit AX5000 und mit Bosch-Rexroth IndraDrive CS sind zu berücksichtigen. Beispiel siehe unten.

Die TcNcDrive.lib sollte dann verwendet werden, wenn der Antrieb über die NC mit den Bibliotheken TcNc.lib bzw. TcMc.lib verwendet wird. Hierzu wird auf den Antrieb über die NC-Achsstruktur (NC_TO_PLC) zugegriffen, die auch in den Bausteinen der TcNc.lib bzw. TcMc.lib verwendet werden. Die Bausteine in der TcNcDrive.lib ermitteln eigenständig über die NC-AchsID aus der NC_TO_PLC-Struktur die Zugriffsdaten auf den Antrieb (NetID, Adresse und Kanalnummer). Siehe Beispiel bei den jeweiligen Funktionsbausteinen in der Dokumentation der TcNcDrive.lib.

Um auf Parameter im Antrieb zuzugreifen, für die kein spezieller Baustein implementiert wurde, können die Bausteine FB_SoERead und FB_SoEWrite verwendet werden.

Funktionen

Name	Beschreibung
F_GetVersionTcNcDrive [▶ 37]	Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.
F_ConvWordToSTAX5000C1D	Siehe Dokumentation TcDrive.lib. Konvertiert das C1D-Wort (S-0-0011) des AX5000 in eine Struktur ST_AX5000_C1D.

Funktionsbausteine

Name	Beschreibung
FB_SoEReset [▶ 10]	Antriebsreset ausführen (S-0-0099)
FB_SoEWritePassword [▶ 11]	Setzen des Antriebspassworts (S-0-0267)
FB_SoEReadDiagMessage [▶ 17]	Lesen der Diagnosenachricht (S-0-0095)
FB_SoEReadDiagNumber [▶ 18]	Lesen der Diagnosenummer (S-0-0390)
FB_SoEReadDiagNumberList [▶ 19]	Lesen der Diagnosenummernliste (bis zu 30 Einträge) (S-0-0375)
FB_SoEExecuteCommand [▶ 12]	Ausführen eines Kommandos
FB_SoEWriteCommandControl [▶ 14]	Setzen des Command Control
FB_SoEReadCommandState [▶ 15]	Prüfen des Kommandostatus
FB_SoEReadClassXDiag [▶ 20]	Lesen der Class 1 Diagnose (S-0-0011) ... Class 3 Diagnose (S-0-0013)
FB_SoERead [▶ 22]	Lesen eines Parameters
FB_SoEWrite [▶ 23]	Schreiben eines Parameters
FB_SoEReadAmplifierTemperature [▶ 25]	Lesen der Antriebstemperatur (S-0-0384)

Name	Beschreibung
FB_SoEReadMotorTemperature [▶ 26]	Lesen der Motortemperatur (S-0-0383)
FB_SoEReadDcBusCurrent [▶ 27]	Lesen des Dc-Bus-Stroms (S-0-0381)
FB_SoEReadDcBusVoltage [▶ 29]	Lesen der Dc-Bus-Spannung (S-0-0380)
FB_SoEAX5000ReadActMainVoltage [▶ 31]	Lesen der Netzspannung (P-0-0200)
FB_SoEAX5000SetMotorCtrlWord [▶ 32]	Setzen des Motor Control Words (P-0-0096)
FB_SoEAX5000FirmwareUpdate [▶ 33]	Automatischer Firmware-Update des AX5000

Beispielprojekt und Beispielkonfiguration für AX5000-Diagnose

Siehe https://infosys.beckhoff.com/content/1031/tcplclibncdrive/Resources/14929629323.zip

Beispielprojekt und Beispielkonfiguration für IndraDriveCS-Diagnose

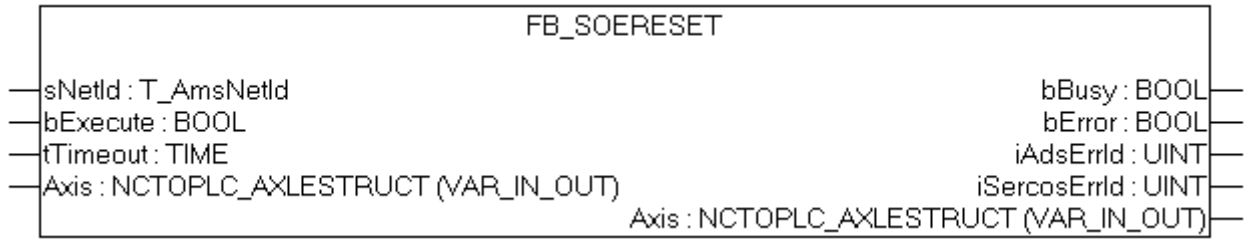
Siehe https://infosys.beckhoff.com/content/1031/tcplclibncdrive/Resources/14929630987.zip

Voraussetzungen

Komponente	Version
TwinCAT auf dem Entwicklungsrechner	2.10 Build 1335 oder höher
TwinCAT auf dem Windows CE-Image	2.10 Build 1333 oder höher
TwinCAT auf dem Windows XP-Image	2.10 Build 1333 oder höher

