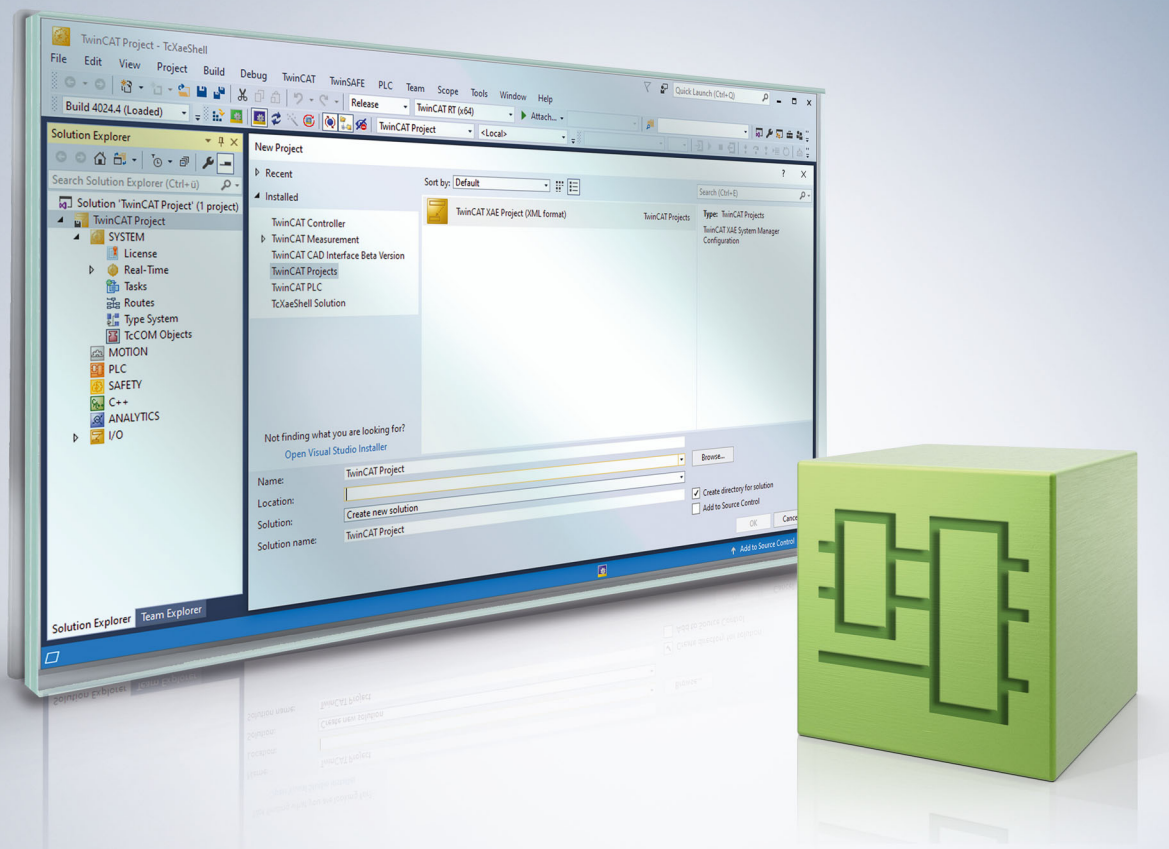
BECKHOFF New Automation Technology

Handbuch | DE

TE1000

TwinCAT 3 | PLC-Bibliothek: Tc2_EtherCAT



06.03.2024 | Version: 1.7.1

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, stets die aktuell gültige 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 weiterentwickelt.
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 Übersicht

Die SPS-Bibliothek Tc2_EtherCAT enthält Funktionsbausteine, mit denen Dienste bzw. Funktionen auf einem EtherCAT-Master-Gerät und/oder an dessen Slave-Geräten ausgeführt werden können.

3 EtherCAT Befehle

3.1 FB_EcPhysicalReadCmd

Mit dem Funktionsbaustein `FB_EcPhysicalReadCmd` kann ein EtherCAT-Lese-Kommando (FPRD, APRD, BRD) an einen bestimmten EtherCAT-Slave oder an alle EtherCAT-Slaves gesendet werden. Dieses Kommando kann von der SPS verwendet werden, um ein Register oder den DPRAM des EtherCAT Slave Controllers auszulesen.

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    adp      : UINT;
    ado      : UINT;
    len      : UDINT;
    eType    : E_EcAdressingType := eAdressingType_Fixed;
    pDstBuf  : PVOID;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
adp	UINT	Dieser Wert bestimmt, welcher EtherCAT-Slave mit diesem Kommando adressiert werden soll. Die Bedeutung dieses Wertes hängt von der mit eType ausgewählten Adressierungsart ab. (Siehe adp-Wert)
ado	UINT	Physikalischer Speicher (DPRAM) oder Register, das ausgelesen werden soll.
len	UDINT	Anzahl der zu lesenden Bytes.
eType	E_EcAdressingType	Abhängig vom Wert von eType werden verschiedene EtherCAT-Kommandos geschickt. (Siehe eType)
pDstBuf	PVOID	Die Adresse (Pointer) auf den Empfangspuffer.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

adp-Wert

Dieser Wert bestimmt, welcher EtherCAT-Slave mit diesem Kommando adressiert werden soll. Die Bedeutung dieses Wertes hängt von der mit eType ausgewählten Adressierungsart ab:

eType	Beschreibung
eAdressingType_Fixed	Der Slave wird anhand seiner konfigurierten EtherCAT-Adresse adressiert. Diese EtherCAT-Adressen können mithilfe des Bausteins FB_EcGetAllSlaveAddr ausgelesen werden.
eAdressingType_AutoInc	Der Slave wird anhand seiner Position im Ring adressiert. Der erste Teilnehmer hat die Adresse 0 (adp=0) und für alle darauf folgenden Slaves wird adp um eins dekrementiert: 1. Slave adp = 0 2. Slave adp = 16#ffff (-1) 3. Slave adp = 16#fffe(-2) 4. Slave adp = 16#fffd(-3) etc.
eAdressingType_BroadCAST	Alle Slaves werden von diesem Kommando adressiert. adp kann auf 0 gesetzt werden.

eType

Abhängig vom Wert von eType werden verschiedene EtherCAT-Kommandos geschickt:

eType	Kommando
eAdressingType_Fixed	Configured Address Physical Read (FPRD)
eAdressingType_AutoInc	Auto Increment Physical Read (APRD)
eAdressingType_BroadCAST	Broadcast Read (BRD)

Die einzelnen Kommandos unterscheiden sich in der Adressierungsart (siehe adp).

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    wkc    : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
wkc	UINT	Der Working Counter wird von jedem EtherCAT-Slave, der dieses Kommando erfolgreich bearbeitet hat, um eins inkrementiert. Wenn nur ein EtherCAT-Slave von diesem Kommando adressiert worden ist, muss dieser Wert somit 1 entsprechen.

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_PhysicalReadCmd
VAR
    fbReadCmd : FB_EcPhysicalReadCmd;
    bExecute  : BOOL;
    value     : UINT;
    adp       : UINT:=16#3E9;
    ado       : UINT:=16#1100;
    eType     : E_EcAdressingType := eAdressingType_Fixed;
    sNetId    : T_AmsNetId:='192.168.1.5.3.1';
    wkc       : UINT;
    bError    : BOOL;
    nErrId    : UDINT;
END_VAR
```

```
fbReadCmd (sNetId:=sNetID, ado:=ado, adp:=adp, eType:=eType, LEN := SIZEOF(value), pDstBuf:=ADR(valu
e), bExecute:=bExecute);
wkc := fbReadCmd.wkc;
bError:= fbReadCmd.bError;
nErrId:= fbReadCmd.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

3.2 FB_EcPhysicalWriteCmd

Mit dem Funktionsbaustein `FB_EcPhysicalWriteCmd` kann ein EtherCAT-Write-Kommando (FPWR, APWR, BWR) an einen bestimmten EtherCAT-Slave oder an alle EtherCAT-Slaves gesendet werden. Dieses Kommando kann von der SPS verwendet werden, um ein Register oder den DPRAM des EtherCAT Slave Controllers zu beschreiben.

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    adp      : UINT;
    ado      : UINT;
    len      : UDINT;
    eType    : E_EcAdressingType := eAdressingType_Fixed;
    pSrcBuf  : PVOID;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
adp	UINT	Dieser Wert bestimmt, welcher EtherCAT-Slave mit diesem Kommando adressiert werden soll. Die Bedeutung dieses Wertes hängt von der mit eType ausgewählten Adressierungsart ab. (Siehe adp-Wert)
ado	UINT	Physikalischer Speicher(DPRAM) oder Register das ausgelesen werden soll.
len	UDINT	Anzahl der zu schreibenden Bytes.
eType	E_EcAdressingType	Abhängig von dem Wert von eType werden verschiedene EhterCAT-Kommandos geschickt. (Siehe eType)
pSrcBuf	PVOID	Die Adresse (Pointer) auf den Sendepuffer.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

adp-Wert

Dieser Wert bestimmt, welcher EtherCAT-Slave mit diesem Kommando adressiert werden soll. Die Bedeutung dieses Wertes hängt von der mit eType ausgewählten Adressierungsart ab:

eType	Beschreibung
eAdressingType_Fixed	Der Slave wird anhand seiner konfigurierten EtherCAT-Adresse adressiert. Diese EtherCAT-Adressen können mithilfe des Bausteins FB_EcGetAllSlaveAddr ausgelesen werden.
eAdressingType_AutoInc	Der Slave wird anhand seiner Position im Ring adressiert. Der erste Teilnehmer hat die Adresse 0 (adp=0) und für alle darauf folgenden Slaves wird adp um eins dekrementiert: 1. Slave adp = 0 2. Slave adp = 16#ffff (-1) 3. Slave adp = 16#fffe(-2) 4. Slave adp = 16#fffd(-3) etc.
eAdressingType_BroadCAST	Alle Slaves werden von diesem Kommando adressiert. adp sollte auf 0 gesetzt werden.

eType

Abhängig von dem Wert von eType werden verschiedene EhterCAT-Kommandos geschickt:

eType	Kommando
eAdressingType_Fixed	Configured Address Physical Write (FPWR)
eAdressingType_AutoInc	Auto Increment Physical Write (APWR)
eAdressingType_BroadCAST	Broadcast Write (BWR)

Die einzelnen Kommandos unterscheiden sich in der Adressierungsart (siehe adp).

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    wkc    : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
wkc	UINT	Der Working Counter wird von jedem EtherCAT-Slave, der dieses Kommando erfolgreich bearbeitet hat, um eins inkrementiert. Wenn nur ein EtherCAT-Slave von diesem Kommando adressiert worden ist, muss dieser Wert somit 1 entsprechen.

Beispiel für eine Implementierung in ST:

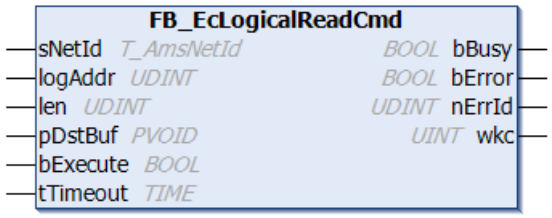
```
PROGRAM Test_PhysicalWriteCmd
VAR
    fbWriteCmd : FB_EcPhysicalWriteCmd;
    bExecute   : BOOL;
    value      : UINT :=16#5555;
    adp        : UINT:=16#3E9;
    ado        : UINT:=16#1100;
    eType      : E_EcAdressingType := eAdressingType_Fixed;
```

```
    sNetId     : T_AmsNetId:='192.168.1.5.3.1';
    wkc        : UINT;
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR

fbWriteCmd (sNetId:=sNetID, ado:=ado, adp:=adp, eType:=eType, LEN := SIZEOF(value), pSrcBuf:=ADR(val
ue), bExecute:=bExecute);
wkc := fbWriteCmd.wkc;
bError:= fbWriteCmd.bError;
nErrId:= fbWriteCmd.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

3.3 FB_EcLogicalReadCmd



Mit dem Funktionsbaustein `FB_EcLogicalReadCmd` wird ein logisches EtherCAT-Write-Kommando (LRD) vom Master gesendet. In jedem Slave können lokale Adressbereiche (DPRAM) auf globale logische Adressebereich gemappt werden. Somit adressiert dieses Kommando alle EtherCAT-Slaves, die ein Mapping für den ausgewählten logischen Adressbereich konfiguriert haben.

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    logAddr  : UDINT;
    len      : UDINT;
    pDstBuf  : PVOID;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
logAddr	UDINT	Logische Adresse
len	UDINT	Anzahl der zu lesenden Bytes
pDstBuf	PVOID	Adresse (Pointer) auf den Empfangspuffer
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    wkc    : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
wkc	UINT	Der Working Counter wird von jedem EtherCAT-Slave, der dieses Kommando erfolgreich bearbeitet hat, um eins inkrementiert. Wenn nur ein EtherCAT-Slave von diesem Kommando adressiert worden ist, muss dieser Wert somit 1 entsprechen.

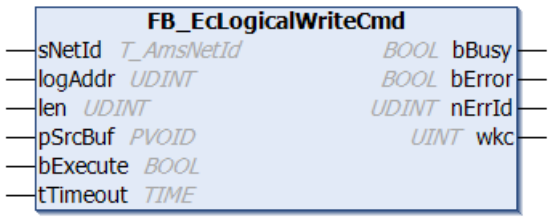
Beispiel für eine Implementierung in ST:

```
PROGRAM Test_LogicalReadCmd
VAR
    fbReadCmd : FB_EcLogicalReadCmd;
    bExecute  : BOOL;
    value     : USINT;
    logAddr   : UDINT :=16#10000;
    sNetId    : T_AmsNetId:='192.168.1.5.3.1';
    wkc       : UINT;
    bError    : BOOL;
    nErrId    : UDINT;
END_VAR

fbReadCmd (sNetId:=sNetID, logAddr:=logAddr, LEN := SIZEOF(value), pDstBuf:=ADR(value), bExecute:=bE
xecute);
wkc := fbReadCmd.wkc;
bError:= fbReadCmd.bError;
nErrId:= fbReadCmd.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

3.4 FB_EcLogicalWriteCmd



Mit dem Funktionsbaustein `FB_EcLogicalWriteCmd` wird ein logisches EtherCAT-Write-Kommando (LWR) vom Master gesendet. In jedem Slave können lokale Adressbereiche (DPRAM) auf globale logische Adressebereich gemappt werden. Somit adressiert dieses Kommando alle EtherCAT-Slaves, die ein Mapping für den ausgewählten logischen Adressbereich konfiguriert haben.

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    logAddr  : UDINT;
    len      : UDINT;
    pSrcBuf  : PVOID;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
logAddr	UDINT	Logische Adresse
len	UDINT	Anzahl der zu schreibenden Bytes
pSrcBuf	PVOID	Adresse (Pointer) auf den Sendepuffer
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    wkc    : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
wkc	UINT	Der Working Counter wird von jedem EtherCAT-Slave, der dieses Kommando erfolgreich bearbeitet hat, um eins inkrementiert. Wenn nur ein EtherCAT-Slave von diesem Kommando adressiert worden ist, muss dieser Wert somit 1 entsprechen.

Beispiel für eine Implementierung in ST:

```
PROGRAM Test_LogicalWriteCmd
VAR
    fbWriteCmd : FB_EcLogicalWriteCmd;
    bExecute   : BOOL;
    value      : USINT :=16#55;
    logAddr    : UDINT :=16#10000;
    sNetId     : T_AmsNetId:='192.168.1.5.3.1';
    wkc        : UINT;
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR

fbWriteCmd (sNetId:=sNetID, logAddr:=logAddr, LEN := SIZEOF(value), pSrcBuf:=ADR(value), bExecute:=b
Execute);
wkc := fbWriteCmd.wkc;
bError :=fbWriteCmd.bError;
nErrId :=fbWriteCmd.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4 EtherCAT Diagnose

4.1 FB_EcGetAllSlaveAbnormalStateChanges

Mit dem Funktionsbaustein `FB_EcGetAllSlaveAbnormalStateChanges` können die unerwarteten EtherCAT-Zustandswechsel von allen an den Master angeschlossenen Slaves ausgelesen werden. Bei erfolgreichem Aufruf enthält der im Parameter pBufAddr übergebene Puffer die Anzahl der unerwarteten Zustandswechsel aller Slaves als Array von UDINTs. EtherCAT-Zustandswechsel sind dann unerwartet, wenn sie nicht vom EtherCAT-Master angefordert sind, z.B. wenn ein EtherCAT-Slave von sich aus vom Zustand OP in den Zustand SAFEOP fällt.

Eingänge

```
VAR_INPUT
    sNetId   :  T_AmsNetId; (*AmsNetId of the EtherCAT master  device*)
    pAddrBuf :  POINTER TO ARRAY [0..EC_MAX_SLAVES] OF UDINT;
    (*Contains the address of the buffer the counters for the state changes f.i. Op to SafeOp-
Err are copied to.*)
    cbBufLen :  UDINT; (*Size of the buffer pAddrBuf.     The size of the buffer must be at least nS
lave *4 Bytes *)
    bExecute :  BOOL; Function Block execution is triggered by a rising    edge at this input*)
    tTimeout :  TIME; (*States the time before the function is cancelled.*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pAddrBuf	POINTER TO ARRAY [0..EC_MAX_SLAVES] OF UDINT	Adresse eines Arrays von UDINTs, in das die Anzahl der unerwarteten Zustandswechsel der einzelnen Slaves geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Funktionsbaustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.2 FB_EcGetAllSlaveAddr

Mit dem Funktionsbaustein `FB_EcGetAllSlaveAddr` können die Adressen von allen an den Master angeschlossenen Slaves ausgelesen werden. Bei erfolgreichem Aufruf enthält der im Parameter pAddrBuf übergebene Puffer die Adressen aller Slaves als Array von UINTs.

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    pAddrBuf : POINTER TO ARRAY[0..EC_MAX_SLAVES] OF UINT;
    cbBufLen : UDINT
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pAddrBuf	POINTER TO ARRAY [0..EC_MAX_SLAVES] OF UINT	Adresse eines Arrays von UINTs, in das die Adressen der einzelnen Slaves geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves.

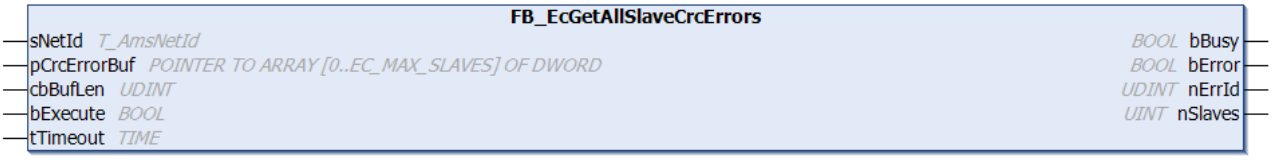
Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetAllSlaveAddresses
VAR
    fbGetAllSlaveAddr : FB_EcGetAllSlaveAddr;
    sNetId            : T_AmsNetId := '172.16.2.131.2.1';
    bExecute          : BOOL;
    slaveAddresses    : ARRAY[0..255] OF UINT;
    nSlaves           : UINT := 0;
    bError            : BOOL;
    nErrId            : UDINT;
END_VAR

fbGetAllSlaveAddr(sNetId:= sNetId,pAddrBuf := ADR(slaveAddresses), cbBufLen:= SIZEOF(slaveAddresses)
, bExecute:=bExecute);
nSlaves := fbGetAllSlaveAddr.nSlaves;
bError := fbGetAllSlaveAddr.bError;
nErrId := fbGetAllSlaveAddr.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.3 FB_EcGetAllSlaveCrcErrors



Mit dem Funktionsbaustein `FB_EcGetAllSlaveCrcErrors` können die CRC-Error-Zähler aller am Master angeschlossenen Slaves ausgelesen werden. Die CRC-Error an den einzelnen Ports eines Slaves werden addiert.

Um die CRC-Error der einzelnen Ports (A, B und C) eines Slaves auszulesen, muss der Funktionsbock FB_EcGetSlaveCrcError [▶ 28] aufgerufen werden.

Um die CRC-Error der einzelnen Ports (A, B, C und D) eines Slaves auszulesen, muss der Funktionsbock FB_EcGetSlaveCrcErrorEx [▶ 29] aufgerufen werden.

Eingänge

```
VAR_INPUT
    sNetId       : T_AmsNetId;
    pCrcErrorBuf : POINTER TO ARRAY[0..EC_MAX_SLAVES] OF DWORD;
    cbBufLen     : UDINT;
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pCrcErrorBuf	POINTER TO ARRAY [0..EC_MAX_SLAVES] OF DWORD	Adresse eines Arrays von DWORDs, in das die CRC-Error-Zähler geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves.

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetAllSlaveCrcErrors
VAR
    fbGetAllSlaveCrcErrors : FB_EcGetAllSlaveCrcErrors;
    sNetId                 : T_AmsNetId := '172.16.2.131.2.1';
    bExecute               : BOOL;
    crcErrors              : ARRAY[0..255] OF DWORD;
    nSlaves                : UINT := 0;
    bError                 : BOOL;
    nErrId                 : UDINT;
END_VAR

fbGetAllSlaveCrcErrors(sNetId:= sNetId, pCrcErrorBuf := ADR(crcErrors), cbBufLen:= SIZEOF(crcErrors)
, bExecute:=bExecute);
nSlaves := fbGetAllSlaveCrcErrors.nSlaves;
bError := fbGetAllSlaveCrcErrors.bError;
nErrId := fbGetAllSlaveCrcErrors.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.4 FB_EcGetAllSlavePresentStateChanges

Mit dem Funktionsbaustein `FB_EcGetAllSlavePresentStateChanges` können die EtherCAT-Zustandswechsel vom Zustand „Slave ist vorhanden“ in „INIT_NO_COMM“ von allen an den Master angeschlossenen Slaves ausgelesen werden. Bei erfolgreichem Aufruf enthält der im Parameter pAddrBuf übergebene Puffer die Anzahl der Zustandswechsel aller Slaves als Array von UDINTs. Der EtherCAT-Zustandswechsel von „Slave ist vorhanden“ zu „INIT_NO_COMM“ bedeutet, dass die Verbindung zum Slave unterbrochen wurde. Zum Beispiel durch Abziehen des EtherCAT-Kabels.

Eingänge

```
VAR_INPUT
    sNetId    :  T_AmsNetId; (*AmsNetId of the EtherCAT master device*)
    pAddrBuf  :  POINTER TO ARRAY [0..EC_MAX_SLAVES] OF UDINT; (*Contains the address of the buffer
the counters for the state changes from Present to INIT_NO_COMM are copied to.*)
    cbBufLen  :  UDINT; (*Size of the buffer pAddrBuf. The size of the buffer must be at least nSlav
e *4 Bytes *)
```

```
    bExecute  :  BOOL; (*Function Block execution is triggered by a rising edge at this input*)
    tTimeout  :  TIME; (*States the time before the function is cancelled.*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pAddrBuf	POINTER TO ARRAY [0..EC_MAX_SLAVES]OF UDINT	Adresse eines Arrays von UDINTs, in das die Anzahl der Zustandswechsel von "Slave ist vorhanden" in INIT_NO_COMM der einzelnen Slaves geschrieben werden sollen.
bBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Funktionsbaustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.5 FB_EcGetConfSlaves

Mit dem Funktionsbaustein `FB_EcGetConfSlaves` kann eine Liste der konfigurierten Slaves aus dem EtherCAT-Master-Objektverzeichnis ausgelesen werden.

Eingänge

```
VAR_INPUT
    sNetId              : T_AmsNetId;
    pArrEcConfSlaveInfo : POINTER TO ARRAY[0..EC_MAX_SLAVES] OF ST_EcSlaveConfigData;
    cbBufLen            : UDINT;
    bExecute            : BOOL;
    tTimeout            : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pArrEcConf SlaveInfo	POINTER TO ARRAY [0..EC_MAX_SLAVES] OF ST_ EcSlaveConfigData	Adresse eines Arrays von Strukturen des Typs ST_EcSlaveConfigData [▶ 118], in das Daten von jedem konfigurierten Slave geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

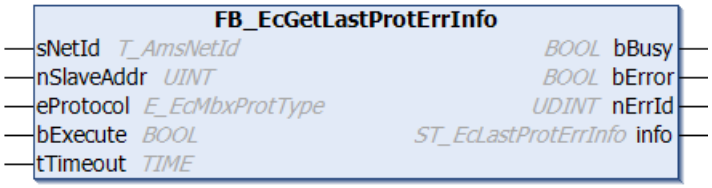
```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Liefert die Anzahl der konfigurierten Slaves.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.6 FB_EcGetLastProtErrInfo



Mit dem Funktionsbaustein `FB_EcGetLastProtErrInfo` können zusätzliche Fehlerinformationen zum zuletzt aufgetretenen Mailboxprotokollfehler ausgelesen werden. Ein fehlerfreies Mailboxkommando setzt den letzten Fehler jedes Mal zurück.

Eingänge

```
VAR_INPUT
    sNetId     : T_AmsNetId;
    nSlaveAddr : UINT;
```

```
    eProtocol  : E_EcMbxProtType := eEcMbxProt_FoE;
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen Fehlerinformationen ausgelesen werden sollen.
eProtocol	E_EcMbxProtType	EtherCAT-Mailboxprotokolltyp [▶ 116]
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    info   : ST_EcLastProtErrInfo;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
info	ST_EcLastProtErr Info	Struktur mit zusätzlichen Fehlerinformationen [▶ 117]

Beispiel in ST:

Bei einer steigenden Flanke am bGet werden zusätzliche Fehlerinformationen zum zuletzt aufgetretenen Mailboxprotokollfehler ausgelesen.

