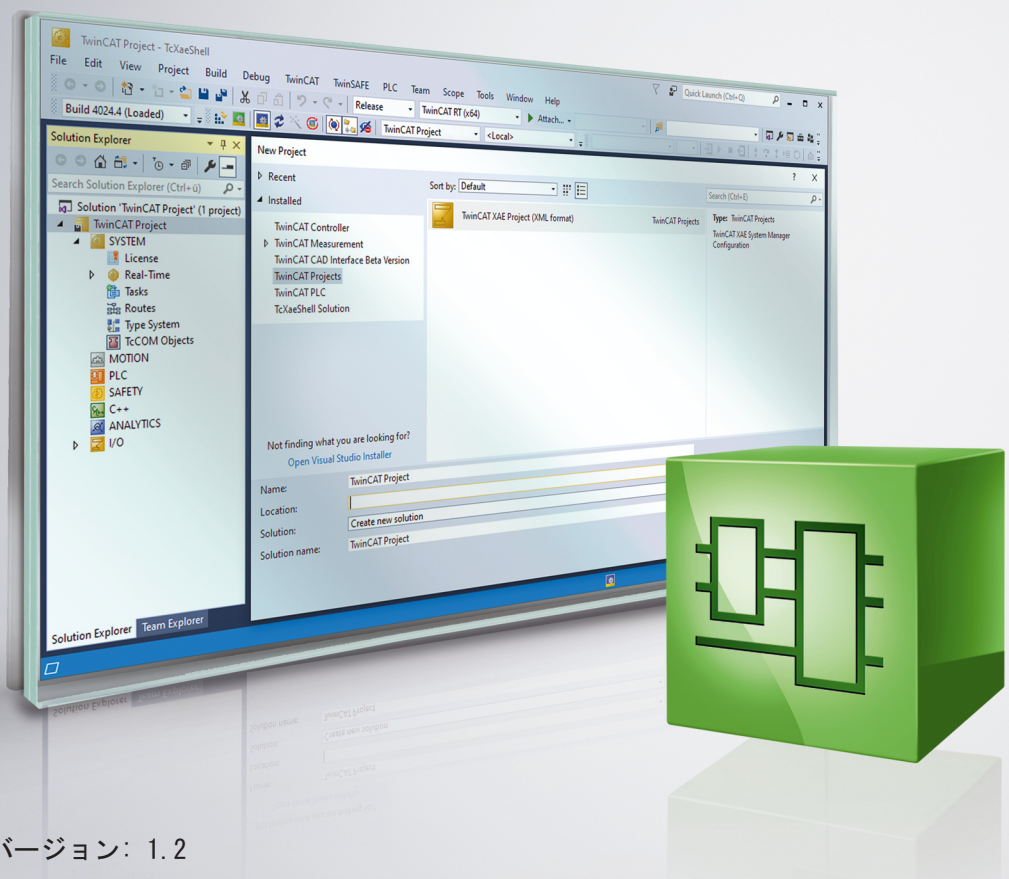
BECKHOFF New Automation Technology

取扱説明書 | JA

TE1000

TwinCAT 3 | PLC Lib: Tc2_EtherCAT



2021-10-07 | バージョン: 1.2

目次

1 序文

1.1 取扱説明書に関する注記

この説明は対応する国内規格を熟知した、トレーニングを受けた制御、オートメーションエンジニアリングの専門技術者のみの使用を対象としています。
コンポーネントのインストールとコミッショニングの際には、取扱説明書および以下の注意事項と説明に従うことが重要です。
技術者には各設置およびコミッショニングのそれぞれの時点で、発行された取扱説明書を使用する義務があります。

本製品を使用するうえでの責任者は、本製品の用途および使用方法が、関連するすべての法律、法規、ガイドラインおよび規格を含む、安全に関するすべての要件を満たしていることを確認してください。

免責事項

この取扱説明書の記載内容は、 一般的な製品説明および性能を記載したものであり、場合により記載通りに動作しないことがあります。
製品の情報・仕様は予告なく変更されます。
この説明書に記載されているデータ、図および説明に基づいて、すでに納品されている製品の変更を要求することはできません。