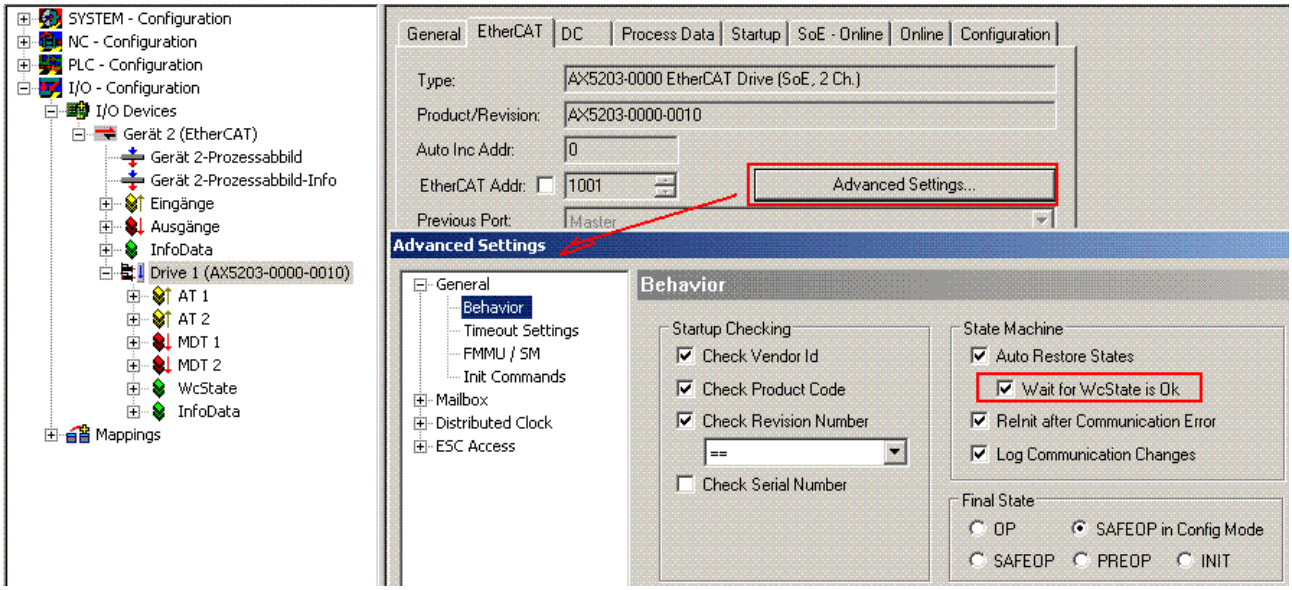
3 Allgemeine SoE FBs

3.1 FB_SoEReset



Mit dem Funktionsbaustein FB_SoEReset kann ein Antriebsreset (S-0-0099) ausgeführt werden. Bei mehrkanaligen Geräten müssen ggf. beide Kanäle einen Reset ausführen. Die Timeoutzeit muss 10s betragen, da der Reset je nach Fehler bis zu 10s dauern kann.

Für den AX5000 muss das Flag "Wait For WcState is OK" in den Advanced EtherCAT Settings aktiviert sein.



Ein NC-Reset wird nicht ausgeführt. Falls ein NC-Reset nötig ist, kann er über den MC_Reset-Baustein aus der TcMc.lib ausgeführt werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

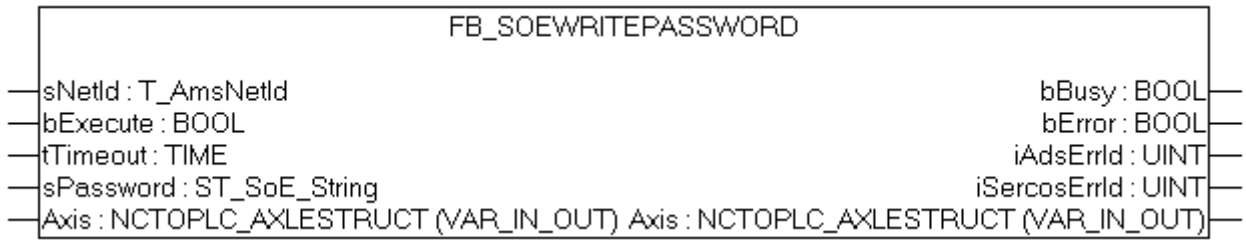
iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEWritePassword kann das Antriebspasswort (S-0-0267) gesetzt werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

sPassword: enthält das Passwort als Sercos-String

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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 Kommando FBs

3.3.1 FB_SoEExecuteCommand

Mit dem Funktionsbaustein FB_SoEExecuteCommand kann ein Kommando ausgeführt werden.

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId := '';
    nIdn     : WORD;
```

```
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

nIdn: Parameternummer, auf das sich das FB_SoEExecuteCommand_ByDriveRef bezieht, "P_0_IDN + 160" für P-0-0160

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

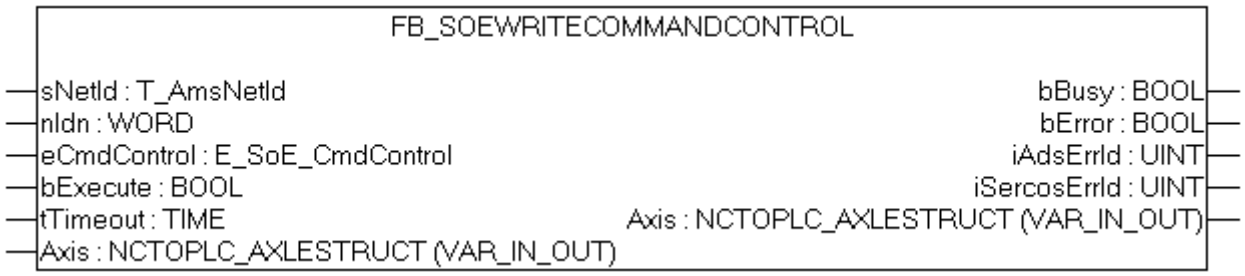
iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEWriteCommandControl kann ein Kommando vorbeireitet, gestartet oder abgebrochen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

nIdn: Parameternummer, auf das sich das FB_SoEWriteCommandControl_ByDriveRef bezieht, z.B. "P_0_IDN + 160" für P-0-0160

eCmdControl: Gibt an, ob das vorbereitet (eSoE_CmdControl_Set := 1), ausgeführt (eSoE_CmdControl_SetAndEnable := 3) oder abgebrochen (eSoE_CmdControl_Cancel := 0) werden soll