```
PROGRAM MAIN
VAR
    fbGetInfo : FB_EcGetLastProtErrInfo := ( sNetID := '172.16.6.195.2.1',
                        nSlaveAddr := 1004,
                        eProtocol := eEcMbxProt_FoE,
                        tTimeout := DEFAULT_ADS_TIMEOUT );
    bGet : BOOL;
    bBusy : BOOL;
    bError : BOOL;
    nErrID : UDINT;
    sInfo : T_MaxString;
END_VAR

fbGetInfo( bExecute:= bGet,
        bBusy=>bBusy,
        bError=>bError,
        nErrId=>nErrId );

sInfo := BYTEARR_TO_MAXSTRING( fbGetInfo.info.binDesc );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.7 FB_EcGetMasterDevState

Mit dem Funktionsbaustein `FB_EcGetMasterDevState` kann der aktuelle Zustand des EtherCAT-Masters gelesen werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

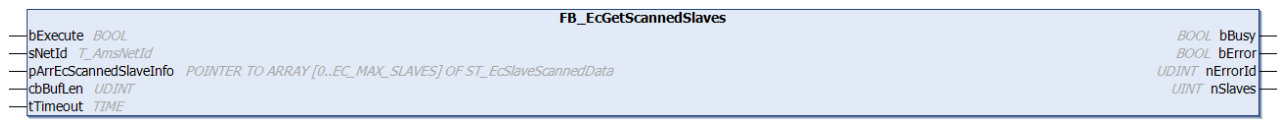
```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    nDevState : WORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
nDevState	WORD	Aktueller Zustand des Master-Geräts

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.8 FB_EcGetScannedSlaves



Mit dem Funktionsbaustein `FB_EcGetScannedSlaves` kann eine Liste der aktuell verfügbaren (gescannten) Slaves aus dem EtherCAT-Master-Objektverzeichnis ausgelesen werden. Hierfür wird ein Online-Scan ausgeführt, bei dem die EEPROMs der EtherCAT-Slaves eingelesen werden. Das Scannen nimmt je nach Anzahl der angeschlossenen Slaves einige Zeit in Anspruch.

Eingänge

```
VAR_INPUT
    bExecute               : BOOL;
    sNetId                 : T_AmsNetId;
    pArrEcScannedSlaveInfo : POINTER TO ARRAY[0..EC_MAX_SLAVES] OF ST_EcSlaveScannedData;
    cbBufLen               : UDINT;
    tTimeout               : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pArrEcScannedSlaveInfo	POINTER TO ARRAY[0..EC_MAX_SLAVES] OF ST_EcSlaveScannedData	Adresse eines Arrays von Strukturen des Typs ST_EcSlaveScannedData [▶ 119], in das Daten jeden gescannten Slaves geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
ND_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Liefert die Anzahl der gescannten Slaves.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.9 FB_EcGetSlaveCount

Mit dem Funktionsbaustein `FB_EcGetSlaveCount` kann die Anzahl der Slaves, die an den Master angeschlossen sind, ermittelt werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetSlaveCount
VAR
    fbGetSlaveCount : FB_EcGetSlaveCount;
    sNetId          : T_AmsNetId := '172.16.2.131.2.1';
    bExecute        : BOOL;
    nSlaves         : UINT;
    bError          : BOOL;
    nErrId          : UDINT;
END_VAR

fbGetSlaveCount(sNetId:= sNetId, bExecute:=bExecute);
nSlaves := fbGetSlaveCount.nSlaves;
bError := fbGetSlaveCount.bError;
nErrId := fbGetSlaveCount.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.10 FB_EcGetSlaveCrcError

Mit dem Funktionsbaustein `FB_EcGetSlaveCrcError` können die CRC-Error-Zähler der einzelnen Ports (A, B und C) eines Slave ausgelesen werden. Bei erfolgreichem Aufruf enthält die Ausgangsvariable crcError vom Typ `ST_EcCrcError` die angeforderten CRC-Error-Zähler.

Der Funktionsbaustein FB_EcGetSlaveCrcError kann nur mit Slaves mit bis zu 3 Ports (z.B. EK1100) eingesetzt werden. Der Funktionsbaustein `FB_EcGetSlaveCrcErrorEx` kann auch mit Slaves mit bis zu 4 Ports (z.B. EK1122) eingesetzt werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen CRC-Error-Zähler ausgelesen werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
    nErrId   : UDINT;
    crcError : ST_EcCrcError;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
crcError	ST_EcCrcError	CRC-Error [▶ 116]-Zähler der einzelnen Ports

Beispiel für eine Implementierung in ST:

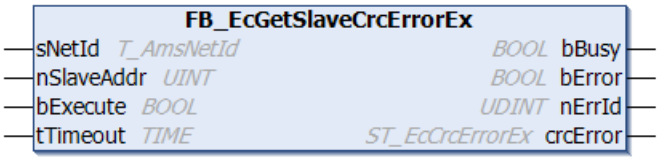
```
PROGRAM TEST_GetSlaveCrcError
VAR
    fbGetSlaveCrcError : FB_EcGetSlaveCrcError;
    sNetId : T_AmsNetId := '172.16.2.131.2.1';
    bExecute : BOOL;
    crcError : ST_EcCrcError;
    nSlaveAddr : UINT := 1001;
```

```
    bError : BOOL;
    nErrId : UDINT;
END_VAR

fbGetSlaveCrcError(sNetId:= sNetId, nSlaveAddr:= nSlaveAddr, bExecute:=bExecute);
crcError := fbGetSlaveCrcError.crcError;
bError := fbGetSlaveCrcError.bError;
nErrId := fbGetSlaveCrcError.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.11 FB_EcGetSlaveCrcErrorEx



Mit dem Funktionsbaustein `FB_EcGetSlaveCrcErrorEx` können die CRC-Error-Zähler der einzelnen Ports (A, D, B und C) eines Slave ausgelesen werden. Bei erfolgreichem Aufruf enthält die Ausgangsvariable crcError vom Typ `ST_EcCrcErrorEx` die angeforderten CRC-Error-Zähler.

Der Funktionsbaustein `FB_EcGetSlaveCrcErrorEx` kann mit Slaves mit bis zu 4 Ports (z.B. EK1122) eingesetzt werden. Der Funktionsbaustein `FB_EcGetSlaveCrcError` kann hingegen nur mit Slaves mit bis zu 3 Ports (z.B. EK1100) eingesetzt werden.

Eingänge

```
VAR_INPUT
    sNetId      :  T_AmsNetId; (*AmsNetId of the EtherCAT master device*)
    nSlaveAddr  :  UINT; (*Address of the slave device*)
    bExecute    :  BOOL; (*Function block execution is triggered by a rising edge at this input.*)
    tTimeout    :  TIME; (*States the time before the function is cancelled.*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält (Typ: T_AmsNetId).
nSlaveAddr	UINT	Feste Adresse der EtherCAT-Slaves, dessen CRC-Error-Zähler ausgelesen werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy     :  BOOL;
    bError    :  BOOL;
    nErrId    :  UDINT;
    CrcError  :  ST_EcCrcErrorEx; (*Crc error of the EtherCAT slave device*)
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
CrcError	ST_EcCrcErrorEx	CRC-Error-Zähler der einzelnen Ports (Typ: ST_EcCrcErrorEx [▶ 117])

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.12 FB_EcGetSlaveIdentity

Mit dem Funktionsbaustein `FB_EcGetSlaveIdentity` kann die CANopen Identity eines einzelnen EtherCAT-Slave-Gerätes ausgelesen werden. Bei erfolgreichem Aufruf enthält die Ausgangsvariable identity vom Typ `ST_EcSlaveIdentity` die angeforderte Identity-Information.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
    nErrId   : UDINT;
    identity : ST_EcSlaveIdentity;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
identity	ST_EcSlaveIdentity	CANopen Identity [▶ 119] des EtherCAT-Gerätes

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetSlaveIdentity
VAR
    fbGetSlaveIdentity : FB_EcGetSlaveIdentity;
    sNetId             : T_AmsNetId := '172.16.2.131.2.1';
    bExecute           : BOOL;
    identity           : ST_EcSlaveIdentity;
    nSlaveAddr         : UINT := 1001;
    bError             : BOOL;
    nErrId             : UDINT;
END_VAR

fbGetSlaveIdentity(sNetId:= sNetId, nSlaveAddr:= nSlaveAddr, bExecute:=bExecute);
identity := fbGetSlaveIdentity.identity;
bError := fbGetSlaveIdentity.bError;
nErrId := fbGetSlaveIdentity.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.13 FB_EcGetSlaveTopologyInfo

Mit dem Funktionsbaustein `FB_EcGetSlaveTopolgyInfo` können Informationen zur Topologie ermittelt werden.

Eingänge

```
VAR_INPUT
    sNetId    :  T_AmsNetId; (*AmsNetId of the EtherCAT master device*)
    pAddrBuf  :  POINTER TO ARRAY [0..EC_MAX_SLAVES] OF ST_TopologyDataEx; (*Contains the address of
 the buffer the topology data are copied to.*)
    cbBufLen  :  UDINT; (*Size of the buffer pAddrBuf. The size of the buffer must be at least nSlav
e * 64 Bytes*)
    bExecute  :  BOOL; (*Function block execution is triggered by a rising edge at this input*)
    tTimeout  :  TIME; (*States the time before the function is cancelled*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pAddrBuf	POINTER TO ARRAY [0..EC_MAX_SLAVES] OF ST_TopologyDataEx	Adresse eines Arrays von Strukturen vom Typ ST_TopologyDataEx [▶ 123], das die Topologiedaten enthält.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Funktionsbaustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

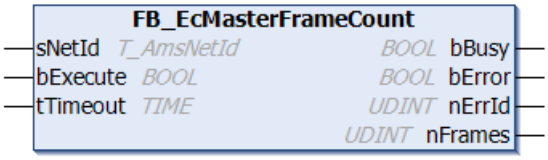
```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.14 FB_EcMasterFrameCount



Mit dem Funktionsbaustein `FB_EcMasterFrameCount` kann die Anzahl der EtherCAT-Frames, die im Master konfiguriert sind, ermittelt werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

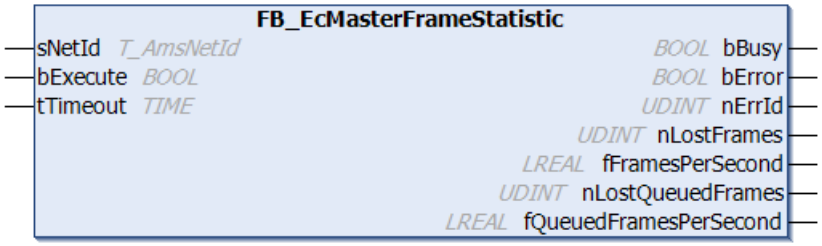
```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nFrames : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
nFrames	UDINT	Anzahl der EtherCAT-Frames

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.15 FB_EcMasterFrameStatistic



Mit dem Funktionsbaustein `FB_EcMasterFrameStatistic` kann die Frame-Statistik des EtherCAT-Masters ausgelesen werden. Es wird zwischen zyklischen und azyklischen (queued) Frames unterschieden. Azyklische Frames werden z.B. für die Initialisierung oder für Parameterzugriffe auf EtherCAT-Slaves verwendet. Frames gelten dann als verloren, wenn sie nicht zum Master zurückkommen oder wenn sie ungültig sind.
Die Anzahl der "Lost Frames" (die verloren gegangenen oder ungültigen zyklischen Frames), die Anzahl der zykischen Frames pro Sekunde, die Anzahl der "Lost Queued Frames" (die verloren gegangenen oder ungültigen azyklischen Frames) und die Anzahl der "Queued Frames" pro Sekunde werden am Ausgang des Bausteins zur Verfügung gestellt.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy                : BOOL;
    bError               : BOOL;
    nErrId               : UDINT;
    nLostFrames          : UDINT;
    fFramesPerSecond     : LREAL;
    nLostQueuedFrames    : UDINT;
    fQueuedFramesPerSecond : LREAL;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
nLostFrames	UDINT	Liefert die momentane Anzahl der verloren gegangenen bzw. ungültigen zyklischen Frames.
fFramesPer Second	LREAL	Liefert die momentane Anzahl der zyklischen Frames pro Sekunde.
nLostQueuedFra mes	UDINT	Liefert die momentane Anzahl der verloren gegangenen bzw. ungültigen Queued (azyklischen) Frames.
fQueuedFramesP erSecond	LREAL	Liefert die momentane Anzahl der Queued (azyklischen) Frames pro Sekunde.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.16 FB_EcMasterFrameStatisticClearCRC

Mit dem Funktionsbaustein `FB_EcMasterFrameStatisticClearCRC` können die CRC-Fehlerzähler aller EtherCAT-Slaves gelöscht werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.17 FB_EcMasterFrameStatisticClearFrames

Mit dem Funktionsbaustein `FB_EcMasterFrameStatisticClearFrames` können die Zähler der "Lost Frames" gelöscht werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

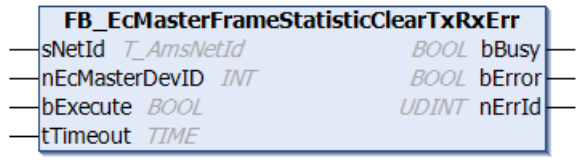
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.18 FB_EcMasterFrameStatisticClearTxRxErr



Mit dem Funktionsbaustein `FB_EcMasterFrameStatisticClearTxRxErr` können die Fehlerzähler des Miniport-Treiber der Netzwerkkarte gelöscht werden.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung der CPU (PC) enthält. (Typ: T_AMSNetId)
nEcMasterDevID	INT	Device ID des EtherCAT-Masters.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

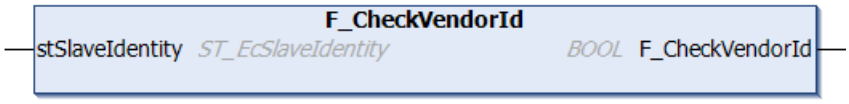
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.19 F_CheckVendorId



Die Funktion `F_CheckVendorId` liefert ein TRUE, falls die VendorID Beckhoff ist, ansonsten ein FALSE.

Rückgabewert

```
METHOD F_CheckVendorId : BOOL
```

Name	Typ	Beschreibung
F_CheckVendorId	BOOL	TRUE, falls die VendorID Beckhoff ist, ansonsten FALSE.

Eingänge

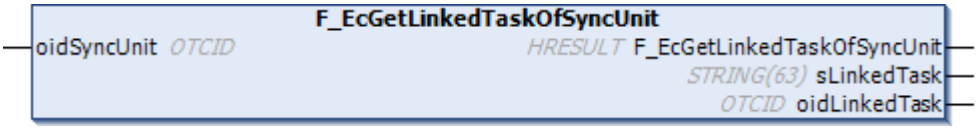
```
VAR_INPUT
    stSlaveIdentity : ST_EcSlaveIdentity;
END_VAR
```

Name	Typ	Beschreibung
stSlaveIdentity	ST_EcSlaveIdentity	Slave Identity, die mit dem FB_EcGetSlaveIdentity [▶ 30] eingelesen werden kann.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

4.20 F_EcGetLinkedTaskOfSyncUnit



Mit dieser Funktion können Name und Objekt-ID der verknüpften Task einer EtherCAT Sync Unit ausgelesen werden. Der Rückgabewert der Funktion signalisiert, ob der Aufruf erfolgreich war und gibt im Fehlerfall den entsprechenden Fehlercode aus.

Rückgabewert

```
METHOD F_EcGetLinkedTaskOfSyncUnit : HRESULT
```

Name	Typ	Beschreibung
F_EcGetLinkedTaskOfSyncUnit	HRESULT	Signalisiert, ob der Aufruf erfolgreich war und gibt im Fehlerfall den entsprechenden Fehlercode aus.

Eingänge

```
VAR_INPUT
    oidSyncUnit : OTCID; // object ID of sync unit
END_VAR
```

Name	Typ	Beschreibung
oidSyncUnit	OTCID	An diesem Eingang wird die Objekt-ID der Sync Unit angegeben. Diese ist im Prozessabbild des EtherCAT Masters zu finden.

Ausgänge

```
VAR_OUTPUT
    sLinkedTask   : STRING;
    oidLinkedTask : OTCID; // object ID of linked task
END_VAR
```

Name	Typ	Beschreibung
sLinkedTask	STRING	Liefert den Namen der verknüpfen Task.
oidLinkedTask	OTCID	Liefert die Objekt-ID der verknüpften Task.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.4024.22	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.3.17.0

4.21 F_EcGetSyncUnitName

Mit dieser Funktion kann der Name einer EtherCAT Sync Unit über deren Objekt-ID ausgelesen werden. Der Rückgabewert der Funktion signalisiert, ob der Aufruf erfolgreich war und gibt im Fehlerfall den entsprechenden Fehlercode aus.

Rückgabewert

```
METHOD F_EcGetSyncUnitName : HRESULT
```

Name	Typ	Beschreibung
F_EcGetSyncUnitName	HRESULT	Signalisiert, ob der Aufruf erfolgreich war und gibt im Fehlerfall den entsprechenden Fehlercode aus.

Eingänge

```
VAR_INPUT
    oidSyncUnit : OTCID; // object ID of sync unit
END_VAR
```

Name	Typ	Beschreibung
oidSyncUnit	OTCID	An diesem Eingang wird die Objekt-ID der Sync Unit angegeben. Diese ist im Prozessabbild des EtherCAT Masters zu finden.

Ausgänge

```
VAR_OUTPUT
    sSyncUnitName : STRING(63);
END_VAR
```

Name	Typ	Beschreibung
sSyncUnitName	STRING	Liefert den Namen der Sync Unit.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.4024.48	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.4.2.0

5 EtherCAT State Machine

5.1 FB_EcGetAllSlaveStates

Mit dem Funktionsbaustein `FB_EcGetAllSlaveStates` kann der EtherCAT-Status und der Link-Status von allen an den Master angeschlossenen Slaves ausgelesen werden. Bei erfolgreichem Aufruf enthält der im Parameter pStateBuf übergebene Puffer die angeforderte Status-Information als Array vom Typ ST_EcSlaveState.

Eingänge

```
VAR_INPUT
    sNetId    : T_AmsNetId;
    pStateBuf : POINTER TO ARRAY[0..EC_MAX_SLAVES] OF ST_EcSlaveState;
    cbBufLen  : UDINT;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
pStateBuf	POINTER TO ARRAY[0..EC_MAX_SLAVES] OF ST_EcSlaveState	Adresse eines ARRAY of ST_EcSlaveStates [▶ 121], in das die Slave-Status geschrieben werden sollen.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls. Fehler 1798 (0x706) verweist auf einen Nullpointer an der Pufferadresse. Fehler 1797 (0x705) verweist auf einen zu kleinen Puffer.
nSlaves	UINT	Anzahl der an den Master angeschlossenen Slaves

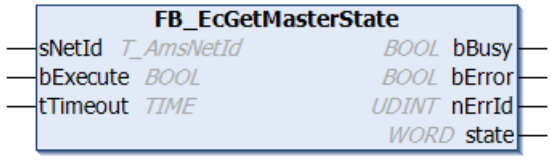
Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetAllSlaveStates
VAR
    fbGetAllSlaveStates : FB_EcGetAllSlaveStates;
    sNetId              : T_AmsNetId := '172.16.2.131.2.1';
    bExecute            : BOOL;
    devStates           : ARRAY[0..255] OF ST_EcSlaveState;
    nSlaves             : UINT := 0;
    bError              : BOOL;
    nErrId              : UDINT;
END_VAR

fbGetAllSlaveStates(sNetId:= sNetId, pStateBuf := ADR(devStates), cbBufLen:=SIZEOF(devStates), bExec
ute:=bExecute);
nSlaves := fbGetAllSlaveStates.nSlaves;
bError := fbGetAllSlaveStates.bError;
nErrId := fbGetAllSlaveStates.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.2 FB_EcGetMasterState



Mit dem Funktionsbaustein `FB_EcGetMasterState` kann der EtherCAT-Zustand des Masters ausgelesen werden. Bei erfolgreichem Aufruf enthält die Ausgangsvariable state vom Typ WORD die angeforderte Status-Information.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    state  : WORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
state	WORD	Aktueller EtherCAT-Zustand des Masters. (Siehe State)

state

Aktueller EtherCAT-Zustand des Masters. Die möglichen Werte sind:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Master ist im Init-Zustand
EC_DEVICE_STATE_PREOP	0x02	Master ist im Pre-Operational-Zustand
EC_DEVICE_STATE_SAFEOP	0x04	Master ist im Safe-Operational-Zustand
EC_DEVICE_STATE_OP	0x08	Master ist im Operational -Zustand

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetMasterState
VAR
    fbGetMasterState : FB_EcGetMasterState;
    sNetId           : T_AmsNetId := '172.16.2.131.2.1';
    bExecute         : BOOL;
    state            : WORD;
    bError           : BOOL;
    nErrId           : UDINT;
END_VAR

fbGetMasterState(sNetId:= sNetId, bExecute:=bExecute);
state := fbGetMasterState.state;
bError := fbGetMasterState.bError;
nErrId := fbGetMasterState.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.3 FB_EcGetSlaveState

Mit dem Funktionsbaustein `FB_EcGetSlaveState` kann der EtherCAT-Status und der Link-Status eines einzelnen EtherCAT-Slave-Gerätes ausgelesen werden. Bei erfolgreichem Aufruf enthält die Ausgangsvariable state vom Typ ST_EcSlaveState die angeforderte Status-Information.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen Status ausgelesen werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    state  : ST_EcSlaveState;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
state	ST_EcSlaveState	Struktur, die den EtherCAT-Status und den Link-Status des Slaves enthält. (Typ: ST_EcSlaveState [▶ 121])

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_GetSlaveState
VAR
    fbGetSlaveState : FB_EcGetSlaveState;
    sNetId          : T_AmsNetId := '172.16.2.131.2.1';
    bExecute        : BOOL;
    state           : ST_EcSlaveState;
    nSlaveAddr      : UINT := 1001;
    bError          : BOOL;
    nErrId          : UDINT;
END_VAR

fbGetSlaveState(sNetId:= sNetId, nSlaveAddr:= nSlaveAddr, bExecute:=bExecute);
state := fbGetSlaveState.state;
bError := fbGetSlaveState.bError;
nErrId := fbGetSlaveState.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.4 FB_EcReqMasterState

Mit dem Funktionsbaustein kann der EtherCAT-Zustand eines Master-Geräts angefordert und gesetzt werden. Der geforderte EtherCAT-Zustand wird in der Variablen state übergeben. Sobald der Funktionsbaustein den EtherCAT-Zustand angefordert hat, wird er inaktiv. Im Gegensatz zum Funktionsbaustein FB_EcSetMasterState wartet er nicht, bis der neue Zustand gesetzt ist.

Siehe auch: FB_EcSetMasterState [▶ 47]

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
state	WORD	EtherCAT-Zustand, der vom Master angefordert wird. (Siehe state)

State

EtherCAT-Zustand, der vom Master angefordert wird. Die möglichen Werte für state sind:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Init-Zustand vom Master anfordern
EC_DEVICE_STATE_PREOP	0x02	Pre-Operational-Zustand vom Master anfordern
EC_DEVICE_STATE_SAFEOP	0x04	Safe-Operational-Zustand vom Master anfordern
EC_DEVICE_STATE_OP	0x08	Operational-Zustand vom Master anfordern

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_ReqMasterState
VAR
    fbReqMasterState : FB_EcReqMasterState;
    sNetId           : T_AmsNetId:= '172.16.2.131.2.1';
    bExecute         : BOOL;
    state            : WORD :=EC_DEVICE_STATE_INIT;
    bError           : BOOL;
```

```
    nErrId          : UDINT;
END_VAR

fbReqMasterState(sNetId:= sNetId, bExecute:=bExecute, state:=state);
bError := fbGetMasterState.bError;
nErrId := fbGetMasterState.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.5 FB_EcReqSlaveState

Mit dem Funktionsbaustein kann ein Slave in einen vorgegebenen EtherCAT-Zustand gesetzt werden. Der geforderte EtherCAT-Zustand wird in der Variablen state übergeben. Sobald der Funktionsbaustein den Befehl zum Zustandswechsel abgeschickt hat, wird er inaktiv. Im Unterschied zum Funktionsbaustein FB_EcSetSlaveState wartet er nicht, bis der EtherCAT-Slave den neuen Zustand erreicht hat.

Siehe auch: FB_EcSetSlaveState [▶ 48]

Eingänge

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nSlaveAddr  : UINT;
    bExecute    : BOOL;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
    state       : WORD;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen EtherCAT-Zustand gesetzt werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
state	WORD	EtherCAT-Zustand, der vom Master angefordert wird. (Siehe State)

State

EtherCAT-Zustand, in den der Slave gesetzt werden soll. Die möglichen Werte für state sind:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Slave in den Init-Zustand setzen
EC_DEVICE_STATE_PREOP	0x02	Slave in den Pre-Operational-Zustand setzen
EC_DEVICE_STATE_BOOTSTRAP	0x03	Slave in Bootstrap-Zustand setzen, dieser Zustand wird verwendet um ein Firmware-Download durchzuführen.
EC_DEVICE_STATE_SAFEOP	0x04	Slave in den Safe-Operational-Zustand setzen
EC_DEVICE_STATE_OP	0x08	Slave in den Operational-Zustand setzen
EC_DEVICE_STATE_ERROR	0x10	Wenn bei dem EtherCAT-Slave das Error-Bit im Status-Byte gesetzt ist (state.deviceState & EC_DEVICE_STATE_ERROR = TRUE), kann das Error-Bit durch Setzen von EC_DEVICE_STATE_ERROR wieder zurückgesetzt werden.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_ReqSlaveState
VAR
    fbGetSlaveState : FB_EcReqSlaveState;
    sNetId          : T_AmsNetId:= '172.16.2.131.2.1';
    bExecute        : BOOL;
    state           : WORD := EC_DEVICE_STATE_INIT;
    nSlaveAddr      : UINT := 1001;
    bError          : BOOL;
    nErrId          : UDINT;
END_VAR

fbGetSlaveState(sNetId:= sNetId, nSlaveAddr:= nSlaveAddr, bExecute:=bExecute, state:=state);
bError := fbGetSlaveState.bError;
nErrId := fbGetSlaveState.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.6 FB_EcSetMasterState

Mit dem Funktionsbaustein kann der EtherCAT-Zustand eines Master-Geräts angefordert und gesetzt werden. Der geforderte EtherCAT-Zustand wird mit der Variablen reqState übergeben. Der Funktionsbaustein fordert den EtherCAT-Zustand an und bleibt im Gegensatz zum Funktionsbaustein FB_EcReqMasterState aktiv, bis der neue Zustand gesetzt oder die maximale tTimeout-Zeit überschritten wird. Der aktuelle Zustand wird in der Variablen currState ausgegeben.