商標

Beckhoff®、TwinCAT®、TwinCAT/BSD®、TC/BSD®、EtherCAT®、EtherCAT G®、EtherCAT G10®、EtherCAT P®、Safety over EtherCAT®、TwinSAFE®、XFC®、XTS®およびXPlanar®は、Beckhoff Automation GmbHの登録商標です。
この取扱説明書で使用されているその他の名称は商標である可能性があり、第三者が独自の目的のために使用すると所有者の権利を侵害する可能性があります。

特許出願

EtherCAT Technologyについては、欧州特許 EP1590927、EP1789857、
EP1456722およびEP2137893、ドイツ特許DE102015105702
に記載されていますが、これらに限定されるものではありません。

EtherCAT®は、Beckhoff Automation GmbH（ドイツ)によりライセンスを受けた登録商標および特許技術です。

著作権

1.2 安全に関する指示事項

安全に関する注意事項

この取扱説明書に記載された安全に関する指示や注意事項はよくお読みになり、必ず指示に従ってください。

納入仕様

すべての製品は、用途に適した特定のハードウェア構成およびソフトウェア構成を有する状態で供給されます。ハードウェアまたはソフトウェアに取扱説明書に記載されている以外の変更を加えることは許可されていません。許可されていない変更を加えると、Beckhoff Automation GmbH & Co. KGの保証の対象外となります。

使用者の資格

この説明書は関連する国内法規を熟知した、制御およびオートメーションエンジニアリングの専門家の使用を目的としています。

安全記号の説明

この取扱説明書では、安全に関する指示や注意事項とともに以下の安全記号を使用します。安全に関する指示事項はよくお読みになり、必ず指示に従ってください。

⚠ 危険

重大な人的傷害の危険

この記号が付いた安全に関する注意事項に従わないと、人命および健康に直ちに危害を及ぼします。

⚠ 警告

人的傷害の危険

この記号が付いた安全に関する注意事項に従わないと、人命および健康に危険を及ぼします。

⚠ 注意

人的傷害の恐れ

この記号が付いた安全に関する注意事項に従わないと、人命および健康に危険を及ぼす恐れがあります。

注記

物的損害と環境汚染

この記号が付いた安全に関する注意事項に従わないと、物的損害と環境汚染をもたらす恐れがあります。

ヒントまたはアドバイス

この記号が示す情報により、さらに理解が深まります。

2 概要

PLCライブラリTc2_EtherCATには、サービスを実行するファンクションブロック、またはEtherCATマスタデバイスやそのスレーブデバイスの機能が含まれています。

診断用のサンプルプロジェクトとサンプルコンフィグレーション

TC3_IEC_MinECATDiag (Resources/zip/2364613387.zip)をご参照ください。

3 CoEインターフェイス

3.1 FB_EcCoeSdoRead

ファンクションブロックFB_EcCoeSdoReadは、SDO (Service Data Object)アクセスを使用してEtherCATスレーブのオブジェクトディクショナリからデータをリードします。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、リードするオブジェクトを選択できます。サブエレメントを含む完全なパラメータにアクセスするには、アクセスファンクションブロックFB_EcCoeSdoReadEx [▶ 11]を使用する必要があります。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SDO Uploadコマンドを送信するEtherCATスレーブの固定アドレス。

nSubIndex: リードするオブジェクトのサブインデックス。

nIndex: リードするオブジェクトのインデックス。

pDstBuf: 受信バッファのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

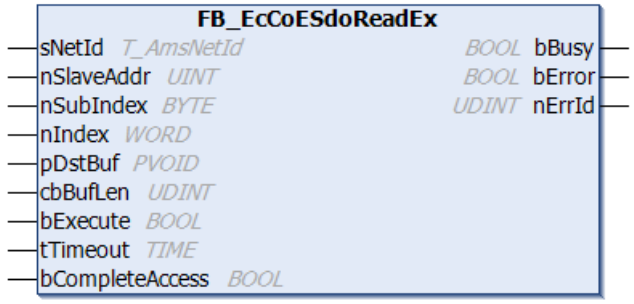
STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

3.2 FB_EcCoeSdoReadEx



ファンクションブロックFB_EcCoeSdoReadExは、SDO (Service Data Object)アクセスを使用してEtherCATスレーブのオブジェクトディクショナリからデータをリードします。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、リードするオブジェクトを選択できます。bCompleteAccess := TRUEとすると、サブエレメントのパラメータ全体をリードできます。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SDO Uploadコマンドを送信するEtherCATスレーブの固定アドレス。

nSubIndex: リードするオブジェクトのサブインデックス。

nIndex: リードするオブジェクトのインデックス。

pDstBuf: 受信バッファのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

bCompleteAccess: bCompleteAccessがセットされている場合、パラメータ全体を1回のアクセスでライトできます。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