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

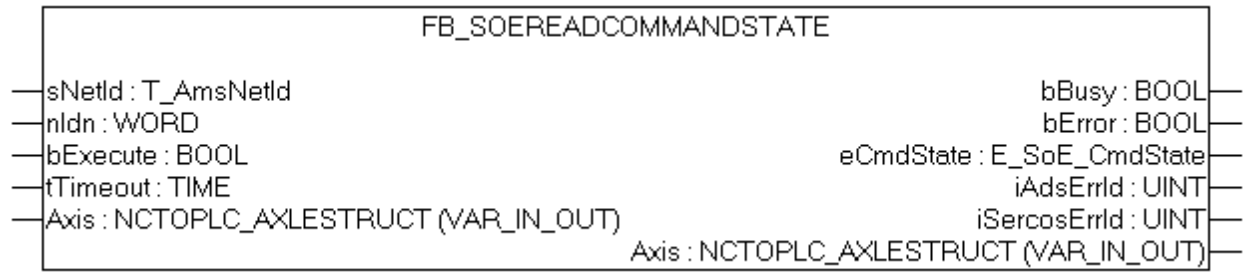
iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadCommandState kann die Kommandoausführung überprüft werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

Idn: Parameternummer, auf das sich das FB_SoEReadCommandState_ByDriveRef bezieht, z.B. "P_0_IDN + 160" für P-0-0160

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

dwAttribute: Liefert das Attribut des Sercos-Parameters.

eCmdState: Liefert den Kommandostatus

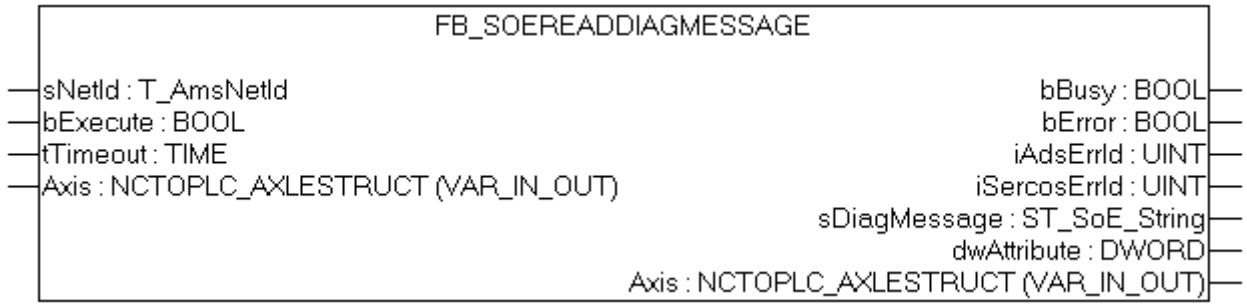
```
    eSoE_CmdState_NotSet =
0
- kein Kommando aktiv

    eSoE_CmdState_Set =
1
- Kommando gesetzt (vorbereitet) aber (noch) nicht ausgeführt

    eSoE_CmdState_Executed =
2
- Kommando wurde ausgeführt

    eSoE_CmdState_SetEnabledExecuted =
3    - Kommando gesetzt (vorbereitet) und
ausgeführt

    eSoE_CmdState_SetAndInterrupted =
5     - Kommando wurde gesetzt aber
unterbrochen

    eSoE_CmdState_SetEnabledNotExecuted = 7 -
Kommandoausführung ist noch aktiv

    eSoE_CmdState_Error =
15
- Fehler bei der Kommandoausführung, es wurde in den Fehlerstate
gewechselt
```