Siehe auch: FB_EcReqMasterState [▶ 43]

Eingänge

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    bExecute : BOOL;
    tTimeout : TIME := T#10s;
    reqState : WORD;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
reqState	WORD	(Siehe reqState)

reqState

EtherCAT-Zustand, der vom Master angefordert wird. Die möglichen Werte für reqState sind:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Init-Zustand vom Master anfordern
EC_DEVICE_STATE_PREOP	0x02	Pre-Operational-Zustand vom Master anfordern
EC_DEVICE_STATE_SAFEOP	0x04	Safe-Operational-Zustand vom Master anfordern
EC_DEVICE_STATE_OP	0x08	Operational-Zustand vom Master anfordern

Ausgänge

```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
    nErrId   : UDINT;
    currState : WORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
currState	WORD	Aktueller EtherCAT-Zustand des Masters

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

5.7 FB_EcSetSlaveState

Mit dem Funktionsbaustein kann ein Slave in einen vorgegebenen EtherCAT-Zustand gesetzt werden. Der geforderte EtherCAT-Zustand wird in der Variablen reqState übergeben. Der Funktionsbaustein schickt den Befehl zum Zustandswechsel ab und bleibt im Unterschied zum Funktionsbaustein FB_EcRegSlaveState so lange aktiv, bis der EtherCAT-Slave den neuen Zustand erreicht hat oder die maximale tTimeout-Zeit überschritten wird. Der aktuelle Zustand wird in der Variablen currState ausgegeben.

Siehe auch: FB_EcReqSlaveState [▶ 45]

Eingänge

```
VAR_INPUT
    sNetId     : T_AmsNetId;
    nSlaveAddr : UINT;
    bExecute   : BOOL;
    tTimeout   : TIME := T#10s;
    reqState   : WORD;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkerkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen EtherCAT-Zustand gesetzt werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
reqState	WORD	EtherCAT-Zustand, in den der Slave gesetzt werden soll. (Siehe reqState)

reqState

EtherCAT-Zustand, in den der Slave gesetzt werden soll. Die möglichen Werte für reqState sind:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Slave in den Init-Zustand setzen
EC_DEVICE_STATE_PREOP	0x02	Slave in den Pre-Operational-Zustand setzen
EC_DEVICE_STATE_BOOTSTRAP	0x03	Slave in Bootstrap-Zustand setzen. Dieser Zustand wird verwendet, um ein Firmware-Download durchzuführen.
EC_DEVICE_STATE_SAFEOP	0x04	Slave in den Safe-Operational-Zustand setzen
EC_DEVICE_STATE_OP	0x08	Slave in den Operational-Zustand setzen
EC_DEVICE_STATE_ERROR	0x10	Wenn bei dem EtherCAT-Slave das Error-Bit im Status-Byte gesetzt ist (currState.deviceState AND EC_DEVICE_STATE_ERROR = TRUE), kann das Error-Bit durch Setzen von EC_DEVICE_STATE_ERROR wieder zurückgesetzt werden.

Ausgänge

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    currState : ST_EcSlaveState;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
currState	ST_EcSlaveState	Aktueller EtherCAT-Zustand [▶ 121] des Slaves

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6 CoE Interface

6.1 FB_EcCoeSdoRead

Mit dem Funktionsbaustein `FB_EcCoeSdoRead` können per SDO(Service Daten Objekt)-Zugriff Daten aus dem Objektverzeichnis eines EtherCAT-Slaves ausgelesen werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt ausgelesen werden soll. Für den Zugriff auf den kompletten Parameter mit Unterelementen muss der Baustein FB_EcCoeSdoReadEx [▶ 51] verwendet werden.

Eingänge

```
VAR_INPUT
    sNetId     : T_AmsNetId;
    nSlaveAddr : UINT;
    nSubIndex  : BYTE;
    nIndex     : WORD;
    pDstBuf    : PVOID;
    cbBufLen   : UDINT;
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Upload-Kommando geschickt werden soll.
nSubIndex	BYTE	Subindex des Objekts, das gelesen werden soll.
nIndex	WORD	Index des Objekts, das gelesen werden soll.
pDstBuf	PVOID	Adresse (Pointer) auf den Empfangspuffer.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Beispiel für eine Implementierung in ST

```
PROGRAM TEST_SdoRead
VAR
    fbSdoRead  : FB_EcCoESdoRead;
    sNetId     : T_AmsNetId := '172.16.2.131.2.1';
    bExecute   : BOOL;
    nSlaveAddr : UINT := 1006;
    nIndex     : WORD := 16#1018;
    nSubIndex  : BYTE :=1;
    vendorId   : UDINT;
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR

fbSdoRead(sNetId:= sNetId,nSlaveAddr :=nSlaveAddr, nIndex:=nIndex, nSubIndex :=nSubIndex, pDstBuf:=
ADR(vendorId), cbBufLen:=SIZEOF(vendorId),bExecute:=bExecute);
bError:=fbSdoRead.bError;
nErrId:=fbSdoRead.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.2 FB_EcCoeSdoReadEx

Mit dem Funktionsbaustein `FB_EcCoeSdoReadEx` können per SDO (Service Daten Objekt)-Zugriff Daten aus dem Objektverzeichnis eines EtherCAT-Slaves ausgelesen werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt ausgelesen werden soll. Über `bCompleteAccess := TRUE` kann der Parameter mit Unterelementen eingelesen werden.

Eingänge

```
VAR_INPUT
    sNetId          : T_AmsNetId; (* AmsNetId of the EtherCAT master device.*)
    nSlaveAddr      : UINT; (* Address of the slave device.*)
    nSubIndex       : BYTE; (* CANopen Sdo subindex.*)
    nIndex          : WORD; (* CANopen Sdo index.*)
    pDstBuf         : PVOID; (* Contains the address of the buffer for the received data. *)
    cbBufLen        : UDINT; (* Contains the max. number of bytes to be received. *)
    bExecute        : BOOL; (* Function block execution is triggered by a rising edge at this input.
*)
    tTimeout        : TIME := DEFAULT_ADS_TIMEOUT;
```

```
(* States the time before the function is cancelled. *)
    bCompleteAccess : BOOL; (* access complete object*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_Ams NetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlave Addr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Upload-Kommando geschickt werden soll.
nSubIndex	BYTE	Subindex des Objekts, das gelesen werden soll.
nIndex	WORD	Index des Objekts, das gelesen werden soll.
pDstBuf	PVOID	Adresse (Pointer) auf den Empfangspuffer
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
bComplete Access	BOOL	Bei gesetztem bCompleteAccess kann der gesamte Parameter in einem Zugriff eingelesen werden.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.3 FB_EcCoeSdoWrite

Mit dem Funktionsbaustein `FB_EcCoeSdoWrite` kann per SDO-Download ein Objekt aus dem Objektverzeichnis eines EtherCAT-Slaves beschrieben werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt beschrieben werden soll. Für den Zugriff auf den kompletten Parameter mit Unterelementen muss der Baustein FB_EcCoeSdoWriteEx [▶ 54] verwendet werden.

Eingänge

```
VAR_INPUT
    sNetId     : T_AmsNetId;
    nSlaveAddr : UINT;
    nSubIndex  : BYTE;
    nIndex     : WORD;
    pSrcBuf    : PVOID;
    cbBufLen   : UDINT;
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Download-Kommando geschickt werden soll.
nSubIndex	BYTE	Subindex des Objekts, das geschrieben werden soll.
nIndex	WORD	Index des Objekts, das geschrieben werden soll.
pSrcBuf	PVOID	Adresse (Pointer) auf den Sendepuffer
cbBufLen	UDINT	Anzahl der zu sendenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Beispiel für eine Implementierung in ST:

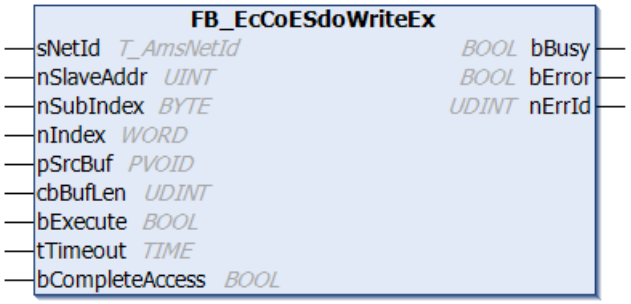
```
PROGRAM TEST_SdoWrite

VAR
    fbSdoWrite : FB_EcCoESdoWrite;
    sNetId     : T_AmsNetId := '172.16.2.131.2.1'; (* NetId of EtherCAT Master *)
    nSlaveAddr : UINT := 1005; (* Port Number of EtherCAT Slave *)
    nIndex     : WORD := 16#4062; (* CoE Object Index *)
    nSubIndex  : BYTE := 1; (* Subindex of CoE Object *)
    nValue     : UINT := 2; (* variable to be written to the CoE Object *)
    bExecute   : BOOL; (* rising edge starts writing to the CoE Object *)
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR


fbSdoWrite(
    sNetId     := sNetId,
    nSlaveAddr := nSlaveAddr,
    nIndex     := nIndex,
    nSubIndex  := nSubIndex,
    pSrcBuf    := ADR(nValue),
    cbBufLen   := SIZEOF(nValue),
    bExecute   := bExecute
);
```

```
IF NOT fbSdoWrite.bBusy THEN
    bExecute := FALSE;
    IF NOT fbSdoWrite.bError THEN
        (* write successful *)
        bError := FALSE;
        nErrId := 0;
    ELSE
        (* write failed *)
        bError := fbSdoWrite.bError;
        nErrId := fbSdoWrite.nErrId;
    END_IF

    fbSdoWrite(bExecute := FALSE);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.4 FB_EcCoeSdoWriteEx



Mit dem Funktionsbaustein `FB_EcCoeSdoWriteEx` kann per SDO-Download ein Objekt aus dem Objektverzeichnis eines EtherCAT Slaves beschrieben werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt beschrieben werden soll. Über `bCompleteAccess := TRUE` kann der Parameter mit Unterelementen geschrieben werden.

Eingänge

```
VAR_INPUT
    sNetId          : T_AmsNetId; (* AmsNetId of the EtherCAT master device.*)
    nSlaveAddr      : UINT; (* Address of the slave device.*)
    nSubIndex       : BYTE; (* CANopen Sdo subindex.*)
    nIndex          : WORD; (* CANopen Sdo index.*)
    pSrcBuf         : PVOID; (* Contains the address of the buffer containing the data to be send. *
)
    cbBufLen        : UDINT; (* Contains the max. number of bytes to be received. *)
    bExecute        : BOOL; (* Function block execution is triggered by a rising edge at this input.
*)
    tTimeout        : TIME := DEFAULT_ADS_TIMEOUT;
(* States the time before the function is cancelled. *)
    bCompleteAccess : BOOL; (* access complete object*)
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_Ams NetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlave Addr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Download-Kommando geschickt werden soll.
nSubIndex	BYTE	Subindex des Objekts, das geschrieben werden soll.
nIndex	WORD	Index des Objekts, das geschrieben werden soll.
pSrcBuf	PVOID	Adresse (Pointer) auf den Sendepuffer
cbBufLen	UDINT	Anzahl der zu sendenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
bCompleteAccess	BOOL	Bei gesetztem bCompleteAccess kann der gesamte Parameter in einem Zugriff geschrieben werden.

Ausgänge

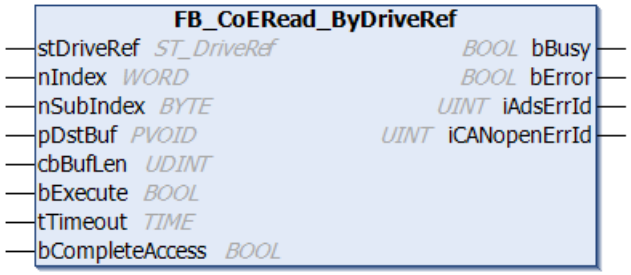
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.5 FB_CoERead_ByDriveRef



Mit dem Funktionsbaustein `FB_CoERead_ByDriveRef` können Antriebsparameter mit Hilfe des "CANopen over EtherCAT (CoE)"-Protokolls gelesen werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt ausgelesen werden soll. Über `bCompleteAccess := TRUE` kann der Parameter mit Unterelementen gelesen werden.

Eingänge

```
VAR_INPUT
    stDriveRef       :  ST_DriveRef; (*Contains sNetID of EcMaster, nSlaveAddr of EcDrive, nDriveNo
of EcDrive, either preset or read from NC*)
    nIndex           :  WORD; (*SoE IDN: e.g. "S_0_IDN+1" for S-0-0001 or "P_0_IDN+23" for
```

```
P-0-0023*)
    nSubIndex        :  BYTE;
    pDstBuf          :  PVOID; (*Contains the address of the buffer for the received data*)
    cbBufLen         :  UDINT; (*Contains the max. number of bytes to be received*)
    bExecute         :  BOOL; (*Function block execution is triggered by a rising edge at this
input*)
    tTimeout         :  TIME; (*States the time before the function is cancelled*)
    bCompleteAccess  :  BOOL;
END_VAR
```

Name	Typ	Beschreibung
stDriveRef	ST_DriveRef	Struktur, welche die AMS- Netzwerkkennung des EtherCAT-Master-Gerätes und die Adresse des Slave-Gerätes enthält. Die Referenz auf den Antrieb kann im System Manager direkt in die SPS gelinkt werden. Hierzu muss eine Instanz der ST_PlcDriveRef verwendet werden und die NEtID vom Bytearray in einem String konvertiert werden.
nIndex	WORD	Index des Objekts, das gelesen werden soll.
nSubIndex	BYTE	Subindex des Objekts, das gelesen werden soll.
pDstBuf	PVOID	Adresse (Pointer) auf den Empfangspuffer.
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
bCompleteAccess	BOOL	Bei gesetztem bCompleteAccess kann der gesamte Parameter in einem Zugriff eingelesen werden.

Ausgänge

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

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
iAdsErrId	UINT	Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
iCANopenErrId	UINT	Liefert bei gesetztem bError-Ausgang den CANopen-Fehlercode.

Beispiel für eine Implementierung in ST:

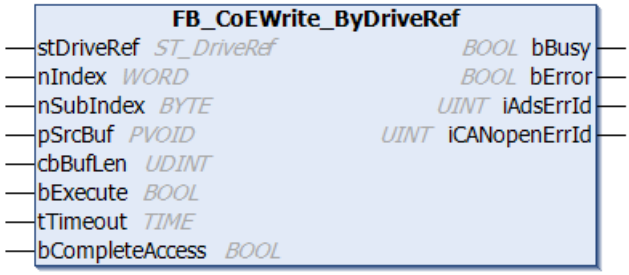
```
PROGRAM MAIN
VAR
    fbCoERead       : FB_CoERead_ByDriveRef;
    stDriveRef      : ST_DriveRef;
    nIndex          : WORD := 16#1018;
    nSubIndex       : BYTE := 1;
    bExecute        : BOOL := TRUE;
    tTimeout        : TIME := T#5S;
    bCompleteAccess : BOOL := TRUE;
    vendorId        : UDINT;
    bError          : BOOL;
    nAdsErrId       : UDINT;
    nCANopenErrId   : UDINT;
END_VAR

fbCoERead(
    stDriveRef:= stDriveRef,
    nIndex:= nIndex,
    nSubIndex:= nSubIndex,
    pDstBuf:= ADR(vendorId),
    cbBufLen:= SIZEOF(vendorId),
    bExecute:= bExecute,
```

```
    tTimeout:= tTimeout,
    bCompleteAccess:= bCompleteAccess,
);

IF NOT fbCoERead.bBusy THEN
    bError:=fbCoERead.bError;
    nAdsErrId:=fbCoERead.iAdsErrId;
    nCANopenErrId:=fbCoERead.iCANopenErrId;
    bExecute := FALSE;
    fbCoERead(bExecute := bExecute);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.6 FB_CoEWrite_ByDriveRef



Mit dem Funktionsbaustein `FB_CoEWrite_ByDriveRef` können Antriebsparameter mit Hilfe des "CANopen over EtherCAT (CoE)"-Protokolls geschrieben werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen. Mit Hilfe der Parameter nSubIndex und nIndex wird ausgewählt, welches Objekt beschrieben werden soll. Über `bCompleteAccess := TRUE` kann der Parameter mit Unterelementen geschrieben werden.

Eingänge

```
VAR_INPUT
    stDriveRef       :  ST_DriveRef; (*Contains sNetID of EcMaster, nSlaveAddr EcDrive, nDriveNo of
EcDrive, either preset or read from NC*)
    nIndex           :  WORD; (*SoE IDN: e.g. "S_0_IDN+1" for S-0-0001 or "P_0_IDN+23" for
P-0-0023*)
    nSubIndex        :  BYTE; (*SoE element*)
    pSrcBuf          :  PVOID; (*Contains the address of the buffer containing the data to be sent*)
    cbBufLen         :  UDINT; (*Contains the max. number of bytes to be received*)
    bExecute         :  BOOL; (*Function block execution is triggered by a rising edge at this
input*)
    tTimeout         :  TIME; (*States the time before the function is cancelled*)
    bCompleteAccess  :  BOOL;
END_VAR
```

Name	Typ	Beschreibung
stDriveRef	ST_DriveRef	Struktur, welche die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes und die Adresse des Slave-Gerätes enthält. Die Referenz auf den Antrieb kann im System Manager direkt in die SPS gelinkt werden. Hierzu muss eine Instanz der ST_PlcDriveRef verwendet werden und die NEtID vom Bytearray in einem String konvertiert werden.
nIndex	WORD	Index des Objekts, das geschrieben werden soll.
nSubIndex	BYTE	Subindex des Objekts, das geschrieben werden soll.
pSrcBuf	PVOID	Adresse (Pointer) auf den Sendepuffer
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu sendenden Daten in Bytes.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.
bCompleteAccess	BOOL	Bei gesetztem bCompleteAccess kann der gesamte Parameter in einem Zugriff eingelesen werden.

Ausgänge

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

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
iAdsErrId	UINT	Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
iCANopenErrId	UINT	Liefert bei gesetztem bError-Ausgang den CANopen-Fehlercode.

Beispiel für eine Implementierung in ST:

```
PROGRAM MAIN
VAR
    fbCoEWrite      : FB_CoEWrite_ByDriveRef;
    stDriveRef      : ST_DriveRef;
    nIndex          : WORD := 16#1018;
    nSubIndex       : BYTE := 1;
    bExecute        : BOOL := TRUE;
    tTimeout        : TIME := T#5S;
    bCompleteAccess : BOOL := TRUE;
    vendorId        : UDINT := 2;
    bError          : BOOL;
    nAdsErrId       : UDINT;
    nCANopenErrId   : UDINT;
END_VAR

fbCoEWrite(
    stDriveRef:= stDriveRef,
    nIndex:= nIndex,
    nSubIndex:= nSubIndex,
    pSrcBuf:= ADR(vendorId),
    cbBufLen:= SIZEOF(vendorId),
    bExecute:= bExecute,
    tTimeout:= tTimeout,
    bCompleteAccess:= bCompleteAccess,
);

IF NOT fbCoEWrite.bBusy THEN
    bError:= fbCoEWrite.bError;
    nAdsErrId:= fbCoEWrite.iAdsErrId;
    nCANopenErrId:= fbCoEWrite.iCANopenErrId;
```

```
    bExecute := FALSE;
    fbCoEWrite(bExecute := bExecute);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.7 FB_EcCoeReadBIC

Mit dem Funktionsbaustein FB_EcCoeReadBIC kann per SDO(Service Daten Objekt)-Zugriff die BIC aus dem Objektverzeichnis eines EtherCAT-Slaves ausgelesen werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen und das Objektverzeichnis muss ein Objekt 0x10E2:01 mit der BIC enthalten.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Upload-Kommando geschickt werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    sBICValue : STRING
    stMSID    : ST_SplittedBIC
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
sBICValue	STRING(1023)	Dieser Ausgang enthält bei fehlerfreiem Durchlauf die BIC des EtherCAT Slaves, nachdem der bBusy-Ausgang zurückgesetzt wurde, z.B. „1P193995SBTN0002agdw1KEL7411 Q1 2P112104020018“.
stMSID	ST_SplittedBIC	Dieser Ausgang enthält bei fehlerfreiem Durchlauf die Teilstrings der BIC des EtherCAT Slaves, nachdem der bBusy-Ausgang zurückgesetzt wurde. Für die oben angeführte BIC sind folgende Teilstrings belegt: sItemNo = „193995“ sBTN = „0002agdw“ sDescription = „EL7411“ sQuantity = „1“ sBatchNo = „112104020018“

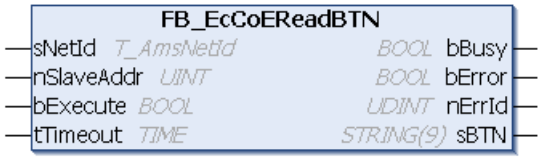
Beispiel für eine Implementierung in ST

```
PROGRAM TEST_EcCoEReadBIC
VAR
    fbCoEBIC   : FB_EcCoEReadBIC;
    sNetId     : T_AmsNetId := '172.16.2.131.2.1';
    bExecute   : BOOL := TRUE;
    nSlaveAddr : UINT := 1006;
    sCoEBIC    : STRING(1023);
    stCoEBIC   : ST_SplittedBIC;
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR

fbCoEBIC(sNetId:= sNetID, nSlaveAddr:= nPort, bExecute:= bExecute, tTimeout:= T#5s);;
IF NOT fbCoEBIC.bBusy THEN
     bExecute := FALSE;
     IF NOT fbCoEBIC.bError THEN
          stCoEBIC   := fbCoEBIC.stMSID;
          sCoEBIC    := fbCoEBIC.sBICValue;
     END_IF
     fbCoEBIC(bExecute:= bExecute);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

6.8 FB_EcCoeReadBTN



Mit dem Funktionsbaustein FB_EcCoeReadBTN kann per SDO(Service Daten Objekt)-Zugriff die BTN aus dem Objektverzeichnis eines EtherCAT-Slaves ausgelesen werden. Dazu muss der Slave eine Mailbox besitzen und das "CANopen over EtherCAT (CoE)"-Protokoll unterstützen unterstützen und das Objektverzeichnis muss ein Objekt 0xF083 mit der BTN enthalten.

Eingänge

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

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SDO-Upload-Kommando geschickt werden soll.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    sBTN   : STRING(9)
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
sBTN	STRING	Dieser Ausgang enthält bei fehlerfreiem Durchlauf die BTN des EtherCAT Slaves, nachdem der bBusy-Ausgang zurückgesetzt wurde, z.B. „0002agdw“.

Beispiel für eine Implementierung in ST

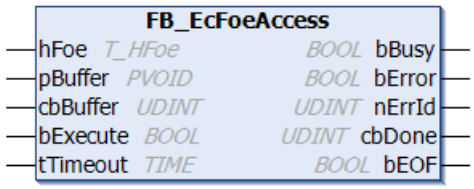
```
PROGRAM TEST_ EcCoEReadBtn
VAR
    fbCoEBTN   : FB_EcCoEReadBtn;
    sNetId     : T_AmsNetId := '172.16.2.131.2.1';
    bExecute   : BOOL := TRUE;
    nSlaveAddr : UINT := 1006;
    sCoEBTN    : STRING;
    bError     : BOOL;
    nErrId     : UDINT;
END_VAR

fbCoEBTN(sNetId:= sNetID, nSlaveAddr:= nPort, bExecute:= bExecute, tTimeout:= T#5S);
IF NOT fbCoEBTN.bBusy THEN
     bExecute := FALSE;
     IF NOT fbCoEBTN.bError THEN
          sCoEBTN := fbCoEBTN.sBTN;
     END_IF
     fbCoEBTN(bExecute:= bExecute);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

7 FoE Interface

7.1 FB_EcFoeAccess



Dieser Funktionsbaustein schreibt oder liest Daten über den Kommunikationsport des "File access over EtherCAT"-Mailbox-Protokolls.

Eingänge

```
VAR_INPUT
    hFoe     : T_HFoe;
    pBuffer  : DWORD;
    cbBuffer : UDINT;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
hFoe	T_HFoe [▶ 127]	"File access over EtherCAT"-Handle
pBuffer	DWORD	Enthält die Adresse des Puffers, in den die Daten gelesen werden sollen (Lesezugriff) oder die Adresse des Puffers, der die zu schreibenden Daten enthält (Schreibzugriff). Der Puffer kann eine Einzelvariable, ein Array oder eine Struktur sein, dessen Adresse mit dem ADR-Operator ermittelt werden kann.
cbBuffer	UDINT	Enthält die Anzahl der zu schreibenden oder zu lesenden Datenbytes.
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

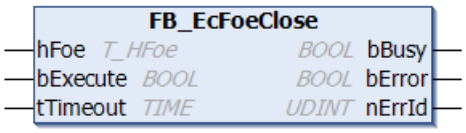
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    cbDone : UDINT;
    bEOF   : BOOL;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn ein Fehler bei der Übertragung des Kommandos auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
cbDone	UDINT	Anzahl der zuletzt erfolgreich geschriebenen oder gelesenen Datenbytes
bEOF	BOOL	End of File, diese Variable wird TRUE, wenn beim Lesezugriff das Ende der Datei erreicht wurde. Beim Schreibzugriff hat diese Variable keine Bedeutung.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

7.2 FB_EcFoeClose



Dieser Funktionsbaustein schließt den Kommunikationsport für das "File access over EtherCAT"-Mailbox-Protokoll.

Eingänge

```
VAR_INPUT
    hFoe     : T_HFoe;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
hFoe	T_HFoe [▶ 127]	"File access over EtherCAT"-Handle
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

Ausgänge

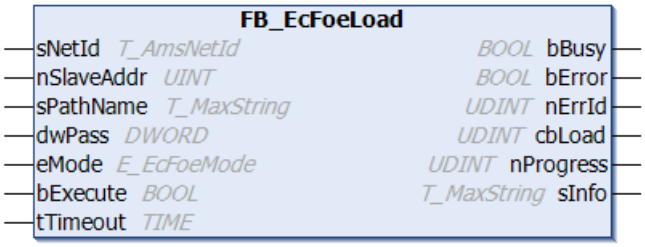
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

7.3 FB_EcFoeLoad



Mit dem Funktionsbaustein `FB_EcFoeLoad` können Dateien über das "File access over EtherCAT"-Mailbox-Protokoll zu einem EtherCAT-Gerät heruntergeladen oder von einem EtherCAT-Gerät hochgeladen werden.