3.3 FB_EcCoeSdoWrite

ファンクションブロックFB_EcCoeSdoWriteは、EtherCATスレーブのオブジェクトディレクトリからのオブジェクトをSDO Downloadによってライトできるようにします。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、ライトするオブジェクトを選択できます。サブエレメントを含む完全なパラメータにアクセスするには、ファンクションブロックFB_EcCoeSdoWriteEx [▶ 14]を使用する必要があります。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SDOダウンロードコマンドの送信するEtherCATスレーブの固定アドレス。

nSubIndex: ライトするオブジェクトのサブインデックス。

nIndex: ライトするオブジェクトのインデックス。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

cbBufLen: 送信するデータの数(バイト単位)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

STでの実装例:

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

3.4 FB_EcCoeSdoWriteEx

FB_EcCoeSdoWriteExファンクションブロックは、EtherCATスレーブのオブジェクトディレクトリからのオブジェクトをSDO Downloadによってライトできるようにします。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、ライトするオブジェクトを選択できます。bCompleteAccess := TRUEとすると、サブエレメントのパラメータ全体をライトできます。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SDO Downloadコマンドの送信するEtherCATスレーブの固定アドレス。

nSubIndex: ライトするオブジェクトのサブインデックス。

nIndex: ライトするオブジェクトのインデックス。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

cbBufLen: 送信するデータの数(バイト単位)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

bCompleteAccess: bCompleteAccessがセットされている場合、パラメータ全体を1回のアクセスでライトできます。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

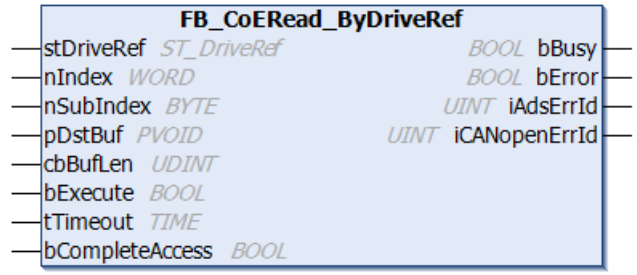
bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

3.5 FB_CoERead_ByDriveRef



ファンクションブロックFB_CoERead_ByDriveRefを使用して、「CANopen over EtherCAT (CoE)」プロトコルによってドライブパラメータをリードできます。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、リードするオブジェクトを選択できます。bCompleteAccess := TRUEとすると、サブエレメントのパラメータ全体をリードできます。

```
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
```

stDriveRef: EtherCATマスタデバイスのAMSネットワークIDとスレーブデバイスのアドレスからなる構造体。ドライブの参照は、System ManagerのPLCに直接リンクできます。このために、ST_PlcDriveRefのインスタンスを使用し、バイト列のNetIDを文字列に変換する必要があります。

nIndex: リードするオブジェクトのインデックス。

nSubIndex: リードするオブジェクトのサブインデックス。

pDstBUF: 受信バッファのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

bCompleteAccess: bCompleteAccessがセットされている場合、パラメータ全体を1回のアクセスでリードできます。

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

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

iAdsErrId: bError出力がセットされている場合、最後に実行されたコマンドのADSエラーコードを返します。

iCANopenErrId: bError出力がセットされている場合、CANopenエラーコードを戻します。

STでの実装例:

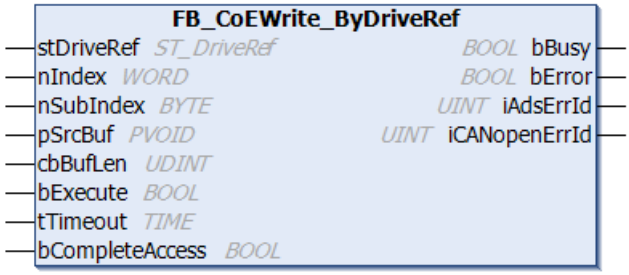
```
PROGRAM MAIN
VAR
    fbCoERead       : FB_CoERead_ByDriveRef;
    stDriveRef      : ST_DriveRef;
    nIndex          : WORD := 16#1018;
    nSubIndex       : BYTE := 1;
    bExecute        : BOOL := TRUE;
    tTimeout        : TIME := T#5S;
    bCompleteAccess : BOOL := TRUE;
    vendorId       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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

3.6 FB_CoEWrite_ByDriveRef



ファンクションブロック FB_CoEWrite_ByDriveRefを使用して、「CANopen over EtherCAT (CoE)」プロトコルに基づくドライブパラメータをライトできます。このためには、スレーブにメールボックスがあることと、「CANopen over EtherCAT」(CoE)プロトコルをサポートしていることが必要です。nSubIndexとnIndexパラメータで、ライトするオブジェクトを選択できます。bCompleteAccess := TRUEとすると、サブエレメントのパラメータ全体をライトできます。

```
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
```

stDriveRef: EtherCATマスタデバイスのAMSネットワークIDとスレーブデバイスのアドレスからなる構造体。ドライブへの参照は、System ManagerのPLCに直接リンクできます。このために、ST_PlcDriveRefのインスタンスを使用し、バイト列のNetIDを文字列に変換する必要があります。

nIndex: ライトするオブジェクトのインデックス。

nSubIndex: ライトするオブジェクトのサブインデックス。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

cbBUFLen: 送信データ用の最大使用可能バッファサイズ(バイト単位)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

bCompleteAccess: bCompleteAccessがセットされている場合、パラメータ全体を1回のアクセスでリードできます。

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

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

iAdsErrId: bError出力がセットされている場合、最後に実行されたコマンドのADSエラーコードを返します。

iCANopenErrId: bError出力がセットされている場合、CANopenエラーコードを戻します。

STでの実装例:

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

4 EtherCATコマンド

4.1 FB_EcPhysicalReadCmd

ファンクションブロックFB_EcPhysicalReadCmdを使用して、EtherCATリードコマンド(FPRD、APRD、BRD)を特定のEtherCATスレーブまたはすべてのEtherCATスレーブに送信できます。PLCはこのコマンドを使用して、レジスタまたはEtherCATスレーブコントローラのDPRAMをリードできます。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

adp: この値は、このコマンドがどのEtherCATスレーブをアドレス指定するのかを決定します。この値の意味は、下記のeTypeで選択されたアドレス指定モードによります。

eType	説明
eAdressingType_Fixed	スレーブは、Configured EtherCATアドレスによってアドレス指定されます。これらのEtherCATアドレスは、ファンクションブロックFB_EcGetAllSlaveAddrによってリードできます。
eAdressingType_AutoInc	スレーブは、リングの中の位置に基づいてアドレス指定されます。最初のデバイスはアドレス0 (adp=0)で、adpはすべてのその後のスレーブに対して1ずつデクリメントします。 1. Slave adp = 0 2. Slave adp = 16#ffff (-1) 3. Slave adp = 16#fffe(-2) 4. Slave adp = 16#fffd(-3) など
eAdressingType_BroadCAST	このコマンドで、すべてのスレーブがアドレス指定されます。adpは0に設定できます。

ado: リードする物理メモリ(DPRAM)またはレジスタ。

len: リードするバイト数。

eType: eTypeの値によって、異なるEtherCATコマンドを送信します。

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

個々のコマンドは、アドレス指定モードによって異なります(adpを参照)。

pDstBuf: 受信バッファのアドレス(ポインタ)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

wkc: ワーキングカウンタは、このコマンドを正常に処理した各EtherCATスレーブがインクリメントします。このコマンドで1つのEtherCATスレーブのみをアドレス指定した場合、この値は1になります。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

4.2 FB_EcPhysicalWriteCmd