Beispiel

```
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 Diagnose FBs

3.4.1 FB_SoEReadDiagMessage



Mit dem Funktionsbaustein FB_SoEReadDiagMessage kann die Diagnosenachricht als Sercos-String (S-0-0095) ausgelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

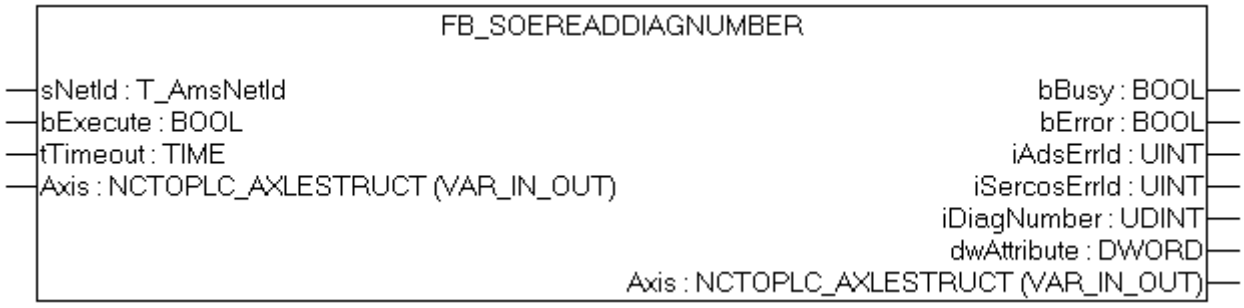
dwAttribute: Liefert das Attribut des Sercos-Parameters.

sDiagMessage: Liefert die Diagnosenachricht.

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadDiagNumber kann die aktuelle Diagnosenummer als UDINT (S-0-0390) ausgelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

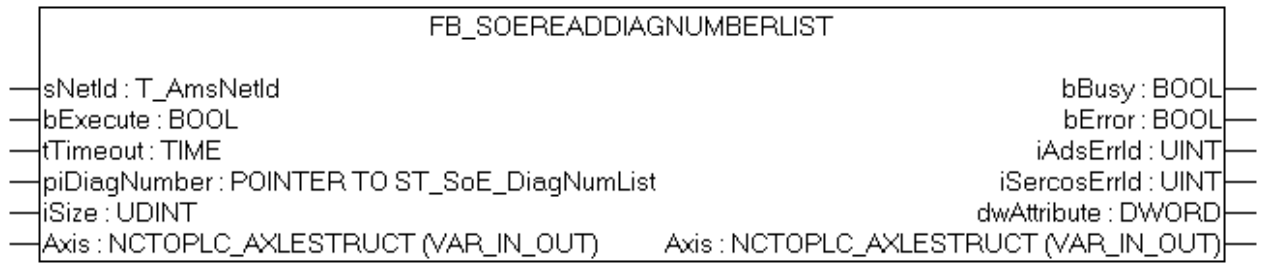
dwAttribute: Liefert das Attribut des Sercos-Parameters.

iDiagNumber: Liefert die aktuelle Diagnosenummer.

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadDiagNumberList kann eine Historie der Diagnosenummern als Liste (S-0-0375) ausgelesen werden.

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: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

piDiagNumber: Zeiger auf Liste der letzten max. 30 Fehlernummern. Die Liste besteht aus aktueller und maximaler Anzahl von Bytes in der Liste, sowie den 30 Listeneinträgen

iSize: Größe der Liste in Bytes (als Sizeof())

VAR_IN_OUT

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

Axis: Achsstruktur

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

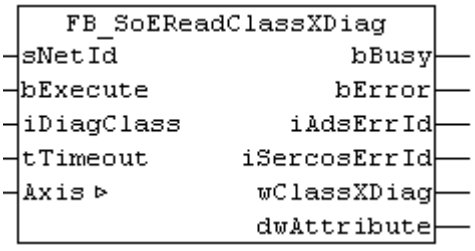
iSercosErrId: Liefert bei einem gesetzten bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

dwAttribute: Liefert das Attribut des Sercos-Parameters.

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadClassXDiag kann die aktuelle Class 1 Diagnose(S-0-0011) ... Class 3 Diagnose (S-0-0013) als WORD ausgelesen werden. Für die Auswertung der Class 1 Diagnose als Struktur ST_AX5000_C1D gibt es eine Konvertierungsfunktion F_ConvWordToSTAX5000C1D. Siehe Dokumentation 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: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

iDiagClass: Gibt an, welche Diagnose gelesen werden soll. Die Diagnose Parameter können sich von Hersteller zu Hersteller unterscheiden. Nicht immer sind alle Diagnose Parameter (C1D ... C3D) oder alle Bits darin implementiert.
1: Fehler: Class 1 Diag (S-0-0011)
2: Warnungen: Class 2 Diag (S-0-0012)
3: Informationen: Class 3 Diag (S-0-0013)

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

wClassXDiag: Liefert die aktuelle Class X Diagnose.

dwAttribute: Liefert das Attribut des Sercos-Parameters.

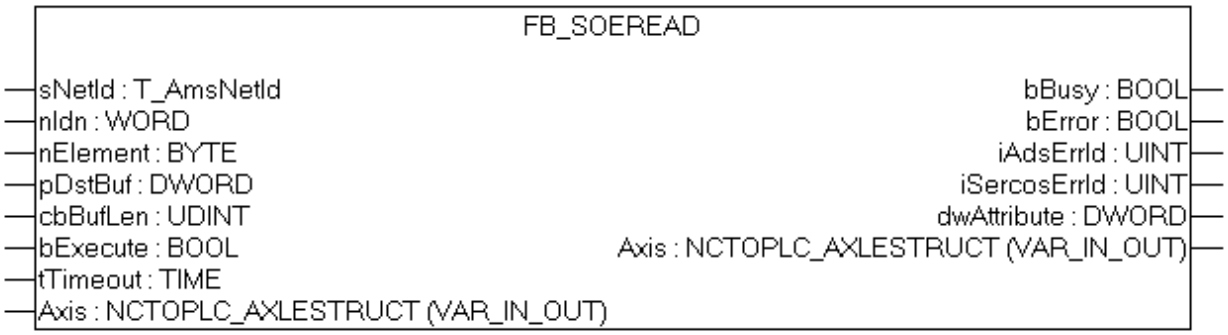
Beispiel

```
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 FBs für aktuelle Werte

3.5.1 FB_SoERead



Mit dem Funktionsbaustein FB_SoERead kann ein Parameter eingelesen werden.

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: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

nIdn: Parameternummer, auf das sich das FB_SoERead bezieht, "S_0_IDN + 33" für S-0-0033

nElement: Gibt an, auf welchen Teil des Parameters zugegriffen werden soll, z.B. 16#40 ist der Wert (Value) des Parameters

```
    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;
```

pDstBuf: ADR() der Variablen, in die der Wert gelesen werden soll.

cbBufLen: SIZEOF() der Variablen, in die der Wert gelesen werden soll.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

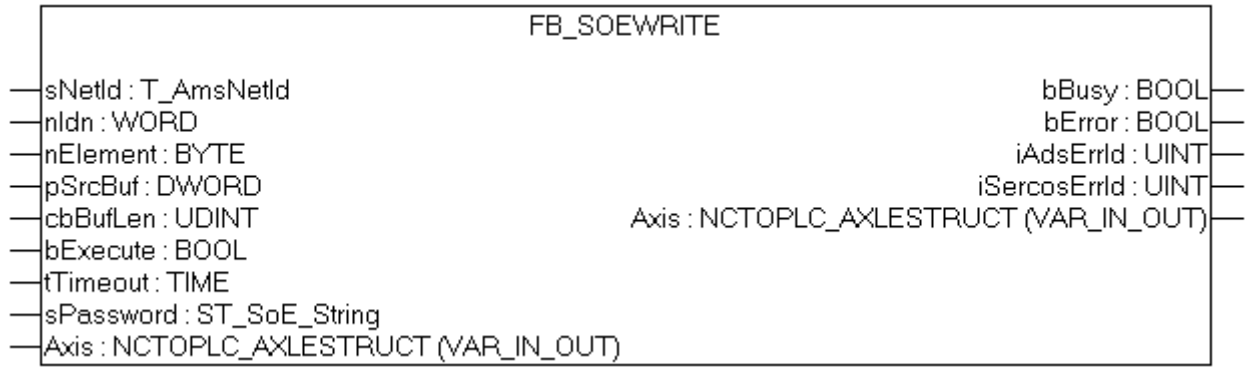
iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

dwAttribute: Liefert das Attribut des Sercos-Parameters.