> Der Dateipfad kann nur auf das lokale Dateisystem des Rechners zeigen. Das bedeutet, Netzwerkpfade können hier nicht angegeben werden. Um Dateien über das FoE-Protokoll hoch- oder herunterladen zu können, setzt der Funktionsbaustein das EtherCAT-Gerät automatisch in den BOOTSTRAP-Mode zurück. Zum Schluss versucht der Funktionsbaustein das Gerät wieder in den ursprünglichen Zustand zu versetzen.

Eingänge

```
VAR_INPUT
    sNetId     : T_AmsNetId ;
    nSlaveAddr : UINT;
    sPathName  : T_MaxString;
    dwPass     : DWORD := 0;
    eMode      : E_EcFoeMode := eFoeMode_Write;
    bExecute   : BOOL;
    tTimeout   : TIME := T#200s;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält.
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, dessen Datei hoch- oder heruntergeladen werden soll.
sPathName	T_MaxString	Enthält den Pfad- und Dateinamen der zu schreibenden oder zu lesenden Datei. (z.B.: 'C:\FOE_Test\EL6751\ECATFW__EL6751_C6_V0030.efw')
dwPass	DWORD	Passwort (Default: 0)
eMode	E_EcFoeMode [▶ 116]	"File access over EtherCAT"-Zugriffsmode (Default: Schreibzugriff)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf (Default: 200s.).

Ausgänge

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    cbLoad    : UDINT;
    nProgress : UDINT;
    sInfo     : T_MaxString;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
cbLoad	UDINT	Anzahl der erfolgreich geschriebenen oder gelesenen Datenbytes
nProgress	UDINT	Prozentualer Fortschritt beim Schreibzugriff (Bereich: 0 - 100%). Beim Lesezugriff wird diese Variable zurzeit nicht benutzt und ist immer 0.
sInfo	T_MaxString	Zusätzliche Befehlsinformation als String (reserviert)

Beispiel in ST:

Bei einer steigenden Flanke an der bLoad-Variablen wird der Firmware-Download über das "File access over EtherCAT"-Mailbox-Protokoll gestartet.

```
PROGRAM MAIN
VAR
    fbDownload : FB_EcFoeLoad := (
        sNetID     := '5.0.34.38.3.1',
        nSlaveAddr := 1004,
        sPathName  := 'C:\FOE_Test\EL6751\ECATFW__EL6751_C6_V0030.efw',
        dwPass     := 0,
        eMode      := eFoeMode_Write );
    bLoad      : BOOL;
    bBusy      : BOOL;
    bError     : BOOL;
    nErrID     : UDINT;
    nBytesWritten : UDINT;
    nPercent      : UDINT;
END_VAR

fbDownload( bExecute := bLoad,
        bBusy => bBusy,
        bError => bError,
        nErrId => nErrID,
        cbLoad => nBytesWritten,
        nProgress => nPercent );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

7.4 FB_EcFoeOpen

Mit diesem Funktionsbaustein wird der Kommunikationsport für das "File access over EtherCAT"-Mailbox-Protokoll geöffnet.

Eingänge

```
VAR_INPUT
    sNetId    : T_AmsNetId;
    nPort     : UINT;
    sPathName : T_MaxString;
    dwPass    : DWORD;
    eMode     : E_EcFoeMode;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält.
nPort	UINT	Feste Adresse des EtherCAT-Gerätes
sPathName	T_MaxString	Dateipfadname (z.B.: 'c:\TwinCAT\FOE\Data.fwp') (Siehe unten weitere Erläuterungen zum sPathName.)
dwPass	DWORD	Passwort
eMode	E_EcFoeMode [▶ 116]	Zugriffsmode (Schreib-/Lesezugriff)
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

sPathName

Als Dateipfadname (z.B.: 'c:\TwinCAT\FOE\Data.fwp') aus dem eingegebenen Dateipfad wird standardmäßig nur der Dateiname, ohne Dateinamenserweiterung extrahiert und als Dateiname für das FoE-Protokoll verwendet (in diesem Beispiel: ‚Data'). Ab der Bibliotheksversion >= 3.3.12.0 können auch Dateinamen inklusive der Dateinamenserweiterung verwendet werden (in diesem Beispiel: ‚Data.fwp').

Über die globale boolische Variable
`Tc2_EtherCAT.bEcFoeOpenFileNameWithFileExt`
kann die Verwendung der Dateinamenserweiterung für alle Instanzen des FB_EcFoeOpen-Funktionsbausteins aktiviert bzw. deaktiviert werden. Standardmäßig hat die Variable den Wert: FALSE (keine Dateinamenserweiterung). Wenn Sie den Wert auf TRUE setzen, wird die Verwendung der Dateinamenserweiterungen aktiviert.
Beachten Sie, dass die FoE-Funktionsbausteine ursprünglich für Firmwareupdates verwendet wurden, bei denen keine Dateinamenserweiterung verwendet wurde. Sollten Sie Firmwareupdates durchführen wollen, müssen Sie eventuell sicherstellen, dass die globale Variable ihren ursprünglichen Standardwert FALSE besitzt.

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
    hFoe   : T_HFoe;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
hFoe	T_HFoe [▶ 127]	"File access over EtherCAT" - Handle

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

7.5 FB_EcFoeReadFile

Mit dem Funktionsbaustein FB_EcFoeReadFile können Dateien über das "File access over EtherCAT"-Mailbox-Protokoll von einem EtherCAT-Gerät auf den lokalen Datenträger heruntergeladen werden.

Der Dateipfad kann nur auf das lokale Dateisystem des Rechners zeigen. Das bedeutet, Netzwerkpfade können hier nicht angegeben werden.

Eingänge

```
VAR_INPUT
    sFSrvNetId      : T_AmsNetId := '';
    sFSrvPathName   : T_MaxString;
    sEcNetId        : T_AmsNetId;
    nSlaveAddr      : UINT;
    sFoEPathName    : T_MaxString;
    dwPass          : DWORD := 0;
    bExecute        : BOOL;
    tTimeout        : TIME := T#200s;
END_VAR
```

Name	Typ	Beschreibung
sFSrvNetId	T_AmsNetId	AMS-Netzwerkkennung des Rechners, auf den die gelesene Datei geschrieben werden soll. (Default: lokaler Rechner)
sFSrvPathName	T_MaxString	Enthält den Pfad- und Dateinamen der zu schreibenden Datei (z. B.: 'C:\Data\LogData.csv').
sEcNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält.
nSlaveAddr	UINT	Adresse des EtherCAT-Slaves
sFoEPathName	T_MaxString	Name der Datei auf dem EtherCAT-Slave (z. B. 'LogData')
dwPass	DWORD	Passwort
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf. (Default: 200s.)

Ausgänge

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    cbRead      : UDINT;
    sInfo       : T_MaxString;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
cbRead	UDINT	Anzahl der erfolgreich gelesenen Datenbytes
sInfo	T_MaxString	Zusätzliche FoE-Fehlerinformation (reserviert)

Beispiel in ST:

Bei einer steigenden Flanke an der `bExecute`-Variablen wird das Lesen der angegebenen Datei über das "File access over EtherCAT"-Mailbox-Protokoll gestartet. Es wird die in `sFoEPathName` benannte Datei vom gewählten EtherCAT-Slave (`sEcNetId & nSlaveAddr`) gelesen und die Datei wird auf dem gewählten Rechner (`sFSrvNetID`) unter dem in `sFSrvPathName` genannten Namen abgelegt. Falls für das Lesen der Datei vom EtherCAT-Slave ein Passwort erforderlich ist, kann dieses über `dwPass` angegeben werden.

Der Lese- und Schreibvorgang ist erst beendet, wenn `bBusy` auf FALSE geht. Erst dann können die Fehlerinformationen bzw. die Anzahl der gelesenen Bytes ausgewertet werden.

```
PROGRAM MAIN
VAR
    fbEcReadFile : FB_EcFoeReadFile := (
        sFSrvNetID     := '5.0.34.38.1.1', (* NetID for target file *)
        sFSrvPathName  := 'C:\Data\LogData.csv', (* Pathname for target file *)
        sEcNetId       := '5.0.34.38.3.1', (* NetID of EtherCAT master *)
        nSlaveAddr     := 1004, (* EtherCAT slave address *)
        sFoEPathName   := 'LogData', (* Name of source file *)
        dwPass         := 0
    );
    bExecute   : BOOL := TRUE;
    bBusy      : BOOL;
    bError     : BOOL;
    nErrID     : UDINT;
    nBytesRead : UDINT;
END_VAR

fbEcReadFile (
        bExecute := bExecute,
        bBusy => bBusy,
        bError => bError,
        nErrId => nErrID
);
IF NOT bBusy THEN
    bExecute := FALSE;

    IF NOT bError THEN
         (* done, no error *)
        nBytesRead := fbEcReadFile.cbRead;
    ELSE
        (* evaluate error *)
        nBytesRead := 0;
    END_IF
    fbEcReadFile (bExecute := FALSE);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliothe-ken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.3.14

7.6 FB_EcFoeWriteFile

Mit dem Funktionsbaustein FB_EcFoeWriteFile können Dateien über das "File access over EtherCAT"-Mailbox-Protokoll von einem lokalen Datenträger auf ein EtherCAT-Gerät geschrieben werden.

Der Dateipfad kann nur auf das lokale Dateisystem des Rechners zeigen. Das bedeutet, Netzwerkpfade können hier nicht angegeben werden.

Eingänge

```
VAR_INPUT
    sFSrvNetId     : T_AmsNetId := '';
    sFSrvPathName  : T_MaxString;
    sEcNetId       : T_AmsNetId;
    nSlaveAddr     : UINT;
    sFoEPathName   : T_MaxString;
    dwPass         : DWORD := 0;
    bExecute       : BOOL;
    tTimeout       : TIME := T#200s;
END_VAR
```

Name	Typ	Beschreibung
sFSrvNetId	T_AmsNetId	AMS-Netzwerkkennung des Rechners, von dem die zu schreibende Datei gelesen werden soll. (Default: lokaler Rechner)
sFSrvPathName	T_MaxString	Enthält den Pfad- und Dateinamen der zu lesenden Datei (z. B.: 'C:\Data\LogData.csv').
sEcNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält.
nSlaveAddr	UINT	Adresse des EtherCAT-Slaves
sFoEPathName	T_MaxString	Name der Datei auf dem EtherCAT-Slave (z. B. 'LogData')
dwPass	DWORD	Passwort
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf. (Default: 200s.)

Ausgänge

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    cbWritten : UDINT;
    nProgress : UDINT;
    sInfo     : T_MaxString;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
cbWritten	UDINT	Anzahl der erfolgreich geschriebenen Datenbytes
nProgress	UDINT	Prozentualer Fortschritt beim Schreibzugriff (Bereich: 0 - 100%).
sInfo	T_MaxString	Zusätzliche FoE-Fehlerinformation (reserviert)

Beispiel in ST:

Bei einer steigenden Flanke an der `bExecute`-Variablen wird das Schreiben der angegebenen Datei über das "File access over EtherCAT"-Mailbox-Protokoll gestartet. Es wird die in `sFSrvPathName` benannte Datei vom gewählten Rechner (`sFSrvNetID`) gelesen und die Datei wird auf dem gewählten EtherCAT-Slave (`sEcNetId & nSlaveAddr`) unter dem in `sFoEPathName` genannten Namen abgelegt. Falls für das Schreiben der Datei auf den EtherCAT-Slave ein Passwort erforderlich ist, kann dieses über `dwPass` angegeben werden.

Der Lese- und Schreibvorgang ist erst beendet, wenn `bBusy` auf FALSE geht. Erst dann können die Fehlerinformationen bzw. die Anzahl der gelesenen Bytes ausgewertet werden.

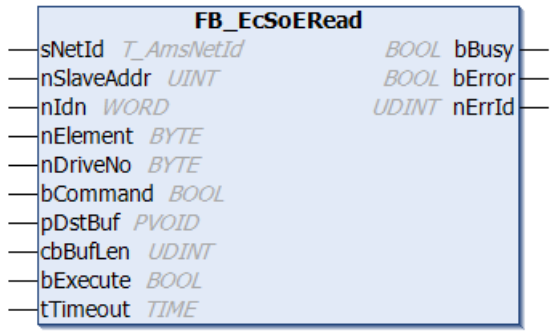
```
PROGRAM MAIN
VAR
    fbEcWriteFile : FB_EcFoeWriteFile := (
        sFSrvNetID     := '5.0.34.38.1.1', (* NetID for source file *)
        sFSrvPathName  := 'C:\Data\LogData.csv', (* Pathname for source file *)
        sEcNetId       := '5.0.34.38.3.1', (* NetID of EtherCAT master *)
        nSlaveAddr     := 1004, (* EtherCAT slave address *)
        sFoEPathName   := 'LogData', (* Name of target file *)
        dwPass         := 0
    );
    bExecute   : BOOL := TRUE;
    bBusy      : BOOL;
    bError     : BOOL;
    nErrID     : UDINT;
    nBytesWritten : UDINT;
END_VAR

fbEcWriteFile (
        bExecute := bExecute,
        bBusy => bBusy,
        bError => bError,
        nErrId => nErrID
);
IF NOT bBusy THEN
    bExecute := FALSE;

    IF NOT bError THEN
         (* done, no error *)
        nBytesWritten := fbEcWriteFile.cbWritten;
    ELSE
        (* evaluate error *)
        nBytesWritten := 0;
    END_IF
    fbEcWriteFile (bExecute := FALSE);
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.4024.56	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.5.1.0

8 SoE Interface

8.1 FB_EcSoeRead



Mit dem Funktionsbaustein `FB_EcSoeRead` können Antriebsparameter mit Hilfe des "Servo Drive Profile over EtherCAT (SoE)"-Protokolls ausgelesen werden. Dazu muss der Slave eine Mailbox besitzen und das SoE-Protokoll unterstützen. Der auszulesende Antriebsparameter wird mit den Parametern nIdn (Identification number), nElement und nDriveNo spezifiziert.

Eingänge

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nSlaveAddr  : UINT;
    nIdn        : WORD;
    nElement    : BYTE;
    nDriveNo    : BYTE;
    bCommand    : BOOL
    pDstBuf     : PVOID;
    cbBufLen    : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SoE-Read-Kommando geschickt werden soll.
nIdn	WORD	Identifikations-Nummer des zu lesenden Parameters
nElement	BYTE	Element-Nummer des zu lesenden Parameters (Siehe nElement)
nDriveNo	BYTE	Nummer des Antriebs
bCommand	BOOL	Dieser Parameter sollte gesetzt werden, wenn interne Kommando-Ausführung verwendet werden soll.
pDstBuf	PVOID	Adresse (Pointer) auf den Empfangspuffer
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Byte
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

nElement

Element-Nummer des zu lesenden Parameters. Folgende Werte sind zulässig:

Wert	Beschreibung
0x01	Data Status
0x02	Name (read only)
0x04	Attribut
0x08	Einheit
0x10	Minimum
0x20	Maximum
0x40	Wert
0x80	Default

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

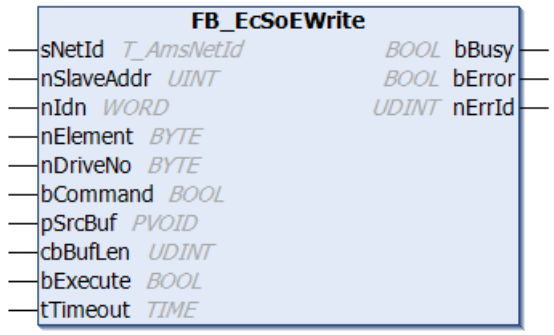
Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_SoERead
VAR
    fbSoERead   : FB_EcSoERead;
    sNetId      : T_AmsNetId:= '172.16.2.131.2.1';
    bExecute    : BOOL;
    nSlaveAddr  : UINT := 1006;
    nIdn        : WORD := 15;
    nElement    : BYTE := 0;
    nDriveNo    : BYTE := 0;
    bCommand    : BOOL := FALSE;
    val         : UINT;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR

fbSoERead(sNetId:= sNetId,nSlaveAddr :=nSlaveAddr, nIdn := nIdn, nElement:=nElement, nDriveNo := nDr
iveNo, bCommand:=bCommand, pDstBuf:= ADR(val), cbBufLen:=SIZEOF(val),bExecute:=bExecute);
bError := fbSoERead.bError;
nErrId := fbSoERead.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

8.2 FB_EcSoeWrite



Mit dem Funktionsbaustein `FB_EcSoeWrite` können Antriebs-Parameter mit Hilfe des "Servo Drive Profile over EtherCAT (SoE)"-Protokolls beschrieben werden. Dazu muss der Slave eine Mailbox besitzen und das SoE-Protokoll unterstützen. Der zu schreibende Antriebs-Parameter wird mit den Parametern nIdn (Identification number), nElement und nDriveNo spezifiziert.

Eingänge

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nSlaveAddr  : UINT;
    nIdn        : WORD;
    nElement    : BYTE;
    nDriveNo    : BYTE;
    pCommand    : BOOL;
    pSrcBuf     : PVOID;
    cbBufLen    : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

Name	Typ	Beschreibung
sNetId	T_AmsNetId	String, der die AMS-Netzwerkkennung des EtherCAT-Master-Gerätes enthält. (Typ: T_AmsNetId)
nSlaveAddr	UINT	Feste Adresse des EtherCAT-Slaves, an den das SoE-Write-Kommando geschickt werden soll.
nIdn	WORD	Identifikations-Nummer des zu schreibenden Parameters.
nElement	BYTE	Element-Nummer des zu schreibenden Parameters (Siehe nElement)
nDriveNo	BYTE	Nummer des Antriebs
bCommand	BOOL	Dieser Parameter sollte gesetzt werden, wenn interne Kommando-Ausführung verwendet werden soll.
pSrcBuf	PVOID	Adresse (Pointer) auf den Sendepuffer
cbBufLen	UDINT	Anzahl der zu sendenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

nElement

Element-Nummer des zu schreibenden Parameters. Folgende Werte sind zulässig:

Wert	Beschreibung
0x01	Data Status
0x02	Name (read only)
0x04	Attribut
0x08	Einheit
0x10	Minimum
0x20	Maximum
0x40	Wert
0x80	Default

Ausgänge

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
nErrId	UDINT	Liefert bei einem gesetzten bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.

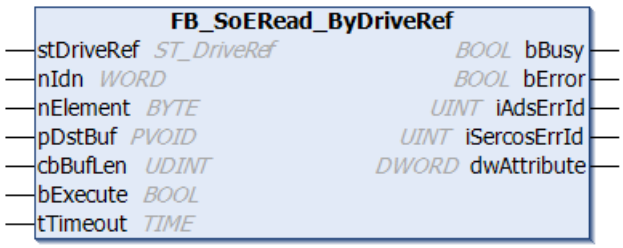
Beispiel für eine Implementierung in ST:

```
PROGRAM TEST_SoEWrite
VAR
    fbSoeWrite  : FB_EcSoEWrite;
    sNetId      : T_AmsNetId:= '172.16.2.131.2.1';
    bExecute    : BOOL;
    nSlaveAddr  : UINT := 1006;
    nIdn        : WORD  := 15;
    nElement    : BYTE := 0;
    nDriveNo    : BYTE := 0;
    bCommand    : BOOL := FALSE;
    val         : UINT;
    bError      : BOOL;
    nErrId      : UDINT;
END_VAR

fbSoEWrite(sNetId:= sNetId,nSlaveAddr :=nSlaveAddr, nIdn := nIdn, nElement:=nElement, nDriveNo := nD
riveNo,bCommand:=bCommand, pSrcBuf:= ADR(val), cbBufLen:=SIZEOF(val),bExecute:=bExecute);
bError := fbSoEWrite.bError;
nErrId := fbSoEWrite.nErrId;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

8.3 FB_SoERead_ByDriveRef



Mit dem Funktionsbaustein `FB_SoeRead_ByRef` können Antriebsparameter mit Hilfe des "Servo Drive Profile over EtherCAT (SoE)"-Protokolls gelesen werden. Dazu muss der Slave eine Mailbox besitzen und das SoE-Protokoll unterstützen. Der zu lesende Antriebsparameter wird mit den Parametern nIdn (Identification number), nElement und stDriveRef spezifiziert.

Über die globale Variable `bSeqReadDrvAttrAndValue := TRUE` der Tc2_EtherCAT-Bibliothek kann ein sequentieller Zugriff auf Attribut und Wert erzwungen werden. Der Standard-Wert dieser Variablen ist FALSE. Geräte der AX5xxx-Serie erlauben den parallelen und den sequentiellen Zugriff auf Attribut und Wert. Bei Geräten von Fremdherstellern kann es erforderlich sein, den Zugriff auf Attribut und Wert zu separieren, was den Zugriff aber insgesamt um einige Zyklen verlangsamt.

Eingänge

```
VAR_INPUT
    stDriveRef : ST_DriveRef; (* contains sNetID of EcMaster, nSlaveAddr of EcDrive, nDriveNo of EcD
rive, either preset or read from NC *)
    nIdn       : WORD; (* SoE IDN: e.g. "S_0_IDN + 1" for S-0-0001 or "P_0_IDN + 23" for P-0-0023*)
    nElement   : BYTE; (* SoE element.*)
    pDstBuf    : PVOID; (* Contains the address of the buffer for the received data. *)
    cbBufLen   : UDINT; (* Contains the max. number of bytes to be received. *)
    bExecute   : BOOL; (* Function block execution is triggered by a rising edge at this input.*)
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;(* States the time before the function is cancelled. *)
END_VAR
```

Name	Typ	Beschreibung
stDriveRef	ST_DriveRef	Die Referenz auf den Antrieb kann im System Manager direkt in die SPS gelinkt werden. Hierzu muss eine Instanz der ST_PlcDriveRef verwendet werden und die NetID vom Bytearray in einen String konvertiert werden.
nIdn	WORD	Identifikations-Nummer des zu lesenden Parameters
nElement	BYTE	Element-Nummer des zu lesenden Parameters (Siehe nElement)
pDstBuf	PVOID	Adresse (Pointer) auf den lesenden Puffer
cbBufLen	UDINT	Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

nElement

Element-Nummer des zu lesenden Parameters. Folgende Werte sind zulässig:

Wert	Beschreibung
0x01	Data Status
0x02	Name (read only)
0x04	Attribut
0x08	Einheit
0x10	Minimum
0x20	Maximum
0x40	Wert
0x80	Default

Ausgänge

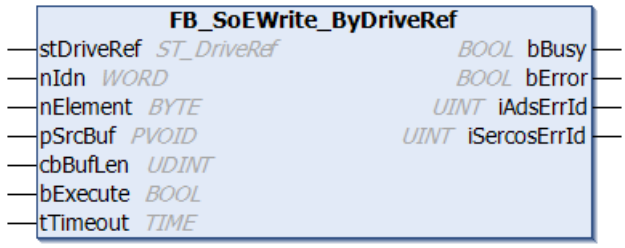
```
VAR_OUTPUT
    bBusy        : BOOL;
    bError       : BOOL;
    iAdsErrId    : UINT;
    iSercosErrId : UINT;
    dwAttribute  : DWORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
iAdsErrId	UINT	Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
iSercosErrId	UINT	Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehls.
dwAttribute	DWORD	Liefert das Attribut des Sercos-Parameters.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

8.4 FB_SoEWrite_ByDriveRef



Mit dem Funktionsbaustein `FB_SoeWrite_ByRef` können Antriebsparameter mit Hilfe des "Servo Drive Profile over EtherCAT (SoE)"-Protokolls beschrieben werden. Dazu muss der Slave eine Mailbox besitzen und das SoE-Protokoll unterstützen. Der zu schreibende Antriebs-Parameter wird mit den Parametern nIdn (Identification number), nElement und stDriveRef spezifiziert.

Über die globale Variable `bSeqReadDrvAttrAndValue := TRUE` der Tc2_EtherCAT Bibliothek kann ein sequentieller Zugriff auf Attribut und Wert erzwungen werden. Der Standard-Wert dieser Variablen ist FALSE. Geräte der AX5xxx-Serie erlauben den parallelen und den sequentiellen Zugriff auf Attribut und Wert. Bei Geräten von Fremdherstellern kann es erforderlich sein, den Zugriff auf Attribut und Wert zu separieren, was den Zugriff aber insgesamt um einige Zyklen verlangsamt.

Eingänge

```
VAR_INPUT
    stDriveRef : ST_DriveRef; (* contains sNetID of EcMaster, nSlaveAddr of EcDrive, nDriveNo of EcD
rive, either preset or read from NC *)
    nIdn       : WORD; (* SoE IDN: e.g. "S_0_IDN + 1" for S-0-0001 or "P_0_IDN + 23" for P-0-0023*)
    nElement   : BYTE; (* SoE element.*)
    pSrcBuf    : PVOID; (* Contains the address of the buffer containing the data to be send. *)
    cbBufLen   : UDINT; (* Contains the max. number of bytes to be received. *)
    bExecute   : BOOL; (* Function block execution is triggered by a rising edge at this input.*)
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;(* States the time before the function is cancelled. *)
END_VAR
```

Name	Typ	Beschreibung
stDriveRef	ST_DriveRef	Die Referenz auf den Antrieb kann im System Manager direkt in die SPS gelinkt werden. Hierzu muss eine Instanz der ST_PlcDriveRef verwendet werden und die NetID vom Bytearray in einen String konvertiert werden.
nIdn	WORD	Identifikations-Nummer des zu lesenden Parameters
nElement	BYTE	Element-Nummer des zu lesenden Parameters (Siehe nElement)
pSrcBuf		Adresse (Pointer) auf den sendenden Puffer
cbBufLen		Maximal verfügbare Puffergröße für die zu lesenden Daten in Bytes
bExecute	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.
tTimeout	TIME	Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

nElement

Element-Nummer des zu lesenden Parameters. Folgende Werte sind zulässig:

Wert	Beschreibung
0x01	Data Status
0x02	Name (read only)
0x04	Attribut
0x08	Einheit
0x10	Minimum
0x20	Maximum
0x40	Wert
0x80	Default

Ausgänge

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

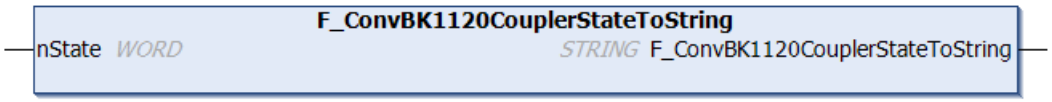
Name	Typ	Beschreibung
bBusy	BOOL	Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt gesetzt, bis eine Rückmeldung erfolgt.
bError	BOOL	Dieser Ausgang wird gesetzt, nachdem der bBusy-Ausgang zurückgesetzt wurde, wenn bei der Übertragung des Kommandos ein Fehler auftritt.
iAdsErrId	UINT	Liefert bei gesetztem bError-Ausgang den ADS-Fehlercode des zuletzt ausgeführten Befehls.
iSercos ErrId	UINT	Liefert bei gesetztem bError-Ausgang den Sercos-Fehler des zuletzt ausgeführten Befehles.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9 Konvertierungsfunktionen

9.1 F_ConvBK1120CouplerStateToString



Die Funktion `F_ConvBK1120CouplerStateToString` liefert den Kopplerstatus des BK1120/BK1150/BK1250 als String. Bei nState = 0 wird ‘No error‘ geliefert, sonst, z.B. bei nState = 1, wird ‘K-Bus error‘ geliefert. Wenn mehrere Fehler anstehen, werden diese mit Komma separiert.

Eingänge

```
VAR_INPUT
    nState : WORD;
END_VAR
```

Name	Typ	Beschreibung

nState	WORD	Kopplerstatus (CouplerState), kann im System Manager von den Eingängen des BK1120/BK1250 in die SPS gelinkt werden.
		`0x0000 = 'No error'`
		`0x0001 = 'K-Bus error'`
		`0x0002 = 'Configuration error'`
		`0x0010 = 'Outputs disabled'`
		`0x0020 = 'K-Bus overrun'`
		`0x0040 = 'Communication error (Inputs)'`
		`0x0080 = 'Communication error (Outputs)'`

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.2 F_ConvMasterDevStateToString

Die Funktion `F_ConvMasterDevStateToString` wandelt den Gerätestatus des EtherCAT-Masters in einen String.

Bei nState = 0 wird ‘OK‘ geliefert, sonst, z.B. bei nState = 1, wird ‘Not OK – Link error‘ geliefert. Wenn mehrere Fehler anstehen, werden diese mit Bindestrich separiert.

Eingänge

```
VAR_INPUT
    nState : WORD;
END_VAR
```

Name	Typ	Beschreibung
nState	WORD	Gerätestatus des EtherCAT-Masters, kann als DevState im System Manager von den Eingängen des EtherCAT-Masters in die SPS gelinkt werden. 0x0001 = 'Link error' 0x0002 = 'I/O locked after link error (I/O reset required)' 0x0004 = 'Link error (redundancy adapter)' 0x0008 = 'Missing one frame (redundancy mode)' 0x0010 = 'Out of send resources (I/O reset required)' 0x0020 = 'Watchdog triggered' 0x0040 = 'Ethernet driver (miniport) not found' 0x0080 = 'I/O reset active' 0x0100 = 'At least one device in 'INIT' state' 0x0200 = 'At least one device in 'PRE-OP' state' 0x0400 = 'At least one device in 'SAFE-OP' state' 0x0800 = 'At least one device indicates an error state' 0x1000 = 'DC not in sync'

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.3 F_ConvProductCodeToString

Die Funktion `F_ConvProductToString` liefert den ProductCode als String, z.B. 'EL6731-0000-0017'. Ab Version 3.3.8.0 der Tc2_EtherCAT-Bibliothek unterstützt diese Funktion auch ELM- und EPP-Slaves wie ‘EPP4374-0002-0018‘ und ‘ELM3704-0001-0016‘.

Eingänge

```
VAR_INPUT
    stSlaveIdentity : ST_EcSlaveIdentity;
END_VAR
```

Name	Typ	Beschreibung
stSlaveIdentity	ST_EcSlaveIdentity	Slave Identity, wie sie mit dem FB_EcGetSlaveIdentity [▶ 30] eingelesen werden kann.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.4 F_ConvSlaveStateToString

Die Funktion `F_ConvSlaveStateToString` liefert den EtherCAT-Slave-State als String. Zur Konvertierung in den String siehe F_ConvStateToString [▶ 81].

Eingänge

```
VAR_INPUT
    state : ST_EcSlaveState;
END_VAR
```

Name	Typ	Beschreibung
state	ST_EcSlaveState	EtherCAT-Slave-State-Struktur (bestehend aus: deviceState : BYTE; linkState : BYTE;)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.5 F_ConvSlaveStateToBits

Die Funktion `F_ConvSlaveStateToBits` liefert den EtherCAT-Slave-State als Struktur TYPE ST_EcSlaveStateBits [▶ 122].

Eingänge

```
VAR_INPUT
    stEcSlaveState : ST_EcSlaveState;
END_VAR
```

Name	Typ	Beschreibung
stEcSlaveState	ST_EcSlaveState	EtherCAT-Slave-State-Struktur (bestehend aus: deviceState : BYTE; linkState : BYTE;)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.6 F_ConvSlaveStateToBitsEx

Die Funktion `F_ConvSlaveStateToBitsEx` liefert den EtherCAT-Slave-State als Struktur ST_EcSlaveStateBitsEx [▶ 122].

Eingänge

```
VAR_INPUT
    stEcSlaveState : ST_EcSlaveState;
END_VAR
```

Name	Typ	Beschreibung
stEcSlaveState	ST_EcSlaveState	EtherCAT-Slave-State-Struktur (bestehend aus: deviceState : BYTE; linkState : BYTE;)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

9.7 F_ConvStateToString

Die Funktion `F_ConvStateToString` liefert den EtherCAT-Slave-State als String. Bei nState = 0 wird ‘ ‘ geliefert, sonst, z.B. bei nState = 1, wird ‘INIT ‘ geliefert. Wenn mehrere Meldungen anstehen, werden diese mit Leerzeichen separiert.

Eingänge

```
VAR_INPUT
    nState : WORD;
END_VAR
```

Name	Typ	Beschreibung

nState	WORD	EtherCAT-Slave-State als WORD
		`0x___1 = 'INIT'`
		`0x___2 = 'PREOP'`
		`0x___3 = 'BOOT'`
		`0x___4 = 'SAFEOP'`
		`0x___8 = 'OP'`
		`0x001_ = 'Slave signals error'`
		`0x002_ = 'Invalid vendorId, productCode... read'`
		`0x004_ = 'Initialization error occurred'`
		`0x008_ = 'Slave disabled'`
		`0x010_ = 'Slave not present'`
		`0x020_ = 'Slave signals link error'`
		`0x040_ = 'Slave signals missing link'`
		`0x080_ = 'Slave Slave signals unexpected link'`
		`0x100_ = 'Communication port A'`
		`0x200_ = 'Communication port B'`
		`0x400_ = 'Communication port C'`
		`0x800_ = 'Communication port D'`

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10 Distributed Clocks

10.1 DCTIME32

10.1.1 ConvertDcTimeToPos

Dieser Funktionsblock konvertiert eine 32-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME32 [▶ 125] in eine zugehörige NC-Achsposition (d.h. diejenige NC-Achsposition, die genau zu diesem Zeitpunkt vorgelegen hat bzw. vorliegen wird).

Eingänge

```
VAR_INPUT
    nAxisId : UDINT;
    nSubIdx : UDINT;
    dcTime  : T_DCTIME32;(* 32 bit distributed clock time *)
END_VAR
```

Name	Typ	Beschreibung
nAxisId	UDINT	ID der NC-Achse
nSubIdx	UDINT	Diese 32-Bit-Eingangsgröße beinhaltet zwei verschiedene Informationen und unterteilt sich somit in zwei 16-Bit-Werte: • Das LowWord (die niederwertigsten 16 Bit) beinhaltet den Sub-Index zum relativen Adressieren eines Encoder-Unterelementes an einer Achse. Der Sub-Index wird von Null an gezählt. Für den typischen Fall einer Achse mit genau einem Encoder ist der Sub-Index Null richtig. • Das HighWord (die höchstwertigen 16 Bit) beinhaltet ein Steuerwort (Bitmaske), das die Art der Positionsberechnung beeinflusst (z.B. den Inter- bzw. Extrapolationstyp). Die Bitmaske 0x0001 bedeutet, dass die Soll-Beschleunigung der Achse mit in die Berechnung einbezogen werden soll.
dcTime	T_DCTIME32	32-Bit-„Distributed Clock System Time“-Variable. Diese Eingangsgröße wird in die korrespondierende NC-Achsposition umgerechnet.

Diese 32-Bit-Zeit darf nur im zeitlichen Nahbereich von ± 2.147 Sekunden um die aktuelle Systemzeit verwendet werden, da sie nur hier eindeutig ist. Innerhalb des Funktionsblocks kann diese Voraussetzung nicht überprüft werden.

Ausgänge

```
VAR_OUTPUT
    fPosition : LREAL;
    iErr      : UDINT;
END_VAR
```

Name	Typ	Beschreibung
fPosition	LREAL	Liefert die zur dcTime korrespondierende NC-Achsposition. Hierbei handelt es sich um eine um die Skalierung und Offset verrechnete NC-Achsposition mit z.B. der physikalischen Einheit Grad oder mm.
iErr	UDINT	Liefert im Fehlerfall eine Fehlernummer, z.B. Fehler 0x4012 (Achs-ID ist nicht erlaubt bzw. Achse ist im System nicht vorhanden).

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.1.2 ConvertPosToDcTime

Dieser Funktionsblock konvertiert eine NC-Achsposition in eine zugehörige 32-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME32 [▶ 125] (d. h. derjenige Zeitpunkt, zu dem genau diese NC-Achsposition erreicht wurde bzw. erreicht wird).

Eingänge

```
VAR_INPUT
    nAxisId   : UDINT;
    nSubIdx   : UDINT;
    fPosition : LREAL;
END_VAR
```

Name	Typ	Beschreibung
nAxisId	UDINT	ID der NC-Achse
nSubIdx	UDINT	Diese 32-Bit-Eingangsgröße setzt sich aus zwei verschiedenen Informationen zusammen und unterteilt sich in zwei 16-Bit-Werte: • Das LowWord (die niederwertigsten 16 Bit) beinhaltet den Sub-Index zum relativen Adressieren eines Encoder-Unterelementes an einer Achse. Der Sub-Index wird von Null an gezählt. Für den typischen Fall einer Achse mit genau einem Encoder ist der Sub-Index Null richtig. • Das HighWord (die höchstwertigen 16 Bit) beinhaltet ein Steuerwort (Bitmaske), das die Art der Positionsberechnung beeinflusst (z. B. den Inter- bzw. Extrapolationstyp). Die Bitmaske 0x0001 bedeutet, dass die Soll-Beschleunigung der Achse mit in die Berechnung einbezogen werden soll.
fPosition	LREAL	NC-Achsposition, die in die korrespondierende 32-Bit-„Distributed Clock System Time“-Variable umgerechnet wird. Wenn die zur Position zugehörige „Distributed Clock System Time“ außerhalb des erwarteten Zeitfensters von ± 2.147 Sekunden liegt, wird diese Umrechnung mit einer Fehlernummer abgelehnt.

Ausgänge

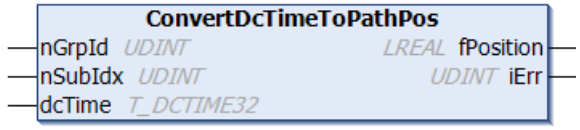
```
VAR_OUTPUT
    dcTime : T_DCTIME32;(* 32 bit distributed clock time *)
    iErr   : UDINT;
END_VAR
```

Name	Typ	Beschreibung
dcTime	T_DCTIME32	Liefert die zum Eingang fPosition zugehörige 32-Bit-„Distributed Clock System Time“-Variable.
iErr	UDINT	Liefert im Fehlerfall eine Fehlernummer, z.B. • Fehler 0x4012: Achs-ID ist nicht erlaubt bzw. Achse ist im System nicht vorhanden. • Fehler 0x4361: Zeitbereichsüberschreitung (Zukunft) • Fehler 0x4362: Zeitbereichsüberschreitung (Vergangenheit) • Fehler 0x4363: Position ist mathematisch nicht zu ermitteln.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.1.3 ConvertDcTimeToPathPos



Dieser Funktionsblock konvertiert eine 32-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME32 [▶ 125] in einen relativen Nci-Wegabstand auf der Kontur des momentan aktiven Nci-Programms (d.h. je nach Zeitpunkt, liefert der Funktionsblock einen positiven oder negativen relativen Abstand zurück).

Eingänge

```
VAR_INPUT
    nGrpId  : UDINT;
    nSubIdx : UDINT;
    dcTime  : T_DCTIME32;(* 32 bit distributed clock time *)
END_VAR
```

Name	Typ	Beschreibung
nGrpId	UDINT	Group ID des zugehörigen Nci-Kanals
nSubIdx	UDINT	Diese 32-Bit-Eingangsgröße beinhaltet zwei verschiedene Informationen und unterteilt sich somit in zwei 16-Bit-Werte: • Das LowWord (die niederwertigsten 16 Bit) beinhaltet den Sub-Index zum relativen Adressieren eines Encoder-Unterelementes an einer Achse. Der Sub-Index wird von Null an gezählt. Für den typischen Fall einer Achse mit genau einem Encoder ist der Sub-Index Null richtig. • Das HighWord (die höchstwertigen 16 Bit) beinhaltet ein Steuerwort (Bitmaske), das die Art der Positionsberechnung beeinflusst (z.B. den Inter- bzw. Extrapolationstyp). Die Bitmaske 0x0001 bedeutet, dass die Soll-Beschleunigung der Achse mit in die Berechnung einbezogen werden soll. Die Bitmaske 0x0010 bedeutet, dass die Berechnung relativ erfolgt und ist momentan zwingend. Der Aufruf wird anderenfalls mit Fehler abgelehnt.
dcTime	T_DCTIME32	32-Bit-“Distributed Clock System Time”-Variable. Diese Eingangsgröße wird in den korrespondierenden relativen Nci-Wegabstand auf der Kontur umgerechnet.

Diese 32-Bit-Zeit darf nur im zeitlichen Nahbereich von ± 2.147 Sekunden um die aktuelle Systemzeit verwendet werden, da sie nur hier eindeutig ist. Innerhalb des Funktionsblocks kann diese Voraussetzung nicht überprüft werden.

Ausgänge

```
VAR_OUTPUT
    fPosition : LREAL;
    iErr      : UDINT;
END_VAR
```

Name	Typ	Beschreibung
fPosition	LREAL	Liefert den zur dcTime korrespondierende relative Nci-Wegabstand auf der Kontur.
iErr	UDINT	Liefert im Fehlerfall eine Fehlernummer

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.1.4 ConvertPathPosToDcTime

Dieser Funktionsblock konvertiert einen relativen Nci-Wegabstand in eine zugehörige 32-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME32 [▶ 125] (d.h. derjenige zugehörige Zeitpunkt, der dem relativen Nci-Wegabstand entspricht bzw. entsprach).

Eingänge

```
VAR_INPUT
    nGrpId    : UDINT;
    nSubIdx   : UDINT;
    fPosition : LREAL;
END_VAR
```

Name	Typ	Beschreibung
nGrpId	UDINT	Group ID des zugehörigen Nci-Kanal
nSubIdx	UDINT	Diese 32-Bit-Eingangsgröße beinhaltet zwei verschiedene Informationen und unterteilt sich somit in zwei 16-Bit-Werte: • Das LowWord (die niederwertigsten 16 Bit) beinhaltet den Sub-Index zum relativen Adressieren eines Encoder-Unterelementes an einer Achse. Der Sub-Index wird von Null an gezählt. Für den typischen Fall einer Achse mit genau einem Encoder ist der Sub-Index Null richtig. • Das HighWord (die höchstwertigen 16 Bit) beinhaltet ein Steuerwort (Bitmaske), das die Art der Positionsberechnung beeinflusst (z.B. den Inter- bzw. Extrapolationstyp). Die Bitmaske 0x0001 bedeutet, dass die Soll-Beschleunigung der Achse mit in die Berechnung einbezogen werden soll. Die Bitmaske 0x0010 bedeutet, dass die Berechnung relativ erfolgt und ist momentan zwingend. Der Aufruf wird anderenfalls mit einem Fehler abgelehnt.
fPosition	LREAL	Relativer Nci-Wegabstand, der in die korrespondierende 32-Bit-„Distributed Clock System Time“ umgerechnet wird. Wenn die zum relativen Nci-Wegabstand zugehörige „Distributed Clock System Time“ außerhalb des erwarteten Zeitfensters von ± 2.147 Sekunden liegt, wird diese Umrechnung mit einer Fehlernummer abgelehnt.

Ausgänge

```
VAR_OUTPUT
    dcTime : T_DCTIME32;(* 32 bit distributed clock time *)
    iErr   : UDINT;
END_VAR
```

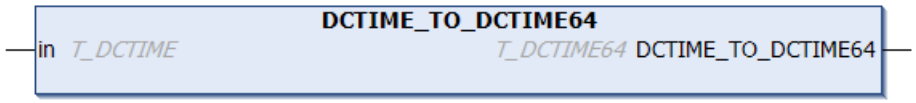
Name	Typ	Beschreibung
dcTime	T_DCTIME32	Liefert die zum Eingang fPosition zugehörige 32-Bit-„Distributed Clock System Time“-Variable.
iErr	UDINT	Liefert im Fehlerfall eine Fehlernummer, z.B. • Fehler 0x4361: Zeitbereichsüberschreitung (Zukunft) • Fehler 0x4362: Zeitbereichsüberschreitung (Vergangenheit) • Fehler 0x4363: Position ist mathematisch nicht zu ermitteln

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2 DCTIME64

10.2.1 DCTIME_TO_DCTIME64



Die Funktion konvertiert eine „Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126] in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125].

FUNCTION DCTIME_TO_DCTIME64: T_DCTIME64

Eingänge

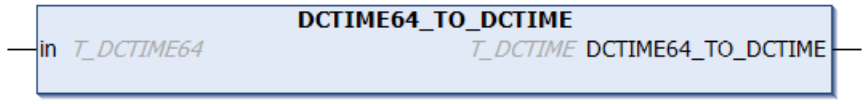
```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	Die zu konvertierende „Distributed Clock System Time”-Variable

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.2 DCTIME64_TO_DCTIME



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in ein „Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126].

FUNCTION DCTIME64_TO_DCTIME: T_DCTIME

Eingänge

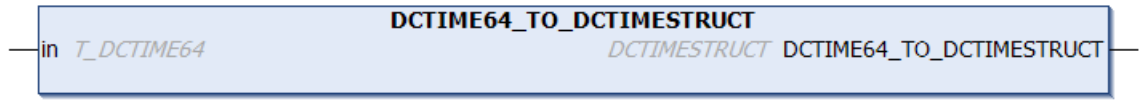
```
VAR_INPUT
    in : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.3 DCTIME64_TO_DCTIMESTRUCT



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in eine strukturierte Variable vom Typ DCTIMESTRUCT [▶ 124].

FUNCTION DCTIME64_TO_DCTIMESTRUCT

Eingänge

```
VAR_INPUT
    in : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

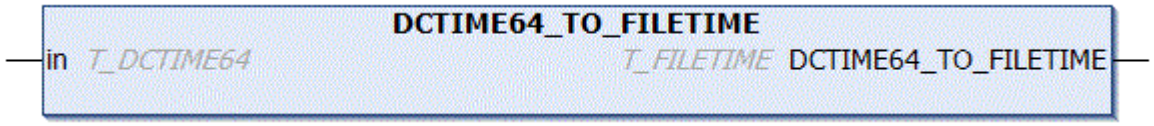
Beispiel:

```
PROGRAM P_TEST
VAR
    dcStruct : DCTIMESTRUCT;
    dcTime : T_DCTIME64;
END_VAR

dcTime : F_GetCurDcTickTime64();
dcStruct := DCTIME64_TO_DCTIMESTRUCT (dcTIME);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.4 DCTIME64_TO_FILETIME64



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in eine 64-Bit-„Windows File Time“-Variable vom Typ T_FILETIME64.

FUNCTION DCTIME64_TO_FILETIME64: T_FILETIME64

Eingänge

```
VAR_INPUT
    in : T_DCTIME64:
END_VAR;
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

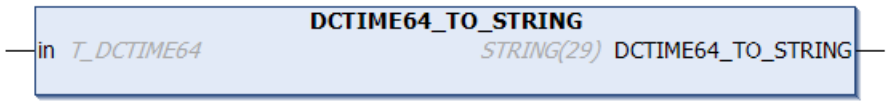
Beispiel:

```
PROGRAM P_TEST
VAR
    ft  : T_FILETIME64;
    dct : T_DCTIME64;
END_VAR

dct := F_GetCurDcTickTime64();
ft  := DCTIME64_TO_FILETIME64(dct);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.4024	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.3.16.0

10.2.5 DCTIME64_TO_STRING



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in einen String.

Der resultierende String hat nach der Konvertierung das folgende Format: **‘YYYY-MM-DD-hh:mm:ss.nnnnnnnnn‘**

- YYYY: Jahr;

- MM: Monat;
- DD: Tag;
- hh: Stunde;
- mm: Minute;
- ss: Sekunde;
- nnnnnnnnn: Nanosekunden

FUNCTION DCTIME64_TO_STRING: STRING (29)

Eingänge

```
VAR_INPUT
    in : T_DCTIME64; (*Distributed Clock Time*)
END_VAR
```

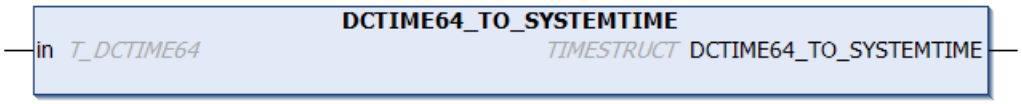
Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

Beispiel:
Siehe Beschreibung der Funktion: F_GetCurDcTickTime64 [▶ 96].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.6 DCTIME64_TO_SYSTEMTIME



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in eine strukturierte „Windows System Time“- Variable vom Typ TIMESTRUCT.

DCTIME64_TO_SYSTEMTIME: TIMESTRUCT

Eingänge

```
VAR_INPUT
    in : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

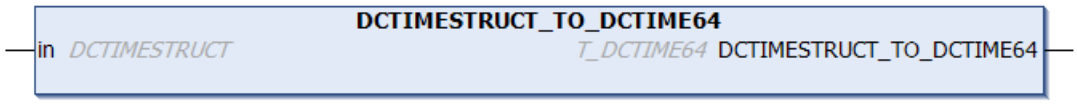
Beispiel:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT
END_VAR

syst := DCTIME64_TO_SYSTEMTIME ( F_GetCurDcTickTime64() )
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.7 DCTIMESTRUCT_TO_DCTIME64



Die Funktion konvertiert die strukturierte Variable vom Typ DCTIMESTRUCT [▶ 124] in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125]. Die Strukturkomponente wDayofWeek wird bei der Konvertierung ignoriert. Die Strukturkomponente wYear muss einen Wert größer oder gleich 2000 oder kleiner 2584 haben. Bei unzulässigen Werten der Strukturkomponenten liefert die Funktion den Wert Null zurück.

FUNCTION DCTIMESTRUCT_TO_DCTIME64: T_DCTIME64

Eingänge

```
VAR_INPUT
    in : DCTIMESTRUCT;
END_VAR
```

Name	Typ	Beschreibung
in	DCTIMESTRUCT	Die zu konvertierende strukturierte Variable

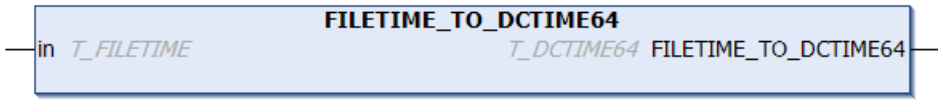
Beispiel:

```
PROGRAM P_TEST
VAR
    dcStruct : DCTIMESTRUCT := ( wYear := 2008, wMonth := 3, wDay := 13,
                    wHour := 1, wMinute := 2, wSecond :=3,
                    wMilliseconds := 123, wMicroseconds := 456, wNanoseconds := 789 );
    dc64 : T_DCTIME64;
END_VAR

dc64 := DCTIMESTRUCT_TO_DCTIME64( dcStruct );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.8 FILETIME64_TO_DCTIME64



Die Funktion konvertiert die 64-Bit-„Windows File Time“-Variable vom Typ T_FILETIME64 in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125]. Bei einem Konvertierungsfehler liefert die Funktion den Wert Null zurück.

FUNCTION FILETIME64_TO_DCTIME64: T_DCTIME64

Eingänge

```
VAR_INPUT
    in : T_FILETIME64;
END_VAR
```

Name	Typ	Beschreibung
in	T_FILETIME64	Die zu konvertierende "Windows File Time“-Variable

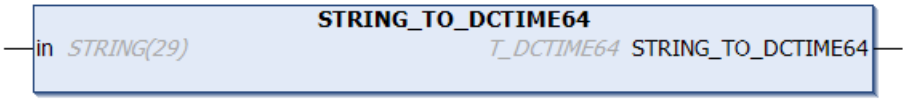
Beispiel:

```
PROGRAM P_TEST
VAR
    ft : T_FILETIME64;
    dct : T_DCTIME64;
END_VAR

ft := F_GetSystemTime();
dct := FILETIME64_TO_DCTIME64(ft);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.4024	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT >= 3.3.16.0

10.2.9 STRING_TO_DCTIME64



Die Funktion konvertiert einen String in eine „Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125].