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEWrite kann ein Parameter geschrieben werden.

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: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

nIdn: Parameternummer, auf das sich das FB_SoERead bezieht, "S_0_IDN + 47" für S-0-0047

nElement: Gibt an, auf welchen Teil des Parameters zugegriffen werden soll, z.B. 16#40 ist der Wert (Value) des Parameters. Meist kann nur auf den Wert schreibend zugegriffen werden, andere Bestandteile des Parameters sind schreibgeschützt.

```
    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: ADR() der Variablen, die den zu schreibenden Wert enthält.

cbBufLen: SIZEOF() der Variablen, die den zu schreibenden Wert enthält

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

sPassword: enthält das Passwort als Sercos-String. Wird z.Z. noch nicht verwendet. Das Passwort muss mit FB_SoEWritePassword [▶ 11] geschrieben werden.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

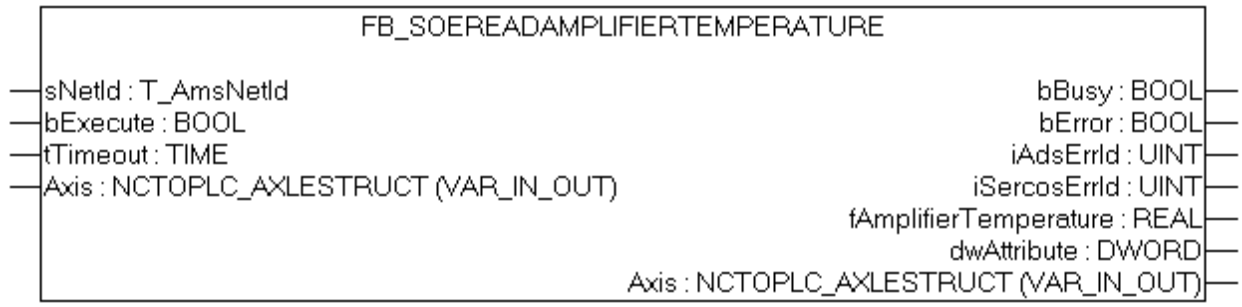
iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadAmplifierTemperature kann die Temperatur des Antriebs (S-0-0384) eingelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

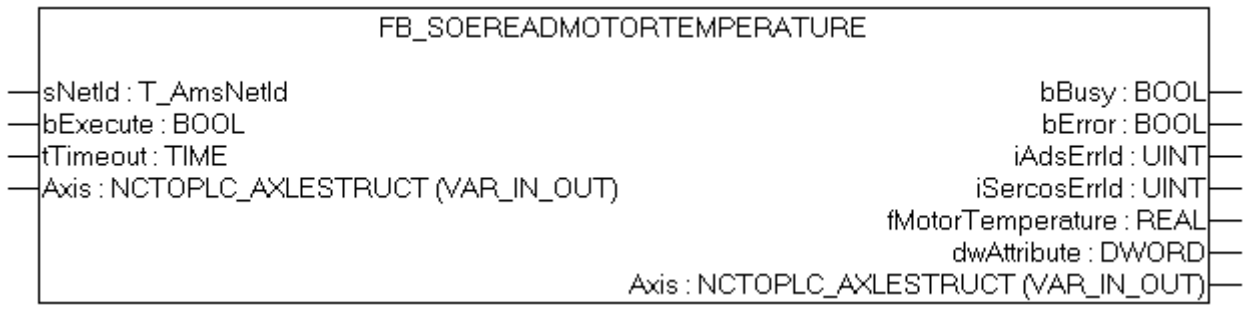
dwAttribute: Liefert das Attribut des Sercos-Parameters.

fAmplifierTemperature: Liefert die Antriebstemperatur (z.B. 26.2 entspricht 26.2°C).

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadMotorTemperature kann die Temperatur des Motor (S-0-0383) eingelesen werden. Falls der Motor keinen Temperatursensor enthält, steht hier 0.0, heißt 0.0°C.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

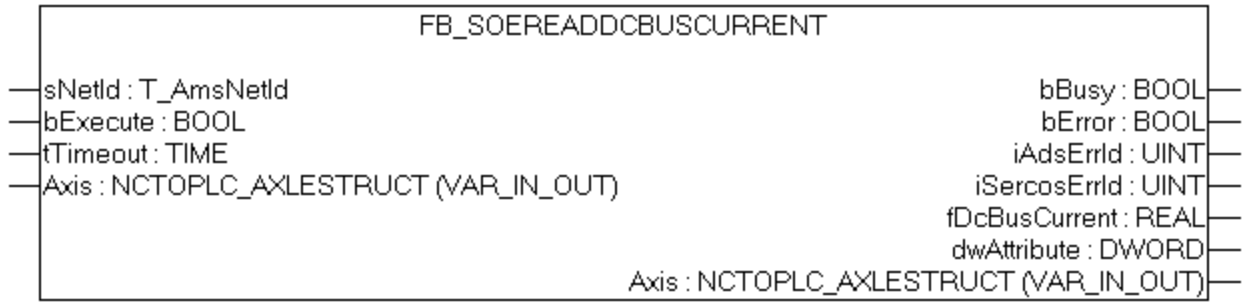
dwAttribute: Liefert das Attribut des Sercos-Parameters.

fMotorTemperature: Liefert die Motortemperatur (z.B. 30.5 entspricht 30.5°C). Falls der Motor keinen Temperatursensor enthält, steht hier 0.0, heißt 0.0°C.

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEAX5000ReadDcBusCurrent kann der DC-Bus-Strom (S-0-0381) eingelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

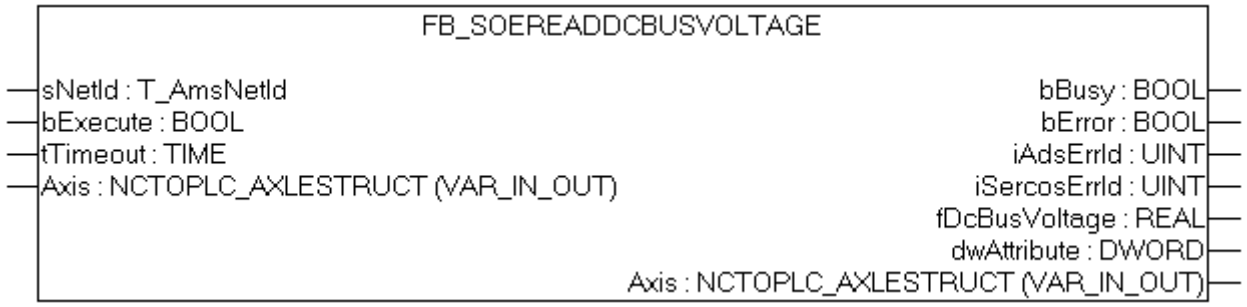
dwAttribute: Liefert das Attribut des Sercos-Parameters.

fDcBusCurrent: Liefert den DC-Bus-Strom (z.B. 2.040 entspricht 2.040A).

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEReadDcBusVoltage kann die DC-Bus-Spannung des Antriebs (S-0-0380) eingelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

dwAttribute: Liefert das Attribut des Sercos-Parameters.

fDcBusVoltage: Liefert die DC-Bus-Spannung (z.B. 294.0 entspricht 294.0V).

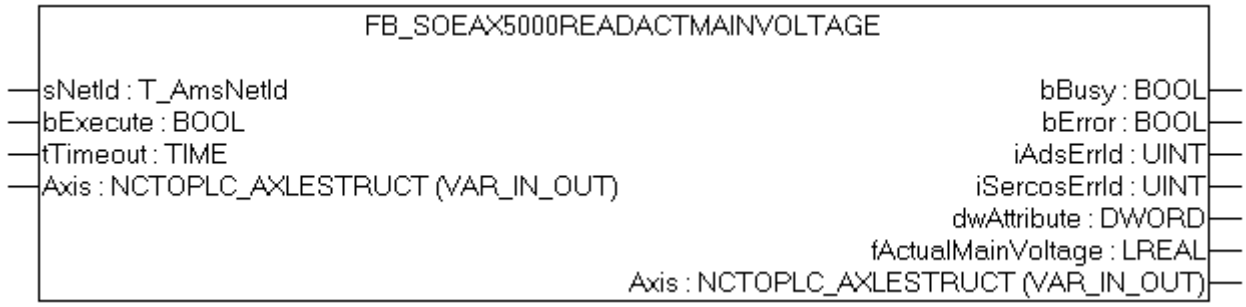
Beispiel

```
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 spezifische FBs

4.1 FB_SoEAX5000ReadActMainVoltage



Mit dem Funktionsbaustein FB_SoEAX5000ReadActMainVoltage kann der aktuelle Scheitelwert der Netzspannung des AX5000 (P-0-0200) eingelesen werden.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

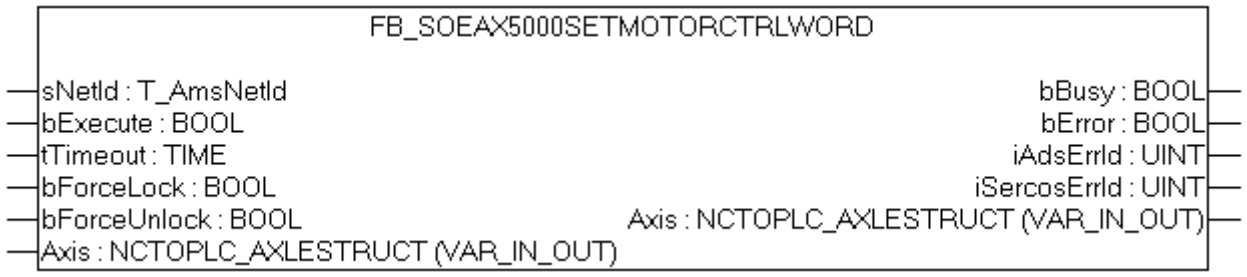
dwAttribute: Liefert das Attribut des Sercos-Parameters.

fActualMainVoltage: Liefert den Scheitelwert der aktuellen Netzspannung des AX5000 (z.B. 303.0 entspricht 303.0V).

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEAX5000SetMotorCtrlWord kann das ForceLock-Bit (Bit 0) bzw. das ForceUnlock-Bit im Motor Control Word (P-0-0096) gesetzt werden, um die Bremse zu setzen oder zu lösen. Im Normalfall wird die Bremse automatisch über das Enable des Antriebs gehandhabt.