FUNCTION STRING_TO_DCTIME64: T_DCTIME64

Eingänge

```
VAR_INPUT
    in : STRING(29);
END_VAR
```

Name	Typ	Beschreibung
in	STRING	Der zu konvertierende String Der String muss folgendes Format haben:**‘YYYY-MM-DD-hh:mm:ss.nnnnnnnnn‘** • YYYY: Jahr; • MM: Monat; • DD: Tag; • hh: Stunde; • mm: Minute; • ss: Sekunde; • nnnnnnnnn: Nanosekunden

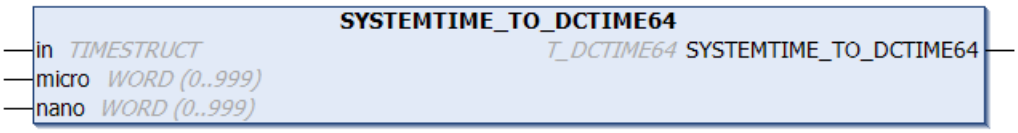
Beispiel:

Siehe Beschreibung der Funktion F_GetCurDcTickTime64 [▶ 96].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.10 SYSTEMTIME_TO_DCTIME64



Die Funktion konvertiert die strukturierte „Windows System Time“-Variable vom Typ TIMESTRUCT in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125]. Bei einem Konvertierungsfehler liefert die Funktion den Wert Null zurück.

FUNCTION SYSTEMTIME_TO_DCTIME64: T_DCTIME64

Eingänge

```
VAR_INPUT
    in    : TIMESTRUCT;
    micro : WORD(0..999); (* Microseconds: 0..999 *)
    nano  : WORD(0..999); (* Nanoseconds: 0..999 *)
END_VAR
```

Name	Typ	Beschreibung
in	TIMESTRUCT	Die zu konvertierende „Windows System Time"-Variable
micro	WORD	Microsekunden
nano	WORD	Nanosekunden

Beispiel:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT := ( wYear := 2009, wMonth := 9, wDay := 16, wHour := 12, wMinute := 22, wSeco
nd := 44, wMilliseconds := 123 );
END_VAR

dct := SYSTEMTIME_TO_DCTIME64( syst, 456, 789 );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.2.11 FB_EcDcTimeCtrl64

Mit diesem Funktionsbaustein können die einzelnen Komponenten wie Jahr, Monat, Tag usw. einer 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] gelesen werden. Der Funktionsbaustein besitzt mehrere A_GETXYZ- Aktionen. Nach dem Aufruf der gewünschten Aktion steht der Wert der XYZ- Komponente in der get-Ausgangsvariablen zur Verfügung. Die put- Eingangsvariabel wird zurzeit nicht benutzt.

Der Funktionsbaustein besitzt folgende Aktionen:

- A_GetYear
- A_GetMonth
- A_GetDay
- A_GetDayOfWeek
- A_GetHour
- A_GetMinute
- A_GetSecond
- A_GetMilli
- A_GetMicro
- A_GetNano

Eingänge

```
VAR_IN_OUT
    put : WORD;
END_VAR
```

Name	Typ	Beschreibung
put	WORD	Eingangsparameter (zurzeit nicht benutzt)

Ein-/Ausgänge

```
VAR_IN_OUT
    in : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

Ausgänge

```
VAR_IN_OUT
    bError : BOOL;
    get    : WORD;
END_VAR
```

Name	Typ	Beschreibung
bError	BOOL	Dieser Ausgang wird gesetzt, wenn ein Fehler beim Aktionsaufruf aufgetreten ist.
get	WORD	Ausgangsparameter (Jahr, Monat, Tag, usw.)

Beispiel für eine Implementierung in ST:

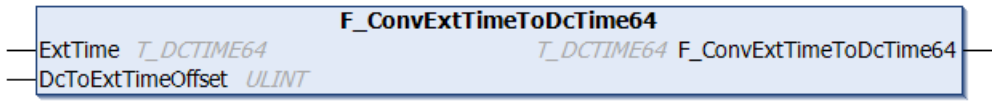
```
PROGRAM P_TEST
VAR
    dcStruct   : DCTIMESTRUCT;
    dcTime     : T_DCTIME64;
    fbCtrl     : FB_EcDcTimeCtrl;

    wYear      : WORD;
    wMonth     : WORD;
    wDay       : WORD;
    wDayOfWeek : WORD;
    wHour      : WORD;
    wMinute    : WORD;
    wSecond    : WORD;
    wMilli     : WORD;
    wMicro     : WORD;
    wNano      : WORD;
END_VAR

dcTime := F_GetCurDcTickTime64();
fbCtrl.A_GetYear( in := dcTime, get => wYear );
fbCtrl.A_GetMonth( in := dcTime, get => wMonth );
fbCtrl.A_GetDay( in := dcTime, get => wDay );
fbCtrl.A_GetDayOfWeek( in := dcTime, get => wDayOfWeek );
fbCtrl.A_GetHour( in := dcTime, get => wHour );
fbCtrl.A_GetMinute( in := dcTime, get => wMinute );
fbCtrl.A_GetSecond( in := dcTime, get => wSecond );
fbCtrl.A_GetMilli( in := dcTime, get => wMilli );
fbCtrl.A_GetMicro( in := dcTime, get => wMicro );
fbCtrl.A_GetNano( in := dcTime, get => wNano );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3 DCTIME64 und ULINT

10.3.1 F_ConvExtTimeToDcTime64



Die Funktion `F_ConvExtTimeToDcTime64` konvertiert eine externe Zeit in die TwinCAT „Distributed Clock“-Systemzeit.

FUNCTION F_ConvExtTimeToDcTime64: T_DCTIME64

Eingänge

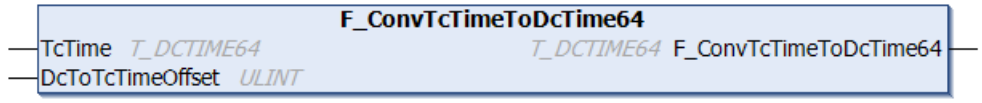
```
VAR_INPUT
   ExtTime           : T_DCTIME64;
   DcToExtTimeOffset : ULINT;
END_VAR
```

Name	Typ	Beschreibung
ExtTime	T_DCTIME64	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeitformat
DcToExtTime Offset	ULINT	Zeitoffset zwischen der TwinCAT „Distributed Clock“-Systemzeit und einer externer Zeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.2 F_ConvTcTimeToDcTime64



Die Funktion `F_ConvTcTimeToDcTime64` konvertiert die TwinCAT Systemzeit in die TwinCAT „Distributed Clock“-Systemzeit.

FUNCTION F_ConvTcTimeToDcTime64: T_DCTIME64

Eingänge

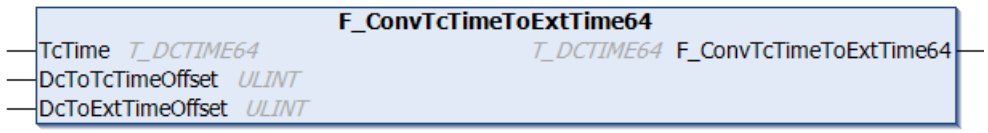
```
VAR_INPUT
   TcTime           : T_DCTIME64;
   DcToTcTimeOffset : ULINT;
END_VAR
```

Name	Typ	Beschreibung
TcTime	T_DCTIME64	TwinCAT Systemzeit im TwinCAT „Distributed Clock“-Systemzeitformat
DcToTcTimeOffset	ULINT	Zeitoffset zwischen der TwinCAT „Distributed Clock“-Systemzeit und der TwinCAT Systemzeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.3 F_ConvTcTimeToExtTime64



Die Funktion `F_ConvTcTimeToExtTime64` konvertiert die TwinCAT „Distributed Clock“-Systemzeit in eine externe Zeit.

FUNCTION F_ConvTcTimeToExtTime64: T_DCTIME64

Eingänge

```
VAR_INPUT
   TcTime            : T_DCTIME64;
   DcToTcTimeOffset  : ULINT;
   DcToExtTimeOffset : ULINT;
END_VAR
```

Name	Typ	Beschreibung
TcTime	T_DCTIME64	TwinCAT Systemzeit im „Distributed Clock“-Format
DcToTcTimeOffset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTimeOffset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.4 F_GetActualDcTime64

Diese Funktion liefert die aktuelle Zeit im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME64 [▶ 125]).

FUNCTION F_GetActaulDCTime: T_DCTIME64

Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

Beispiel in ST:

```
PROGRAM MAIN
VAR
    actDC : T_DCTIME64;
    sAct  : STRING;
END_VAR

actDC := F_GetActualDcTime64();
sAct  := DCTIME64_TO_STRING( actDC );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.5 F_GetCurDcTaskTime64

Diese Funktion liefert die Startzeit der Task, der Zeitpunkt zu dem die Task starten soll, im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME64 [▶ 125]). Die Funktion liefert immer die Startzeit der Task in der sie aufgerufen wurde.

FUNCTION F_GetCurDcTaskTime64: T_DCTIME64

Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.6 F_GetCurDcTickTime64

Die Funktion liefert die Zeit des aktuellen (letzten) Ticks im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME64 [▶ 125]).

FUNCTION F_GetCurDcTickTime64: T_DCTIME64

Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

Beispiel:

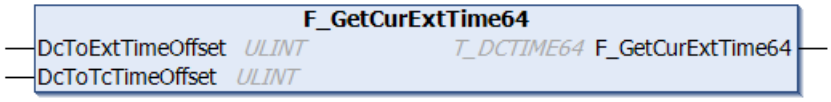
```
PROGRAM MAIN
VAR
    tDC : T_DCTIME64;
    sDC : STRING;
    tDCBack : T_DCTIME64;

    sDCZero : STRING;(* DCTIME64 = zero time starts on 01.01.2000 *)
    tDCBackFromZero : T_DCTIME64;

    tDCFromString : T_DCTIME64;
    sDCBackFromString : STRING;
END_VAR

tDC := F_GetCurDcTickTime64();
sDC := DCTIME64_TO_STRING( tDC );
tDCBack := STRING_TO_DCTIME64( sDC );

sDCZero := DCTIME64_TO_STRING( ULARGE_INTEGER(0, 0) );
tDCBackFromZero := STRING_TO_DCTIME64( sDCZero );
```

```
tDCFromString := STRING_TO_DCTIME64( '2007-03-09-11:31:09.223456789' );
sDCBackFromString := DCTIME64_TO_STRING( tDCFromString );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.7 F_GetCurExtTime64



Die Funktion liefert die externe Zeit im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME64 [▶ 125]).

FUNCTION F_GetCurExtTime64: T_DCTIME64

Eingänge

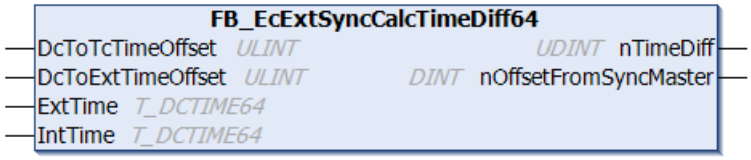
```
VAR_INPUT
   DcToExtTimeOffset : ULINT;
   DcToTcTimeOffset  : ULINT;
END_VAR
```

Name	Typ	Beschreibung
DcToExtTime Offset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
DcToTcTime Offset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.8 FB_EcExtSyncCalcTimeDiff64



Der Funktionsbaustein `FB_EcExtSyncCalcTimeDiff64` berechnet die Differenz zwischen externer und interner Zeit unter Berücksichtigung der Zeitoffsets.

Ein-/Ausgänge

```
VAR_IN_OUT
   DcToTcTimeOffset  : ULINT;
   DcToExtTimeOffset : ULINT;
   ExtTime           : T_DCTIME64;
   IntTime           : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
DcToTcTimeOffset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTimeOffset	ULINT	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
ExtTime	T_DCTIME64	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format
IntTime	T_DCTIME64	Interne Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format

Ausgänge

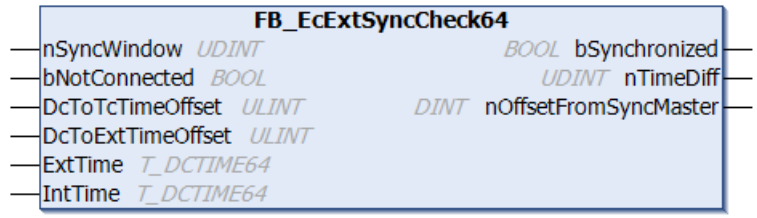
```
VAR_OUTPUT
   nTimeDiff             : UDINT; (*with difference greater than 32 bit timeDiff = 0xffffffff*)
   nOffsetFromSyncMaster : DINT; (*less than 32 bit int Offset = 0x80000000, greater than 32 bit int
Offset = 0x7FFFFFFF*)
END_VAR
```

Name	Typ	Beschreibung
nTimeDiff	UDINT	Wenn die Differenz kleiner als 32 bit ist, dann wird die Zeitdifferenz geliefert. Ist die Differenz größer als 32 bit, dann wird 16#FFFFFFFF geliefert.
nOffsetFromSyncMaster	DINT	Wenn die Differenz größer als 32 bit und der Offset zwischen interner und DC Time kleiner als 32 bit ist, dann wird hier 16#80000000 geliefert. Wenn die Differenz größer als 32 bit und der Offset zwischen interner und DC Time größer als 32 bit ist, dann wird 16#7FFFFFFF geliefert.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.3.9 FB_EcExtSyncCheck64



Der Funktionsbaustein `FB_EcExtSyncCheck64` prüft, ob die interne und die externe Uhr synchron laufen. Siehe Funktionsbaustein FB_EcExtSyncCalcTimeDiff64 [▶ 97].

Eingänge

```
VAR_INPUT
   nSyncWindow    : UDINT;
   bNotConnected  : BOOL;
END_VAR
```

Name	Typ	Beschreibung
nSyncWindow	UDINT	Zeitfenster, innerhalb dessen die interne und die externe Uhr als synchron gelten.
bNotConnected	BOOL	TRUE = Verbindung zur externen Uhr ist unterbrochen.

Ein-/Ausgänge

```
VAR_IN_OUT
   DcToTcTimeOffset  : T_LARGE_INTEGER;
   DcToExtTimeOffset : T_LARGE_INTEGER;
   ExtTime           : T_DCTIME64;
   IntTime           : T_DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
DcToTcTime Offset	T_LARGE_INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTime Offset	T_LARGE_INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
ExtTime	T_DCTIME64	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format
IntTime	T_DCTIME64	Interne Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format

Ausgänge

```
VAR_OUTPUT
   bSynchronized         : BOOL;
   nTimeDiff             : UDINT;
   nOffsetFromSyncMaster : DINT;
END_VAR
```

Name	Typ	Beschreibung
bSynchronized	BOOL	TRUE = externe und interne Uhr laufen synchron
nTimeDiff	UDINT	Aktuelle Zeitdifferenz beider Uhren
nOffsetFrom SyncMaster	DINT	Offset zum Sync Master

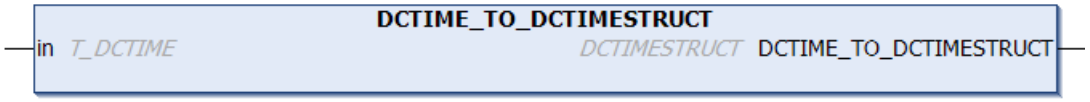
Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4 [veraltet]

10.4.1 [veraltet DCTIME]

10.4.1.1 DCTIME_TO_DCTIMESTRUCT



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion DCTIME64_TO_DCTIMESTRUCT [▶ 87].

Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time"-Variable vom Typ T_DCTIME [▶ 126] in eine strukturierte Variable vom Typ DCTIMESTRUCT [▶ 124].

FUNCTION DCTIME_TO_DCTIMESTRUCT: DCTIMESTRUCT

Eingänge

```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	Die zu konvertierende „Distributed Clock System Time“-Variable.

Beispiel:

```
PROGRAM P_TEST
VAR
    dcStruct : DCTIMESTRUCT;
    dcTime : T_DCTIME;
END_VAR

dcTime := F_GetCurDcTickTime();
dcStruct := DCTIME_TO_DCTIMESTRUCT(dcTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.2 DCTIME_TO_FILETIME

Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion DCTIME64_TO_FILETIME [▶ 113].

Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126] in eine 64 Bit „Windows File Time“-Variable vom Typ T_FILETIME.

FUNCTION DCTIME_TO_FILETIME: T_FILETIME

Eingänge

```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	Die zu konvertierende „Distributed Clock System Time“-Variable

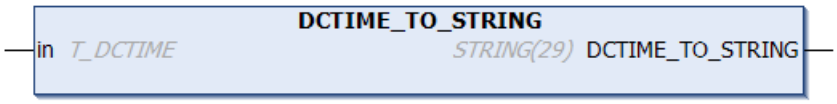
Beispiel:

```
PROGRAM P_TEST
VAR
    ft  : T_FILETIME;
    dct : T_DCTIME;
END_VAR

dct := F_GetCurDcTickTime();
ft  := DCTIME_TO_FILETIME(dct);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.3 DCTIME_TO_STRING



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion DCTIME64_TO_STRING [▶ 88].

Die Funktion konvertiert eine „Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126] in einen String.

Der resultierende String hat nach der Konvertierung das folgende Format: **'YYYY-MM-DD-hh:mm:ss.nnnnnnnnn'**

- YYYY: Jahr;
- MM: Monat;
- DD: Tag;
- hh: Stunde;
- mm: Minute;
- ss: Sekunde;
- nnnnnnnnn: Nanosekunden;

FUNCTION DCTIME_TO_STRING: STRING(29)

Eingänge

```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	Die zu konvertierende „Distributed Clock System Time“-Variable

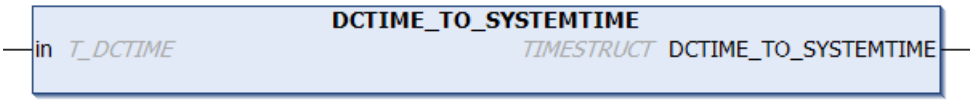
Beispiel:

Siehe in der Beschreibung der Funktion: F_GetCurDcTickTime [▶ 109].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.4 DCTIME_TO_SYSTEMTIME



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion DCTIME64_TO_SYSTEMTIME [▶ 89].

Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126] in eine strukturierte „Windows System Time“-Variable vom Typ TIMESTRUCT.

FUNCTION DCTIME_TO_SYSTEMTIME: TIMESTRUCT

Eingänge

```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	Die zu konvertierende „Distributed Clock System Time“-Variable

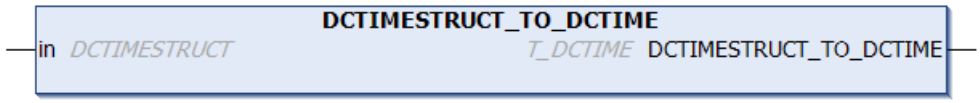
Beispiel:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT;
END_VAR

syst := DCTIME_TO_SYSTEMTIME( F_GetCurDcTickTime() );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.5 DCTIMESTRUCT_TO_DCTIME



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion DCTIMESTRUCT_TO_DCTIME64 [▶ 90].

Die Funktion konvertiert die strukturierte Variable vom Typ DCTIMESTRUCT [▶ 124] in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126].
Die Strukturkomponente wDayOfWeek wird bei der Konvertierung ignoriert. Die wYear-Strukturkomponente muss einen Wert größer oder gleich 2000 oder kleiner 2584 haben. Bei unzulässigen Werten der Strukturkomponenten liefert die Funktion den Wert Null zurück.

FUNCTION DCTIMESTRUCT_TO_DCTIME: T_DCTIME

Eingänge

```
VAR_INPUT
    in : DCTIMESTRUCT;
END_VAR
```

Name	Typ	Beschreibung
in	DCTIMESTRUCT	Die zu konvertierende strukturierte Variable

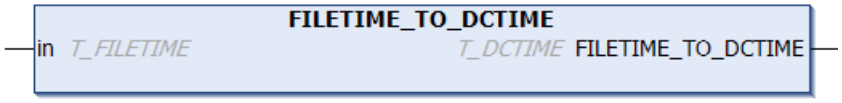
Beispiel:

```
PROGRAM P_TEST
VAR
    dcStruct : DCTIMESTRUCT := ( wYear := 2008, wMonth := 3, wDay := 13,
                                 wHour := 1, wMinute := 2, wSecond :=3,
                                 wMilliseconds := 123, wMicroseconds := 456, wNanoseconds := 789 );
    dc64 : T_DCTIME;
END_VAR

dc64 := DCTIMESTRUCT_TO_DCTIME( dcStruct );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.6 FILETIME_TO_DCTIME



Veraltete Funktion

Die Funktion ist veraltet. Verwenden Sie stattdessen die Funktion FILETIME_TO_DCTIME64 [▶ 113].

Die Funktion konvertiert eine 64-Bit-„Windows File Time"-Variable vom Typ T_FILETIME in eine 64-Bit-„Distributed Clock System Time"-Variable vom Typ T_DCTIME [▶ 126]. Bei einem Konvertierungsfehler liefert die Funktion den Wert Null zurück.

FUNCTION FILETIME_TO_DCTIME: T_DCTIME

Eingänge

```
VAR_INPUT
    in : T_FILETIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_FILETIME	Die zu konvertierende „Windows File Time“-Variable

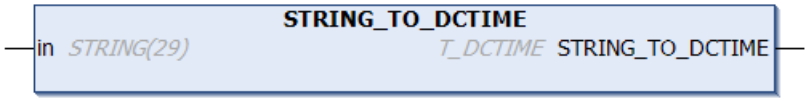
Beispiel:

```
PROGRAM P_TEST
VAR
    fbSysFileTime : GETSYSTEMTIME;
    ft : T_FILETIME;
    dct : T_DCTIME;
END_VAR

fbSysFileTime(timeLoDW=>ft.dwLowDateTime, timeHiDW=>ft.dwHighDateTime);
dct := FILETIME_TO_DCTIME(ft);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.7 STRING_TO_DCTIME



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion STRING_TO_DCTIME64 [▶ 91].

Die Funktion konvertiert einen String in die „Distributed Clock System Time“-Variable vom Typ T_DCTIME [▶ 126].

FUNCTION STRING_TO_DCTIME: T_DCTIME

Eingänge

```
VAR_INPUT
    in : STRING(29);
END_VAR
```

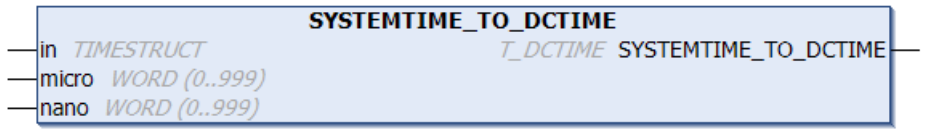
Name	Typ	Beschreibung
in	STRING	Der zu konvertierende String Der String muss folgendes Format haben: **'YYYY-MM-DD-hh:mm:ss.nnnnnnnnn'** • YYYY: Jahr; • MM: Monat; • DD: Tag; • hh: Stunde; • mm: Minute; • ss: Sekunde; • nnnnnnnnn: Nanosekunden;

Siehe in der Beschreibung der Funktion: F_GetCurDcTickTime [▶ 109].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.8 SYSTEMTIME_TO_DCTIME



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion SYSTEMTIME_TO_DCTIME64 [▶ 91].

Die Funktion konvertiert die strukturierte „Windows System Time“-Variable vom Typ TIMESTRUCT in eine 64-Bit-„Distributed Clock System Time"-Variable vom Typ T_DCTIME [▶ 126]. Bei einem Konvertierungsfehler liefert die Funktion den Wert Null zurück.

FUNCTION SYSTEMTIME_TO_DCTIME: T_DCTIME

Eingänge

```
VAR_INPUT
    in    : TIMESTRUCT;
    micro : WORD(0..999); (* Microseconds: 0..999 *)
    nano  : WORD(0..999); (* Nanoseconds: 0..999 *)
END_VAR
```

Name	Typ	Beschreibung
in	TIMESTRUCT	Die zu konvertierende "Windows System Time"-Variable
Micro	WORD	Microsekunden
nano	WORD	Nanosekunden

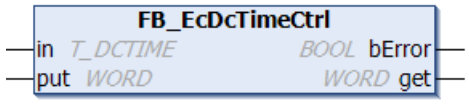
Beispiel:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT := ( wYear := 2009, wMonth := 9, wDay := 16, wHour := 12, wMinute := 22, wSeco
nd := 44, wMilliseconds := 123 );
END_VAR

dct := SYSTEMTIME_TO_DCTIME( syst, 456, 789 );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.1.9 FB_EcDcTimeCtrl



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen den Funktionsbaustein FB_EcDcTimeCtrl64 [▶ 92].

Mit diesem Funktionsbaustein können die einzelnen Komponenten wie Jahr, Monat, Tag usw. einer 64-Bit-„Distributed Clock System Time“-Variablen vom Typ T_DCTIME [▶ 126] gelesen werden. Der Funktionsbaustein besitzt mehrere A_GetXYZ-Aktionen. Nach dem Aufruf der gewünschten Aktion steht der Wert der XYZ-Komponente in der get-Ausgangsvariablen zur Verfügung. Die put-Eingangsvariable wird zurzeit nicht benutzt.

Der Funktionsbaustein besitzt zurzeit folgende Aktionen:

- A_GetYear;
- A_GetMonth;
- A_GetDay;
- A_GetDayOfWeek;
- A_GetHour;
- A_GetMinute;
- A_GetSecond;
- A_GetMilli;
- A_GetMicro;
- A_GetNano;

Eingänge

```
VAR_INPUT
    put : WORD;
END_VAR
```

Name	Typ	Beschreibung
put	WORD	Eingangsparameter (zurzeit nicht benutzt)

Ein/-Ausgänge

```
VAR_IN_OUT
    in : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_DCTIME	TwinCAT „Distributed Clock System Time”-Variable

Ausgänge

```
VAR_OUTPUT
    bError : BOOL;
    get    : WORD;
END_VAR
```

Name	Typ	Beschreibung
bError	BOOL	Dieser Ausgang wird gesetzt, wenn ein Fehler beim Aktionsaufruf aufgetreten ist.
get	WORD	Ausgangsparameter (Jahr, Monat, Tag usw.)