Mit dem ForceLock-Bit kann die Bremse unabhängig vom Enable eingeworfen werden, mit dem ForceUnlock-Bit kann die Bremse unabhängig vom Enable gelöst werden. Bei gleichzeitig gesetztem ForceLock und ForceUnlock hat das ForceLock (Bremse gesetzt) die höhere Priorität.

VAR_INPUT

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

sNetId: Ist ein String, der die AMS-Netzwerkkennung des PCs enthält.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

bForceLock: Bremse unabhängig vom Enable aktivieren.

bForceUnlock: Bremse unabhängig vom Enable lösen.

VAR_IN_OUT

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

Axis: Achsstruktur.

VAR_OUTPUT

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

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

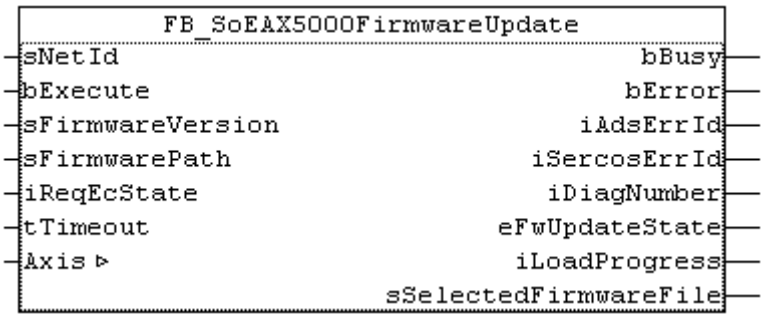
iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

Beispiel

```
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



Mit dem Funktionsbaustein FB_SoEAX5000FirmwareUpdate kann die Firmware des AX5000 überprüft und automatisch auf eine bestimme Version (Revision und Build) oder auf das aktuellste Build der konfigurierten Revision geändert werden.

Zum Updaten wird:
- der konfigurierte Slave-Typ ermittelt, z.B. AX5103-0000-0010
- der aktuelle Slave mit der vorgegebenen Slaveadresse ermittelt, z.B. AX5103-0000-0009
- die aktuelle Slavefirmware ermittelt, z.B. v1.05_b0009
- ein Vergleich der Konfiguration und des gefundenen Slaves, auf Anzahl der Kanäle, Strom, Revision, Firmware ausgeführt
- der Name des erforderlichen Firmware-Files ermittelt und die Datei gesucht

- der Firmwareupdate (falls erforderlich) ausfgefürt
- der aktuelle Slave mit der vorgegebenen Slaveadresse erneut ermittelt
- der Slave in den vorgegebenen EtherCAT-State geschaltet

Ein erfolgreicher Update endet mit **eFwUpdateState = eFwU_FwUpdateDone**, ist der Update nicht erforderlich, wird diese über **eFwUpdateState = eFwU_NoFwUpdateRequired** signalisiert. Der Firmwareupdate erfolgt über den angegeben Kanal (A=0 oder B=1) aus der stDriveRef. Bei zweikanaligen Geräten kann nur einer der beiden Kanäle hierfür verwendet werden. Der andere Kanal signalisiert das über den**eFwUpdateState = eFwU_UpdateViaOtherChannelActive** bzw. **= eFwU_UpdateViaOtherChannel**.

Während des Firmwareupdates (**eFwUpdateState = eFwU_FwUpdateInProgress**) signalisiert**iLoadProgress** den Fortschritt in Prozent.

HINWEIS

Fehlerhaftes Update durch Unterbrechungen

Unterbrechungen während des Updates können dazu führen, dass dieses nicht oder fehlerhaft ausgeführt wird. Die Klemme kann danach ohne die passende Firmware möglicherweise nicht mehr verwendet werden.

Während des Updates gilt:

- Die SPS und TwinCAT dürfen nicht gestoppt werden.
- Die EtherCAT-Verbindung darf nicht unterbrochen werden.
- Der AX5000 darf nicht ausgeschaltet werden.

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
```

sNetId: AMS-NetID der Steuerung (IPC).

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

sFirmwareVersion: Gibt die gewünschte Firmware-Version in Form von **v**x.yy**_b**nnnn an, z.B. "**v**1.05**_b**0009" für Version v1.05 Build 0009.

Release-Builds:
"v1.05_b**00**09" für ein spezifisches Build, zum Beispiel v1.05 Build 0009
"v1.05_b**00**??" aktuellstes Build einer vorgegebenen Version, zum Beispiel v1.05
"v1.??_b**00**??" aktuellstes Build einer vorgegebenen Hauptversion, zum Beispiel v1
"v?.??_b**00**??" aktuellstes Build der aktuellsten Version
"" aktuellstes Build der aktuellsten Version

Kundenspezifische Firmware-Builds:
"v1.05_b**10**09" für ein spezifisches Build, zum Beispiel v1.05 Build 0009
"v1.05_b**10**??" aktuellstes Build einer vorgegebenen Version, zum Beispiel v1.05
"v1.??_b**10**??" aktuellstes Build einer vorgegebenen Hauptversion, zum Beispiel v1
"v?.??_b**10**??" aktuellstes Build der aktuellsten Version

...

"v1.05_b**89**09" für ein spezifisches Build, zum Beispiel v1.05 Build 8909
"v1.05_b**89**??" aktuellstes Build einer vorgegebenen Version, zum Beispiel v1.05
"v1.??_b**89**??" aktuellstes Build einer vorgegebenen Hauptversion, zum Beispiel v1
"v?.??_b**89**??" aktuellstes Build der aktuellsten Version

Debug-Builds:
"v1.05_b**90**09" für ein spezifisches Build, zum Beispiel v1.05 Build 9009

"v1.05_b**90**??"	aktuellstes Build einer vorgegebenen Version, zum Beispiel v1.05
"v1.??_b**90**??"	aktuellstes Build einer vorgegebenen Hauptversion, zum Beispiel v1
"v?.??_b**90**??"	aktuellstes Build der aktuellsten Version

sFirmwarePath: Gibt den Pfad für den Firmwarepool an, in dem sich die Firmware-Dateien befinden, z.B. "C:\TwinCAT\Io\TcDriveManager\FirmwarePool".

iReqEcState: Gewünschter EtherCAT-State nach dem Update (nur wenn tatsächlich ein Update ausgeführt wird). Die States sind in der TcEtherCAT.lib als globale Konstanten definiert.

tTimeout: Da der Firmwareupdate bei großen EtherCAT-Netzwerken länger dauern kann, wird hier nur der Timeout für einzelne interne ADS-Instanzen vorgegeben.