Beispiel für eine Implementierung in ST:

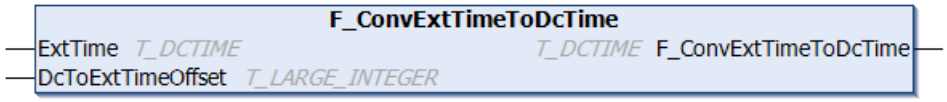
```
PROGRAM P_TEST
VAR
    dcStruct   : DCTIMESTRUCT;
    dcTime     : T_DCTIME;
    fbCtrl     : FB_EcDcTimeCtrl;

    wYear      : WORD;
    wMonth     : WORD;
    wDay       : WORD;
    wDayOfWeek : WORD;
    wHour      : WORD;
    wMinute    : WORD;
    wSecond    : WORD;
    wMilli     : WORD;
    wMicro     : WORD;
    wNano      : WORD;
END_VAR

dcTime := F_GetCurDcTickTime();

fbCtrl.A_GetYear( in := dcTime, get => wYear );
fbCtrl.A_GetMonth( in := dcTime, get => wMonth );
fbCtrl.A_GetDay( in := dcTime, get => wDay );
fbCtrl.A_GetDayOfWeek( in := dcTime, get => wDayOfWeek );
fbCtrl.A_GetHour( in := dcTime, get => wHour );
fbCtrl.A_GetMinute( in := dcTime, get => wMinute );
fbCtrl.A_GetSecond( in := dcTime, get => wSecond );
fbCtrl.A_GetMilli( in := dcTime, get => wMilli );
fbCtrl.A_GetMicro( in := dcTime, get => wMicro );
fbCtrl.A_GetNano( in := dcTime, get => wNano );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2 [veraltet DCTIME und T_LARGE_INTEGER]

10.4.2.1 F_ConvExtTimeToDcTime



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_ConvExtTimeToDcTime64 [▶ 94].

Die Funktion `F_ConvExtTimeToDcTime` konvertiert eine externe Zeit in die TwinCAT „Distributed Clock“-Systemzeit.

FUNCTION F_ConvExtTimeToDcTime: T_DCTIME

Eingänge

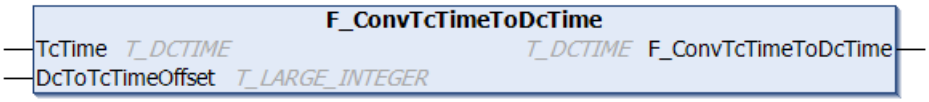
```
VAR_INPUT
   ExtTime           : T_DCTIME;
   DcToExtTimeOffset : T_LARGE_INTEGER;
END_VAR
```

Name	Typ	Beschreibung
ExtTime	T_DCTIME	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeitformat
DcToExtTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen der TwinCAT „Distributed Clock“-Systemzeit und einer externer Zeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.2 F_ConvTcTimeToDcTime



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_ConvTcTimeToDcTime64 [▶ 94].

Die Funktion F_ConvTcTimeToDcTime konvertiert die TwinCAT Systemzeit in die TwinCAT „Distributed Clock“-Systemzeit.

FUNCTION F_ConvTcTimeToDcTime: T_DCTIME

Eingänge

```
VAR_INPUT
   TcTime           : T_DCTIME;
   DcToTcTimeOffset : T_LARGE_INTEGER;
END_VAR
```

Name	Typ	Beschreibung
TcTime	T_DCTIME	TwinCAT Systemzeit im TwinCAT „Distributed Clock“-Systemzeitformat
DcToTcTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen der TwinCAT „Distributed Clock“-Systemzeit und der TwinCAT Systemzeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.3 F_ConvTcTimeToExtTime

Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_ConvTcTimeToExtTime64 [▶ 95].

Die Funktion `F_ConcTcTimeToExtTime` konvertiert die TwinCAT „Distributed Clock“-Systemzeit in eine externe Zeit.

FUNCTION F_ConvTcTimeToExtTime: T_DCTIME

Eingänge

```
VAR_INPUT
   TcTime            :  T_DCTIME;
   DcToTcTimeOffset  :  T_LARGE_INTEGER;
   DcToExtTimeOffset :  T_LARGE_INTEGER;
END_VAR
```

Name	Typ	Beschreibung
TcTime	T_DCTIME	TwinCAT Systemzeit im „Distributed Clock“-Format
DcToTcTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.4 F_GetActualDcTime

Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_GetActualDcTime64 [▶ 95].

Diese Funktion liefert die aktuelle Zeit im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME [▶ 126]).

FUNCTION F_GetActualDcTime: T_DCTIME

Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

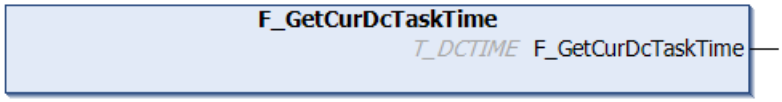
Beispiel:

```
PROGRAM MAIN
VAR
    actDC : T_DCTIME;
    sAct : STRING;
END_VAR

actDC := F_GetActualDcTime();
sAct := DCTIME_TO_STRING( actDC );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.5 F_GetCurDcTaskTime



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_GetCurDcTaskTime64 [▶ 96].

Diese Funktion liefert die Startzeit der Task (Zeitpunkt zu dem die Task starten sollte) im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME [▶ 126]). Die Funktion liefert immer die Startzeit der Task in der sie aufgerufen wurde.

FUNCTION F_GetCurDcTaskTime: T_DCTIME

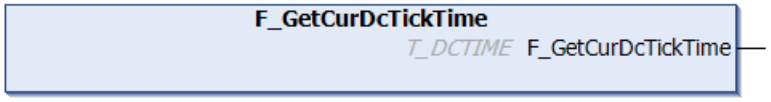
Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.6 F_GetCurDcTickTime



Veraltete Funktion

Die Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_GetCurDcTickTime64 [▶ 96].

Diese Funktion liefert die Zeit des aktuellen (letzten) Ticks im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME [▶ 126]).

FUNCTION F_GetCurDcTickTime: T_DCTIME

Eingänge

```
VAR_INPUT
(*none*)
END_VAR
```

Beispiel:

```
PROGRAM MAIN
VAR
    tDC : T_DCTIME;
    sDC : STRING;
    tDCBack : T_DCTIME;

    sDCZero : STRING;(* DCTIME = zero time starts on 01.01.2000 *)
    tDCBackFromZero : T_DCTIME;

    tDCFromString : T_DCTIME;
    sDCBackFromString : STRING;
END_VAR

tDC := F_GetCurDcTickTime();
sDC := DCTIME_TO_STRING( tDC );
tDCBack := STRING_TO_DCTIME( sDC );

sDCZero := DCTIME_TO_STRING( ULARGE_INTEGER(0, 0) );
tDCBackFromZero := STRING_TO_DCTIME( sDCZero );

tDCFromString := STRING_TO_DCTIME( '2007-03-09-11:31:09.223456789' );
sDCBackFromString := DCTIME_TO_STRING( tDCFromString );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.7 F_GetCurExtTime

Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die Funktion F_GetCurExtTime64 [▶ 97].

Die Funktion liefert die externe Zeit im TwinCAT „Distributed Clock“-Systemzeitformat (T_DCTIME [▶ 126]).

FUNCTION F_GetCurExtTime: T_DCTIME

Eingänge

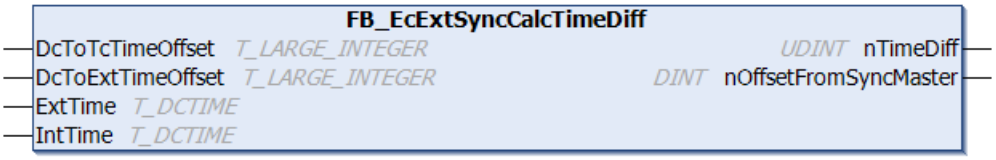
```
VAR_INPUT
   DcToExtTimeOffset : T_LARGE_INTEGER;
   DcToTcTimeOffset  : T_LARGE_INTEGER;
END_VAR
```

Name	Typ	Beschreibung
DcToExtTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
DcToTcTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.8 FB_EcExtSyncCalcTimeDiff



Veralteter Funktionsbaustein

Dieser Funktionsbaustein ist veraltet. Verwenden Sie stattdessen den Funktionsbaustein FB_EcExtSyncCalcTimeDiff64 [▶ 97].

Der Funktionsbaustein `FB_EcExtSyncCalcTimeDiff` berechnet die Differenz zwischen externer und interner Zeit unter Berücksichtigung der Zeitoffsets.

Ein-/Ausgänge

```
VAR_IN_OUT
   DcToTcTimeOffset   : T_LARGE_INTEGER;
   DcToExtTimeOffset  : T_LARGE_INTEGER;
   ExtTime            : T_DCTIME;
   IntTime            : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
DcToTcTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTime Offset	T_LARGE_ INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
ExtTime	T_DCTIME	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format
IntTime	T_DCTIME	Interne Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format

Ausgänge

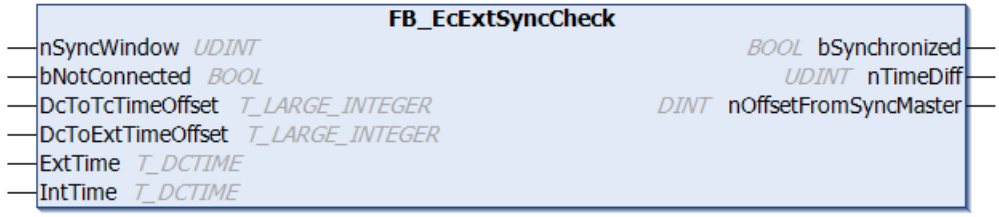
```
VAR_OUTPUT
   nTimeDiff             :  UDINT; (*with difference greater than 32 bit timeDiff = 0xffffffff*)
   nOffsetFromSyncMaster :  DINT; (*less than 32 bit int Offset = 0x80000000, greater than 32 bit
int Offset = 0x7FFFFFFF*)
END_VAR
```

Name	Typ	Beschreibung
nTimeDiff	UDINT	Wenn die Differenz kleiner als 32 bit ist, dann wird die Zeitdifferenz geliefert. Ist die Differenz größer als 32 bit, dann wird 16#FFFFFFFF geliefert.
nOffsetFrom SyncMaster	DINT	Wenn die Differenz größer als 32 bit und der Offset zwischen interner und DC Time kleiner als 32 bit ist, dann wird hier 16#80000000 geliefert. Wenn die Differenz größer als 32 bit und der Offset zwischen interner und DC Time größer als 32 bit ist, dann wird 16#7FFFFFFF geliefert.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.2.9 FB_EcExtSyncCheck



Veralteter Funktionsbaustein

Dieser Funktionsbaustein ist veraltet. Verwenden Sie stattdessen den Funktionsbaustein FB_EcExtSyncCheck64 [▶ 98].

Der Funktionsbaustein `FB_EcExtSyncCheck` prüft, ob die interne und die externe Uhr synchron laufen. Siehe Funktionsbaustein FB_EcExtSyncCalcTimeDiff [▶ 111].

Eingänge

```
VAR_INPUT
   nSyncWindow    : UDINT;
   bNotConnected  : BOOL;
END_VAR
```

Name	Typ	Beschreibung
nSyncWindow	UDINT	Zeitfenster, innerhalb dessen die interne und die externe Uhr als synchron gelten.
bNotConnected	BOOL	TRUE = Verbindung zur externen Uhr ist unterbrochen.

Ein-/Ausgänge

```
VAR_IN_OUT
   DcToTcTimeOffset  : T_LARGE_INTEGER;
   DcToExtTimeOffset : T_LARGE_INTEGER;
   ExtTime           : T_DCTIME;
   IntTime           : T_DCTIME;
END_VAR
```

Name	Typ	Beschreibung
DcToTcTimeOffset	T_LARGE_INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und TwinCAT Systemzeit
DcToExtTimeOffset	T_LARGE_INTEGER	Zeitoffset zwischen TwinCAT „Distributed Clock“-Systemzeit und externer Zeit
ExtTime	T_DCTIME	Externe Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format
IntTime	T_DCTIME	Interne Zeit im TwinCAT „Distributed Clock“-Systemzeit-Format

Ausgänge

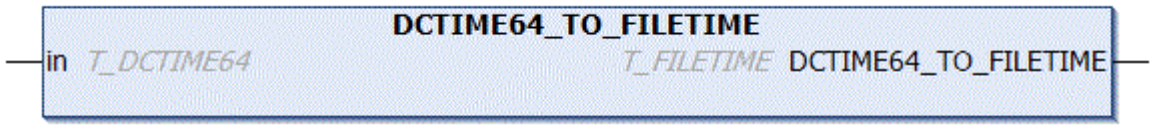
```
VAR_OUTPUT
   bSynchronized         : BOOL;
   nTimeDiff             : UDINT;
   nOffsetFromSyncMaster : DINT;
END_VAR
```

Name	Typ	Beschreibung
bSynchronized	BOOL	TRUE = externe und interne Uhr laufen synchron
nTimeDiff	UDINT	Aktuelle Zeitdifferenz beider Uhren
nOffsetFrom SyncMaster	DINT	Offset zum Sync Master

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.3 DCTIME64_TO_FILETIME



Die Funktion konvertiert eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125] in eine 64-Bit-„Windows File Time“-Variable vom Typ T_FILETIME.

FUNCTION DCTIME64_TO_FILETIME: T_FILETIME

Eingänge

```
VAR_INPUT
    in : T_DCTIME64:
END_VAR;
```

Name	Typ	Beschreibung
in	T_DCTIME64	Die zu konvertierende „Distributed Clock System Time”-Variable

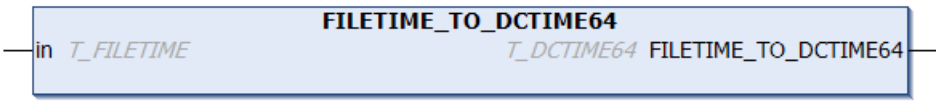
Beispiel:

```
PROGRAM P_TEST
VAR
    ft  : T_FILETIME;
    dct : T_DCTIME64;
END_VAR

dct := F_GetCurDcTickTime64();
ft  := DCTIME64_TO_FILETIME(dct);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

10.4.4 FILETIME_TO_DCTIME64



Die Funktion konvertiert die 64-Bit-„Windows File Time“-Variable vom Typ T_FILETIME in eine 64-Bit-„Distributed Clock System Time“-Variable vom Typ T_DCTIME64 [▶ 125]. Bei einem Konvertierungsfehler liefert die Funktion den Wert Null zurück.

FUNCTION FILETIME_TO_DCTIME64: T_DCTIME64

Eingänge

```
VAR_INPUT
    in : T_FILETIME;
END_VAR
```

Name	Typ	Beschreibung
in	T_FILETIME	Die zu konvertierende "Windows File Time“-Variable

Beispiel:

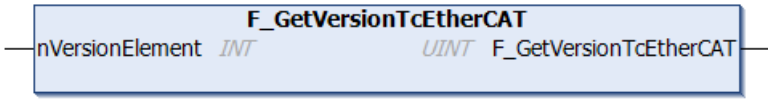
```
PROGRAM P_TEST
VAR
    fbSysFileTime : GETSYSTEMTIME;
    ft : T_FILETIME;
    dct : T_DCTIME64;
END_VAR

fbSysFileTime(timeLoDW=>ft.dwLowDateTime, timeHiDW=>ft.dwHighDateTime);
dct := FILETIME_TO_DCTIME64(ft);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

11 [veraltete Funktionen]

11.1 F_GetVersionTcEtherCAT



Veraltete Funktion

Diese Funktion ist veraltet. Verwenden Sie stattdessen die globale Strukturinstanz stLibVersion_Tc2_EtherCAT

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

FUNCTION F_GetVersionTcEtherCAT : UINT

Eingänge

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

Name	Typ	Beschreibung
nVersionElement	INT	Versionselement, das gelesen werden soll. Mögliche Parameter: • 1 : major number; • 2 : minor number; • 3 : revision number;

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12 Datentypen

12.1 E_EcAdressingType

Die Adressierung in EtherCAT ist entweder positionsabhängig (eAdressingType_AutoInc), basiert auf einer fixen, konfigurierten Adresse (eAdressingType_Fixed) oder betrifft alle Slaves (eAdressingType_Broadcast).

```
TYPE E_EcAdressingType :
(
eAdressingType_AutoInc:=1, (* Adress slave by it's position. (adp = 1-
position, 1.Slave = 0, 2.Slave = 0xffff(-1) etc) *)
                (* EtherCAT commands: APRD, APWR, APRW *)
eAdressingType_Fixed, (* Adress slave by configured ethercat slave address (adp = configured address
) *)
                (* EtherCAT commands: FPRD, FPWR, FPRW *)
eAdressingType_Broadcast    (* Adress all slaves. *)
                (* EtherCAT commands: BRD, BWR, BRW *)
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.2 E_EcFoeMode

Zugriffsmode für das "File access over EtherCAT"-Mailboxprotokoll.

```
TYPE E_EcFoeMode :
(
    eFoeMode_Write := 1,
    eFoeMode_Read
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.3 E_EcMbxProtType

Unterstütze EtherCAT-Mailboxprotokoll-Typen.

```
TYPE E_EcMbxProtType:
(
    eEcMbxProt_CoE := 3,(* CANopen over EtherCAT *)
    eEcMbxProt_FoE := 4,(* File over EtherCAT *)
    eEcMbxProt_SoE := 5 (* Servo Drive Profile over EtherCAT *)
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.4 ST_EcCrcError

Struktur mit den CRC-Error-Zählern der einzelnen Ports (A,B und C) eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcCrcError :
STRUCT
    portA : UDINT;
```

```
    portB : UDINT;
    portC : UDINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
portA	UDINT	CRC-Error Zähler des PortA
portB	UDINT	CRC-Error Zähler des PortB
portC	UDINT	CRC-Error Zähler des PortC

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.5 ST_EcCrcErrorEx

Struktur mit den CRC-Error-Zählern der einzelnen Ports (A,B,C und D) eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcCrcErrorEx :
STRUCT
    portA : UDINT;
    portB : UDINT;
    portC : UDINT;
    portD : UDINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
portA	UDINT	CRC-Error Zähler des PortA
portB	UDINT	CRC-Error Zähler des PortB
portC	UDINT	CRC-Error Zähler des PortC
portD	UDINT	CRC-Error Zähler des PortD

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.6 ST_EcLastProtErrInfo

Die Struktur ST_EcLastProtErrInfo enthält zusätzliche Fehlerinformationen zum zuletzt aufgetretenen "EtherCAT-Mailboxprotokollfehler".

```
TYPE ST_EcSlaveState:
STRUCT
    ownAddr : ST_AmsAddr;
    orgAddr : ST_AmsAddr;
    errCode : UDINT;
    binDesc : ARRAY[0..MAX_STRING_LENGTH] OF BYTE;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
ownAddr	ST_AmsAddr	Eigene AMS-Adresse (Adresse des Kommunikationsteilnehmers der die Fehlerinformationen abfragt)
orgAddr	ST_AmsAddr	AMS-Adresse des Fehlerverursachers (Adresse des Kommunikationsteilnehmers der den Protokollfehler ausgelöst oder verursacht hat)
errCode	UDINT	Mailboxprotokollfehlernummer [▶ 129] (SoE, CoE, FoE error code)
binDesc	ARRAY[0..MAX_STRING_LENGTH] OF BYTE	Zusätzliche Fehlerinformationen als Binärdaten. Die zusätzliche Fehlerinformation ist gerätespezifisch und kann z.B. einen String oder Binärdaten enthalten.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.7 ST_EcMasterStatistic

```
TYPE ST_EcMasterStatistic :
STRUCT
    nSysTime              : UDINT;
    nCycFrameCnt          : UDINT;
    nCycFrameMissedCnt    : UDINT;
    nQueuedFrameCnt       : UDINT;
    nQueuedFrameMissedCnt : UDINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
nSysTime	UDINT	Systemzeit in µs
nCycFrameCnt	UDINT	Anzahl der zyklischen EtherCAT-Frames
nCycFrameMissedCnt	UDINT	Anzahl der verlorenen zyklischen EtherCAT-Frames
nQueuedFrameCnt	UDINT	Anzahl der azyklischen EtherCAT-Frames
nQueuedFrameMissedCnt	UDINT	Anzahl der verlorenen azyklischen EtherCAT-Frames

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.8 ST_EcSlaveConfigData

Die Struktur `ST_EcSlaveConfigData` enthält die EtherCAT-Konfigurationsdaten eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveConfigData:
STRUCT
    nEntries        : WORD;
    nAddr           : WORD;
    sType           : STRING[15];
    sName           : STRING[31];
    nDevType        : DWORD;
    stSlaveIdentity : ST_EcSlaveIdentity;
    nMailboxOutSize : WORD;
    nMailboxInSize  : WORD;
    nLinkStatus     : BYTE;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
nEntries	WORD	Intern verwendet
nAddr	WORD	Adresse eines EtherCAT-Slaves
sType	STRING	EtherCAT-Typ eines Slaves
sName	STRING	Name eines EtherCAT-Slaves
nDevType	DWORD	EtherCAT-Device-Typ eines Slaves
stSlaveIdentity	ST_EcSlaveIdentity	Identity eines EtherCAT-Slaves (siehe ST_EcSlaveIdentity [▶ 119])
nMailboxOutSize	WORD	OutSize der Mailbox eines EtherCAT-Slaves
nMailboxInSize	WORD	InSize der Mailbox eines EtherCAT-Slaves
nLinkStatus	BYTE	Link-Status eines EtherCAT-Slaves

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.9 ST_EcSlaveIdentity

Die Struktur `ST_EcSlaveIdentity` enthält die EtherCAT-Identitätsdaten eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveIdentity :
STRUCT
    vendorId    : UDINT;
    productCode : UDINT;
    revisionNo  : UDINT;
    serialNo    : UDINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
vendorId	UDINT	Vendor-ID des Slave-Gerätes
productCode	UDINT	Produkt-Code des Slave-Gerätes
revisionNo	UDINT	Zeigt die Revision-Nummer des Slave-Gerätes an.
serialNo	UDINT	Zeigt die Serien-Nummer des Slave-Gerät an.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.10 ST_EcSlaveScannedData

Die Struktur `ST_EcSlaveScannedData` enthält die EtherCAT-Konfigurationsdaten eines gescannten EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveConfigData:
STRUCT
    nEntries        : WORD;
    nAddr           : WORD;
    stSlaveIdentity : ST_EcSlaveIdentity;
    ndlStatusReg    : WORD;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
nEntries	WORD	Intern verwendet
nAddr	WORD	Adresse eines EtherCAT-Slaves
stSlaveIdentity	ST_EcSlaveIdentity	Identity eines EtherCAT-Slaves (siehe ST_EcSlaveIdentity [▶ 119])
ndlStatusReg	WORD	Link-Status eines EtherCAT-Slaves aus dem ESC Register $0110/0111_{hex}$ bzw. $272/273_{dec}$. Wenn der Slave nicht erreichbar/offline ist, wird der Status 0 angezeigt. Die Zuordnung „PortNummer <=> Buchse/Ebus Kontakt“ ist der jeweiligen Gerätedokumentation zu entnehmen. Wenn nicht anders beschrieben, ist Port0 der linke Ebus-Kontakt einer EL/ES-Klemme bzw. RJ45-Buchse einer EP-Box, Port1 der rechte abgehende Ebus-Kontakt/RJ45-Buchse.

Die Bitbedeutungen lauten:

Bit	Bedeutung
1	internal use
2	internal use
3	internal use
4	physical link on Port 0 0: no link, 1: Link detected
5	physical link on Port 1 0: no link, 1: Link detected
6	physical link on Port 2 0: no link, 1: Link detected
7	physical link on Port 3 0: no link, 1: Link detected
8	Loop Port 0 0: Open, 1:Closed
9	Communication on Port 0 0:no stable communication, 1:Communication established
10	Loop Port 1 0: Open, 1:Closed
11	Communication on Port 1 0:no stable communication, 1:Communication established
12	Loop Port 2 0: Open, 1:Closed
13	Communication on Port 2 0:no stable communication, 1:Communication established
14	Loop Port 3 0: Open, 1:Closed
15	Communication on Port 3 0:no stable communication, 1:Communication established

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.11 ST_EcSlaveState

Die Struktur `ST_EcSlaveState` enthält den EtherCAT-Status und den Link-Status eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveState:
STRUCT
    deviceState :BYTE;
    linkState   :BYTE;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
deviceState	BYTE	EtherCAT-Status eines Slaves (Siehe deviceState)
linkState	BYTE	Link-Status eines EtherCAT-Slaves (Siehe linkState)

deviceState

EtherCAT-Status eines Slaves. Der Status kann einen der folgenden Werte annehmen:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_INIT	0x01	Init-Zustand
EC_DEVICE_STATE_PREOP	0x02	Pre-Operational Zustand
EC_DEVICE_STATE_BOOTSTRAP	0x03	Bootstrap Zustand
EC_DEVICE_STATE_SAFEOP	0x04	Safe-Operational Zustand
EC_DEVICE_STATE_OP	0x08	Operational-Zustand

Zusätzlich können noch folgende Bits gesetzt sein:

Konstante	Wert	Beschreibung
EC_DEVICE_STATE_ERROR	0x10	Statemachine-Fehler im EtherCAT-Slave
EC_DEVICE_STATE_INVALID_VPRS	0x20	Ungültige VendorId, Product-Code, Revisionsnummer oder Seriennummer
EC_DEVICE_STATE_INITCMD_ERROR	0x40	Fehler beim Senden von Initialsierungs-Kommandos aufgetreten.
EC_DEVICE_STATE_DISABLED	0x80	Slave ist deaktiviert

linkState

Link-Status eines EtherCAT-Slaves. Der Link-Status kann eine Oder-Verknüpfung folgender Bits sein:

Konstante	Wert	Beschreibung
EC_LINK_STATE_OK	0x00	
EC_LINK_STATE_NOT_PRESENT	0x01	Keine EtherCAT-Kommunikation mit dem EtherCAT-Slave
EC_LINK_STATE_LINK_WITHOUT_COMM	0x02	Fehler an Port X(festgelegt durch EC_LINK_STATE_PORT_A/B/C/D). Der Port hat einen Link, aber keine Kommunikation über diesen Port ist möglich.
EC_LINK_STATE_MISSING_LINK	0x04	Fehlender Link an Port X (festgelegt durch EC_LINK_STATE_PORT_A/B/C/D).
EC_LINK_STATE_ADDITIONAL_LINK	0x08	Zusätzlicher Link an Port X(festgelegt durch EC_LINK_STATE_PORT_A/B/C/D).
EC_LINK_STATE_PORT_A	0x10	Port 0
EC_LINK_STATE_PORT_B	0x20	Port 1
EC_LINK_STATE_PORT_C	0x40	Port 2
EC_LINK_STATE_PORT_D	0x80	Port 3

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.12 ST_EcSlaveStateBits

Die Struktur `ST_EcSlaveStateBits` enthält den EtherCAT-Status und den Link-Status eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveStateBits:
STRUCT
    bInit            : BOOL;
    bPreop           : BOOL;
    bBootStrap       : BOOL;
    bSafeOp          : BOOL;
    bOp              : BOOL;
    bError           : BOOL;
    bInvVPRS         : BOOL;
    bInitCmdError    : BOOL;
    bLinkNotPresent  : BOOL;
    bLinkWithoutComm : BOOL;
    bLinkMissing     : BOOL;
    bAdditionalLink  : BOOL;
    bPortA           : BOOL;
    bPortB           : BOOL;
    bPortC           : BOOL;
    bPortD           : BOOL;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.13 ST_EcSlaveStateBitsEx

Die Struktur `ST_EcSlaveStateBitsEx` enthält den EtherCAT-Status und den Link-Status eines EtherCAT-Slave-Gerätes.

```
TYPE ST_EcSlaveStateBitsEx:
STRUCT
    bInit            : BOOL;
    bPreop           : BOOL;
    bBootStrap       : BOOL;
    bSafeOp          : BOOL;
    bOp              : BOOL;
    bError           : BOOL;
    bInvVPRS         : BOOL;
    bInitCmdError    : BOOL;
    bDisabled        : BOOL;
    bLinkNotPresent  : BOOL;
    bLinkWithoutComm : BOOL;
    bLinkMissing     : BOOL;
    bAdditionalLink  : BOOL;
    bPortA           : BOOL;
    bPortB           : BOOL;
    bPortC           : BOOL;
    bPortD           : BOOL;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.14 ST_PortAddr

Die Struktur `ST_PortAddr` enthält EtherCAT-Topologie-Informationen EtherCAT-Slave-Gerätes. EtherCAT-Slave-Geräte haben typischer Weise 2 bis 4 Ports.

```
TYPE ST_PortAddr:
STRUCT
    portA : UINT;
    portB : UINT;
    portC : UINT;
    portD : UINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
portA	UINT	Adresse des vorherigen EtherCAT-Slaves an Port A des aktuellen EtherCAT-Slaves
portB	UINT	Adresse des optional nachfolgenden EtherCAT-Slaves an Port B des aktuellen EtherCAT-Slaves
portC	UINT	Adresse des optional nachfolgenden EtherCAT-Slaves an Port C des aktuellen EtherCAT-Slaves
portD	UINT	Adresse des optional nachfolgenden EtherCAT-Slaves an Port D des aktuellen EtherCAT-Slaves

12.15 ST_TopologyDataEx

Die Struktur `ST_TopologyDataEx` enthält Informationen zur EtherCAT-Topologie und zu Hot Connect Gruppen.

```
TYPE ST_TopologyDataEx:
STRUCT
   nOwnPhysicalAddr  :  UINT;
   nOwnAutoIncAddr   :  UINT;
   stPhysicalAddr    :  ST_PortAddr;
   stAutoIncAddr     :  ST_PortAddr;
   aReserved1        :  ARRAY [0..3] OF UDINT;
   nStatusBits       :  DWORD;
   nHCSlaveCountCfg  :  UINT; (*nStatusBits.0 = TRUE: DcSupprt;.1 = TRUE: DC64Supprt; .2=TRUE: Slave
Present following hot connect info requires runtime >= TC 2.11 R3 B2246 nStatusBits.3 = TRUE: HotCon
nectGroupStart; .4 = HotConnectSlave; .5 = TRUE: HotCOnnectInvalidB; .6 = TRUE: HotConnectInvalidC;
.7 = TRUE: HotConnectInvalidD*)
   nHCSlaveCountAct  :  UINT;
```

```
   aReserved2        :  ARRAY [0..4] OF UDINT;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
nOwnPhysicalAddr	UINT	Eigene physische EtherCAT-Adresse des EtherCAT-Slave-Gerätes
nOwnAutoIncAddr	UINT	Eigene autoinkrement EtherCAT-Adresse des EtherCAT-Slave-Gerätes
stPhysicalAddr	ST_PortAddr	Physische Adressinformationen der EtherCAT-Slave-Geräte an Port A…D
stAutoIncAddr	ST_PortAddr	Autoinkrement Adressinformationen der EtherCAT-Slave-Geräte an Port A…D
aReserved1	ARRAY [0..3] OF UDINT	Reserviert
nStatusBits	DWORD	nStatusBits.0 = TRUE: Distributed Clock werden unterstützt nStatusBits.1 = TRUE: Distributed Clock werden unterstützt (64 bit) nStatusBits.2 = TRUE: Slave ist vorhanden nStatusBits.3 = TRUE: Slave ist Startknoten einer Hot Connect Gruppe nStatusBits.4 = TRUE: Slave ist in einer Hot Connect Gruppe nStatusBits.5 = TRUE: Hot Connect ist ungültig an Port B nStatusBits.6 = TRUE: Hot Connect ist ungültig an Port C nStatusBits.7 = TRUE: Hot Connect ist ungültig an Port D
nHCSlaveCountCfg	UINT	Konfigurierte Anzahl der Hot Connect Gruppenteilnehmer
nHCSlaveCountAct	UINT	Gefundene Anzahl an Hot Connect Gruppenteilnehmern
aReserved2	ARRAY [0..4] OF UDINT	Reserviert

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.16 DCTIMESTRUCT

Strukturierter TwinCAT „Distributed Clock System Time"-Zeitformat. Die kleinste Einheit ist eine Nanosekunde. Dieser Datentyp repräsentiert die **Anzahl der Nanosekunden seit dem 01.01.2000 (GMT)**.

```
TYPE DCTIMESTRUCT :
STRUCT
    wYear         : WORD;
    wMonth        : WORD;
    wDayOfWeek    : WORD;
    wDay          : WORD;
    wHour         : WORD;
    wMinute       : WORD;
    wSecond       : WORD;
    wMilliseconds : WORD;
    wMicroseconds : WORD;
    wNanoseconds  : WORD;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
wYear	WORD	Jahr: 2000 ~ 2584
wMonth	WORD	Monat: 1 ~ 12 (Januar = 1, Februar = 2 usw.)
wDayOfWeek	WORD	Wochentag: 0 ~ 6 (Sonntag = 0, Montag = 1 usw.)
wDay	WORD	Tag des Monats: 1 ~ 31
wHour	WORD	Stunde: 0 ~ 23
wMinute	WORD	Minute: 0 ~ 59
wSecond	WORD	Sekunde: 0 ~ 59
wMilliseconds	WORD	Millisekunde: 0 ~ 999
wMicroseconds	WORD	Microsekunde: 0 ~ 999
wNanoseconds	WORD	Nanosekunde: 0 ~ 999

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.17 T_DCTIME32

32-Bit-TwinCAT-„Distributed Clock System Time”-Zeitformat. Die kleinste Einheit ist eine Nanosekunde.

Diese 32-Bit-„DC System Time“ wird aus der vollständigen absoluten 64-Bit-„DC System Time“ (T_DCTIME [▶ 126]) gebildet, indem nur die niederwertigsten 32 bit verwendet werden. Hierdurch geht die Eigenschaft einer absoluten eindeutigen Zeit verloren und es wird angenommen, dass diese 32-Bit-Zeit nur im zeitlichen Nahbereich von ± 2.147 Sekunden um die aktuelle Systemzeit verwendet wird, da sie nur hier eindeutig ist. Hiervon kann in vielen Anwendungsfällen ausgegangen werden.
Wenn diese Annahme verletzt wird, dann kann es zu Fehlern bei der Interpretation und der Weiterverarbeitung dieser Zeit kommen.

```
TYPE T_DCTIME32 : UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.18 T_DCTIME64

Der Datentyp `T_DCTIME64` repräsentiert die „Distributed Clock System Time“ (kurz „DC Time“ genannt) als linearen 64-Bit-Integer-Wert. Die Zeit wird in Nanosekunden seit dem 1.1.2000 UTC dargestellt. Die kleinste Einheit ist eine Nanosekunde.

```
TYPE T_DCTIME64 : ULINT;
END_TYPE
```

Nützliche "Distributed Clock System Time"-Konstanten	Beschreibung
EC_DCTIME_DELTA_OFFSET64	Anzahl der 100-Nanosekunden-Ticks zwischen dem 01.01.1601 und 01.01.2000. Es ist die Differenz zwischen der "Windows File Time" und der „Distributed Clock System Time".
EC_DCTIME_DATEDELTA_OFFSET	Anzahl der vergangenen Tage seit dem Jahr Null bis zum 1 Januar 2000
EC_DCTIME_TICKSPERMSEC64	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Millisekunde
EC_DCTIME_TICKSPERSEC64	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Sekunde
EC_DCTIME_TICKSPERDAY64	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Tag

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.19 T_DCTIME

Veralteter Datentyp

Dieser Datentyp ist veraltet. Verwenden Sie stattdessen den Datentyp T_DCTIME64 [▶ 125].

Der Datentyp `T_DCTIME` repräsentiert die „Distributed Clock System Time“ (kurz „DC Time“ genannt) als linearen 64-Bit-Integer-Wert. Die Zeit wird in Nanosekunden seit dem 1.1.2000 UTC dargestellt.
Der Datentyp wird als zwei 32-Bit-DWORD-Variablen dargestellt, sodass er in der SPS einfach verarbeitet werden kann. Operationen (Addition und Subtraktion von Zeiten) können mit ui64 Funktionen aus der Bibliothek Tc2_Utilities ausgeführt werden.

```
TYPE T_DCTIME : T_ULARGE_INTEGER;
END_TYPE
```

Nützliche "Distributed Clock System Time"-Konstanten	Beschreibung
EC_DCTIME_DELTA_OFFSET	Anzahl der 100-Nanosekunden-Ticks zwischen dem 01.01.1601 und 01.01.2000. Es ist die Differenz zwischen der "Windows File Time" und der „Distributed Clock System Time".
EC_DCTIME_DATEDELTA_OFFSET	Anzahl der vergangenen Tage seit dem Jahr Null bis zum 1 Januar 2000
EC_DCTIME_TICKSPERMSEC	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Millisekunde
EC_DCTIME_TICKSPERSEC	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Sekunde
EC_DCTIME_TICKSPERDAY	Anzahl der „Distributed Clock System Time"-Nanosekunden pro Tag

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

12.20 T_HFoe

"File access over EtherCAT"-Handle. Das Handle muss vor der Benutzung einmalig mit dem Funktionsbaustein FB_EcFoeOpen [▶ 65] initialisiert werden. Die Variablen dieses strukturierten Typs dürfen nicht direkt beschrieben werden.

```
TYPE T_HFoe :
STRUCT
    sNetID  : T_AmsNetId  := '';
    nPort   : T_AmsPort   := 0;
    handle  : UDINT       := 0;
    eMode   : E_EcFoeMode := eFoeMode_Write;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

13 Konstanten

13.1 Globale Konstanten

VAR_GLOBAL CONSTANT

```
EC_AMSPORT_MASTER :UINT :=16#FFFF;
EC_MAX_SLAVES     :UINT :=16#FFFF;

(*ethercat commands*)
EC_CMD_TYPE_APRD :BYTE :=1;
EC_CMD_TYPE_APWR :BYTE :=2;
EC_CMD_TYPE_APRW :BYTE :=3;
EC_CMD_TYPE_FPRD :BYTE :=4;
EC_CMD_TYPE_FPWR :BYTE :=5;
EC_CMD_TYPE_FPRW :BYTE :=6;
EC_CMD_TYPE_BRD  :BYTE :=7;
EC_CMD_TYPE_BWR  :BYTE :=8;
EC_CMD_TYPE_BRW  :BYTE :=9;
EC_CMD_TYPE_LRD  :BYTE :=10;
EC_CMD_TYPE_LWR  :BYTE :=11;
EC_CMD_TYPE_LRW  :BYTE :=12;

(* Device states *)
EC_DEVICE_STATE_MASK :BYTE :=16#0F;
EC_DEVICE_STATE_INIT :BYTE :=16#01;
EC_DEVICE_STATE_PREOP :BYTE :=16#02;
EC_DEVICE_STATE_BOOTSTRAP :BYTE :=16#03;
EC_DEVICE_STATE_SAFEOP :BYTE :=16#04;
EC_DEVICE_STATE_OP :BYTE :=16#08;
EC_DEVICE_STATE_ERROR :BYTE :=16#10;
EC_DEVICE_STATE_INVALID_VPRS :BYTE :=16#20;
EC_DEVICE_STATE_INITCMD_ERROR :BYTE :=16#40;

(* Link states *)
EC_LINK_STATE_OK :BYTE :=16#00;
EC_LINK_STATE_NOT_PRESENT :BYTE :=16#01;
EC_LINK_STATE_LINK_WITHOUT_COMM :BYTE :=16#02;
EC_LINK_STATE_MISSING_LINK :BYTE :=16#04;
EC_LINK_STATE_ADDITIONAL_LINK :BYTE :=16#08;
EC_LINK_STATE_PORT_A :BYTE :=16#10;
EC_LINK_STATE_PORT_B :BYTE :=16#20;
EC_LINK_STATE_PORT_C :BYTE :=16#40;
EC_LINK_STATE_PORT_D :BYTE :=16#80;


(* Device/Link state IG/IO *)
EC_ADS_IGRP_MASTER_STATEMACHINE :UDINT :=16#00000003;
EC_ADS_IOFFS_MASTER_CURSTATE :UDINT :=16#00000100;
EC_ADS_IOFFS_MASTER_REQSTATE :UDINT :=16#00000101;
EC_ADS_IOFFS_MASTER_INTERNALSTATE :UDINT :=16#00000102;

EC_ADS_IGRP_MASTER_COUNT_SLAVE :UDINT :=16#00000006;
EC_ADS_IOFFS_MASTER_COUNT_SLAVE :UDINT :=16#00000000;
EC_ADS_IOFFS_MASTER_COUNT_PORT :UDINT :=16#00000001;
EC_ADS_IOFFS_MASTER_COUNT_ROUTER :UDINT :=16#00000002;

EC_ADS_IGRP_MASTER_SLAVE_ADDRESSES :UDINT :=16#00000007;
EC_ADS_IGRP_MASTER_SENDCMD :UDINT :=16#00000008;
EC_ADS_IGRP_SLAVE_STATEMACHINE :UDINT :=16#00000009;
EC_ADS_IGRP_MASTER_SLAVE_IDENTITY :UDINT :=16#00000011;
EC_ADS_IGRP_MASTER_SLAVE_CRC :UDINT :=16#00000012;
EC_ADS_IGRP_MASTER_SLAVE_ABNORMAL_STATE_CHANGES :UDINT :=16#00000013;
EC_ADS_IGRP_MASTER_SLAVE_SETPRESENT_CHANGES :UDINT :=16#00000016;
EC_ADS_IGRP_MASTER_DEVICESTATE :UDINT :=16#00000045;
EC_ADS_IGRP_MASTER_COUNT_FRAME :UDINT :=16#00000048;

(* SoE IG/IO *)
EC_ADS_IGRP_ECAT_SOE :UDINT :=16#0000F420;
EC_ADS_IGRP_ECAT_SOE_LASTERROR :UDINT :=16#0000F421;

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;

(* FoE IG/IO *)
EC_ADS_IGRP_FOE_FOPENREAD :UDINT :=16#0000F401;
EC_ADS_IGRP_FOE_FOPENWRITE :UDINT :=16#0000F402;
EC_ADS_IGRP_FOE_FCLOSE :UDINT :=16#0000F403;
EC_ADS_IGRP_FOE_FREAD :UDINT :=16#0000F404;
EC_ADS_IGRP_FOE_FWRITE :UDINT :=16#0000F405;
EC_ADS_IGRP_FOE_PROGRESSINFO :UDINT :=16#0000F406;
EC_ADS_IGRP_FOE_LASTERROR :UDINT :=16#0000F407;

(* CoE IG/IO *)
EC_ADS_IGRP_CANOPEN_SDO :UDINT :=16#0000F302;
EC_ADS_IGRP_CANOPEN_SDO_LASTERROR :UDINT :=16#0000F303;

EC_DCTIME_DATEDELTA_OFFSET : DWORD := 730120; (* Number of past days since year zero until 1 January
 2000 *)
EC_DCTIME_DELTA_OFFSET : T_ULARGE_INTEGER := ( dwHighPart := 16#01BF53EB, dwLowPart := 16#256D4000 )
; (* Number of 100ns ticks between 1.1.1601 and 1.1.2000 *)
EC_DCTIME_TICKSPERMSEC : T_ULARGE_INTEGER := ( dwHighPart := 16#00000000, dwLowPart := 16#000F4240);
(* Number of nanosecond ticks per millisecond *)
EC_DCTIME_TICKSPERSEC : T_ULARGE_INTEGER := ( dwHighPart := 16#00000000, dwLowPart := 16#3B9ACA00);
(* Number of nanosecond ticks per second *)
EC_DCTIME_TICKSPERDAY : T_ULARGE_INTEGER := ( dwHighPart := 16#00004E94, dwLowPart := 16#914F0000);
(* Number of nanosecond ticks per day *)

EC_DCTIME_DELTA_OFFSET64 : ULINT := ULINT#16#01BF53EB_256D4000;
(* Number of 100ns ticks between 1.1.1601 and 1.1.2000 *)
EC_DCTIME_TICKSPERMSEC64 : ULINT := ULINT#16#00000000_000F4240;
(* Number of nanosecond ticks per millisecond *)
EC_DCTIME_TICKSPERSEC64 : ULINT := ULINT#16#00000000_3B9ACA00;
(* Number of nanosecond ticks per second *)
EC_DCTIME_TICKSPERDAY64 : ULINT := ULINT#16#00004E94_914F0000;
(* Number of nanosecond ticks per day *)

bSeqReadDrvAttrAndValue : BOOL := FALSE;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

13.2 Bibliotheksversion

Alle Bibliotheken haben eine bestimmte Version. Diese Version ist u.a. im SPS-Bibliotheks-Repository zu sehen. Eine globale Konstante enthält die Information über die Bibliotheksversion:

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_EtherCAT : ST_LibVersion;
END_VAR
```

stLibVersion_Tc2_EtherCAT: Versionsinformation der Tc2_EtherCAT-Bibliothek (Typ: ST_LibVersion)

Um zu sehen, ob die Version, die Sie haben auch die Version ist, die Sie brauchen, benutzen Sie die Funktion F_CmpLibVersion (definiert in der Tc2_System-Bibliothek).

Alle anderen Möglichkeiten Bibliotheksversionen zu vergleichen, die Sie von TwinCAT 2 kennen, sind veraltet!

13.3 Mailboxprotokoll Fehlercodes

VAR_GLOBAL CONSTANT

```
(* FoE mailbox protocol error codes *)
EC_FOE_PROTERR_NOTDEFINED      : UDINT := 0;
EC_FOE_PROTERR_NOTFOUND        : UDINT := 1;
```

```
EC_FOE_PROTERR_ACCESS          : UDINT := 2;
EC_FOE_PROTERR_DISKFULL        : UDINT := 3;
EC_FOE_PROTERR_ILLEAGAL        : UDINT := 4;
EC_FOE_PROTERR_PACKENO         : UDINT := 5;
EC_FOE_PROTERR_EXISTS          : UDINT := 6;
EC_FOE_PROTERR_NOUSER          : UDINT := 7;
EC_FOE_PROTERR_BOOTSTRAPONLY   : UDINT := 8;
EC_FOE_PROTERR_NOTINBOOTSTRAP  : UDINT := 9;
EC_FOE_PROTERR_INVALIDPASSWORD : UDINT := 10;

(* CoE mailbox protocol error codes *)
EC_COE_PROTERR_TOGGLE : UDINT := 16#05030000; (* Toggle bit not alternated. *)
EC_COE_PROTERR_TIMEOUT : UDINT := 16#05040000; (* SDO protocol timed out. *)
EC_COE_PROTERR_CCS_SCS : UDINT := 16#05040001; (* Client/
server command specifier not valid or unknown. *)
EC_COE_PROTERR_BLK_SIZE : UDINT := 16#05040002; (* Invalid block size (block mode only). *)
EC_COE_PROTERR_SEQNO : UDINT := 16#05040003; (* Invalid sequence number (block mode only). *)
EC_COE_PROTERR_CRC : UDINT := 16#05040004; (* CRC error (block mode only). *)
EC_COE_PROTERR_MEMORY : UDINT := 16#05040005; (* Out of memory. *)
EC_COE_PROTERR_ACCESS : UDINT := 16#06010000; (* Unsupported access to an object. *)
EC_COE_PROTERR_WRITEONLY : UDINT := 16#06010001; (* Attempt to read a write only object. *)
EC_COE_PROTERR_READONLY : UDINT := 16#06010002; (* Attempt to write a read only object. *)
EC_COE_PROTERR_INDEX : UDINT := 16#06020000; (* Object does not exist in the object dictionary. *)
EC_COE_PROTERR_PDO_MAP : UDINT := 16#06040041; (* Object cannot be mapped to the PDO. *)
EC_COE_PROTERR_PDO_LEN : UDINT := 16#06040042; (* The number and length of the objects to be mapped
would exceed PDO length. *)
EC_COE_PROTERR_P_INCOMP : UDINT := 16#06040043; (* General parameter incompatibility reason. *)
EC_COE_PROTERR_I_INCOMP : UDINT := 16#06040047; (* General internal incompatibility in the device. *
)
EC_COE_PROTERR_HARDWARE : UDINT := 16#06060000; (* Access failed due to an hardware error. *)
EC_COE_PROTERR_DATA_SIZE : UDINT := 16#06070010; (* Data type does not match, length of service para
meter does not match *)
EC_COE_PROTERR_DATA_SIZE1 : UDINT := 16#06070012; (* Data type does not match, length of service par
ameter too high *)
EC_COE_PROTERR_DATA_SIZE2 : UDINT := 16#06070013; (* Data type does not match, length of service par
ameter too low *)
EC_COE_PROTERR_OFFSET : UDINT := 16#06090011; (* Sub-index does not exist. *)
EC_COE_PROTERR_DATA_RANGE : UDINT := 16#06090030; (* Value range of parameter exceeded (only for wri
te access). *)
EC_COE_PROTERR_DATA_RANGE1 : UDINT := 16#06090031; (* Value of parameter written too high. *)
EC_COE_PROTERR_DATA_RANGE2 : UDINT := 16#06090032; (* Value of parameter written too low. *)
EC_COE_PROTERR_MINMAX : UDINT := 16#06090036; (* Maximum value is less than minimum value. *)
EC_COE_PROTERR_GENERAL : UDINT := 16#08000000; (* general error *)
EC_COE_PROTERR_TRANSFER : UDINT := 16#08000020; (* Data cannot be transferred or stored to the appli
cation. *)
EC_COE_PROTERR_TRANSFER1 : UDINT := 16#08000021; (* Data cannot be transferred or stored to the appl
ication because of local control. *)
EC_COE_PROTERR_TRANSFER2 : UDINT := 16#08000022; (* Data cannot be transferred or stored to the appl
ication because of the present device state. *)
EC_COE_PROTERR_DICTIONARY : UDINT := 16#08000023; (* Object dictionary dynamic generation fails or n
o object dictionary is present (e.g. object dictionary is generated from file and generation fails b
ecause of an file error). *)
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS- Bibliotheken
TwinCAT v3.1.0	PC oder CX (x86, x64, ARM)	Tc2_EtherCAT

14 Beispiel

Beispielprojekt und Beispielkonfiguration für Diagnose

Siehe https://infosys.beckhoff.com/content/1031/tcplclib_tc2_ethercat/Resources/2364613387.zip

Mehr Informationen:
www.beckhoff.com/te1000

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