VAR_IN_OUT

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

Axis: Achsstruktur.

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: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

iAdsErrId: Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehles

iSercosErrId: Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles

iDiagNumber: Liefert bei gesetztem bError-Ausgang den Antriebsfehler des letzten Firmware-Updates

eFwUpdateState: Liefert den Status der Firmware-Updates. Siehe E_FwUpdateState [▶ 38].

iLoadProgress: Liefert den Fortschritt des eigentlichen Firmware-Update in Prozent.

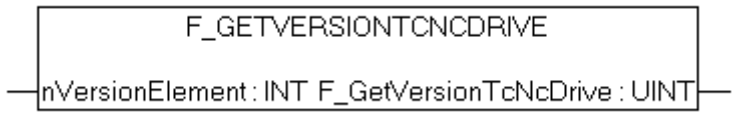
sSelectedFirmwareFile: Zeigt den Namen der gesuchten Firmware-Datei an.

Beispiel

```
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



Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.

FUNCTION F_GetVersionTcNcDrive : UINT

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

nVersionElement : Versionselement, das gelesen werden soll. Mögliche Parameter:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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

Der E_FwUpdateState beschreibt den Zustand eines Firmware-Updates.

```
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
```

Tab. 1: Update-Status

Parameter	Beschreibung
eFwU_NoError	Initialzustand
eFwU_CheckCfgIdentity	Einlesen des konfigurierten Slavetypen (Anzahl Kanäle, Strom, Revision)
eFwU_CheckSlaveCount	Ermitteln der konfigurierten Slaveanzahl
eFwU_CheckFindSlavePos	Suchen der Slave-Adresse im Master-Objektverzeichnis
eFwU_WaitForScan	Warten auf Online-Scan
eFwU_ScanningSlaves	Online-Scan der Slaves
eFwU_CheckScannedIdentity	Einlesen des gescannten Slavetypen (Anzahl Kanäle, Strom, Revision)
eFwU_CheckScannedFirmware	Einlesen der Firmware-Version
eFwU_FindFirmwareFile	Suchen nach der gewählten Firmware-Datei
eFwU_WaitForUpdate	Warten auf State des Updates
eFwU_WaitForSlaveState	Ermitteln des EtherCAT Slave-States
eFwU_StartFwUpdate	Starten des Firmware-Updates
eFwU_FwUpdateInProgress	Firmwareupdate aktiv
eFwU_FwUpdateDone	Firmwareupdate erfolgreich beendet
eFwU_NoFwUpdateRequired	Kein Firmwareupdate erforderlich

Parameter	Beschreibung
eFwU_UpdateViaOtherChannelActive	Update erfolgt über den anderen Achskanal
eFwU_UpdatedViaOtherChannel	Update erfolgte über den anderen Achskanal

Tab. 2: Update-Fehler

Parameter	Beschreibung
eFwU_GetSlaveIdentityError	Einlesen des konfigurierten Slavetypen schlug fehl, siehe iAdsErrId
eFwU_GetSlaveCountError	Ermitteln der konfigurierten Slaveanzahl schlug fehl, siehe iAdsErrId
eFwU_GetSlaveAddrError	Suchen der Slave-Adresse im Master-Objektverzeichnis schlug fehl, siehe iAdsErrId
eFwU_StartScanError	Starten des Online-Scan schlug fehl, siehe iAdsErrId
eFwU_ScanStateError	Online-Scan schlug fehl, siehe iAdsErrId
eFwU_ScanIdentityError	Einlesen des gescannten Slavetypen (Anzahl Kanäle, Strom, Revision) schlug fehl, siehe iAdsErrId
eFwU_GetSlaveStateError	Ermitteln des EtherCAT Slave-States schlug fehl, siehe iAdsErrId
eFwU_ScanFirmwareError	Einlesen der Firmware-Version schlug fehl, siehe iAdsErrId + iSercosErrId
eFwU_FindFileError	Suchen nach der gewählten Firmware-Datei schlug fehl, siehe iAdsErrId
eFwU_CfgTypeInNoAX5xxx	Der konfigurierte Slave ist kein AX5000
eFwU_ScannedTypeInNoAX5xxx	Der gescannte Slave ist kein AX5000
eFwU_ChannelMismatch	Anzahl der konfigurierten bzw. gefundenen Kanäle des AX5000 passen nicht zusammen
eFwU_ChannelMismatch_1Cfg_2Scanned	Einkanaliges Gerät konfiguriert, aber zweikanaliges Gerät gefunden
eFwU_ChannelMismatch_2Cfg_1Scanned	Zweikanaliges Gerät konfiguriert aber einkanaliges Gerät gefunden
eFwU_CurrentMismatch	AX5000-Type paßt vom Strom her nicht, z.B. AX5103 (3A) konfiguriert aber AX5106 (6A) gefunden
eFwU_FwUpdateError	Allgemeiner Updatefehler, siehe iAdsErrId
eFwU_ReqSlaveStateError	Umschalten in den gewünschten EtherCAT-State schlug fehl

Mehr Informationen:
www.beckhoff.de/tx1200

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