ファンクションブロックFB_EcPhysicalWriteCmdを使用して、EtherCATライトコマンド(FPWR、APWR、BWR)を特定のEtherCATスレーブまたはすべてのEtherCATスレーブに送信できます。PLCはこのコマンドを使用して、レジスタまたはEtherCATスレーブコントローラのDPRAMにライトすることができます。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

adp: この値は、このコマンドがどのEtherCATスレーブをアドレス指定するのかを決定します。この値の意味は、下記のeTypeで選択されたアドレス指定モードによります。

eType	説明
eAdressingType_Fixed	スレーブは、Configured EtherCATアドレスによってアドレス指定されます。これらのEtherCATアドレスは、ファンクションブロックFB_EcGetAllSlaveAddrによってリードできます。
eAdressingType_AutoInc	スレーブは、リングの中の位置に基づいてアドレス指定されます。最初のデバイスはアドレス0 (adp=0)で、adpはすべてのその後のスレーブに対して1ずつデクリメントします。 1. Slave adp = 0 2. Slave adp = 16#ffff (-1) 3. Slave adp = 16#fffe(-2) 4. Slave adp = 16#fffd(-3) など
eAdressingType_BroadCAST	このコマンドで、すべてのスレーブがアドレス指定されます。adpは0にセットしてください。

ado: リードする物理メモリ(DPRAM)またはレジスタ。

len: ライトするバイト数。

eType: eTypeの値によって、異なるEtherCATコマンドを送信します。

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

個々のコマンドは、アドレス指定モードによって異なります(adpを参照)。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

wkc: ワーキングカウンタは、このコマンドを正常に処理した各EtherCATスレーブがインクリメントします。このコマンドで1つのEtherCATスレーブのみをアドレス指定した場合、この値は1になります。

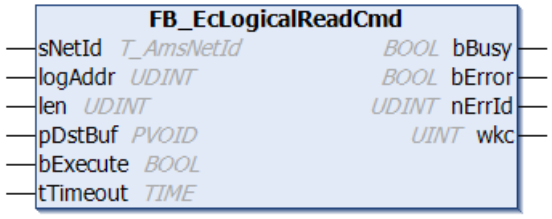
STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

4.3 FB_EcLogicalReadCmd



マスタは、ファンクションブロックFB_EcLogicalReadCmdを使用して論理EtherCATリードコマンド(LRD)を送信します。各スレーブで、ローカルアドレス範囲(DPRAM)をグローバルな論理的アドレス範囲にマッピングします。これにより、このコマンドでは選択した論理アドレス範囲に対してマッピングが設定されているすべてのEtherCATスレーブがアドレス指定されます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

logAddr: 論理アドレス。

len: リードするバイト数。

pDstBuf: 受信バッファのアドレス(ポインタ)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

wkc: ワーキングカウンタは、このコマンドを正常に処理した各EtherCATスレーブがインクリメントします。このコマンドで1つのEtherCATスレーブのみをアドレス指定した場合、この値は1になります。

STでの実装例:

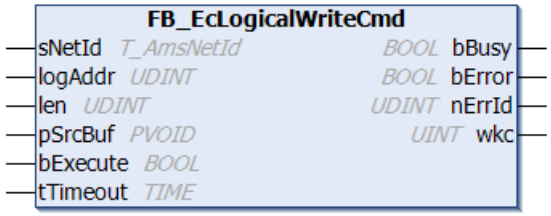
```
PROGRAM Test_LogicalReadCmd
VAR
    fbReadCmd : FB_EcLogicalReadCmd;
    bExecute  : BOOL;
```

```
    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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

4.4 FB_EcLogicalWriteCmd



マスタは、 ファンクションブロックFB_EcLogicalWriteCmdを使用して、 論理EtherCATライトコマンド(LWR)を送信します。各スレーブで、ローカルアドレス範囲(DPRAM)をグローバルな論理的アドレス範囲にマッピングします。これにより、このコマンドでは選択した論理アドレス範囲に対してマッピングが設定されているすべてのEtherCATスレーブがアドレス指定されます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

logAddr: 論理アドレス。

len: ライトするバイト数。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
```

```
    nErrId : UDINT;
    wkc    : UINT;
END_VAR
```

bBusy: この出力はファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

wkc: ワーキングカウンタは、このコマンドを正常に処理した各EtherCATスレーブがインクリメントします。このコマンドで1つのEtherCATスレーブのみをアドレス指定した場合、この値は1になります。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5 EtherCAT診断

5.1 FB_EcGetAllSlaveAbnormalStateChanges

ファンクションブロックFB_EcGetAllSlaveAbnormalStateChangesを使用して、マスタに接続したすべてのスレーブの意図しないEtherCATステートの変更をリードできます。コールが成功すると、パラメータpBufAddrで受信したバッファにすべてのスレーブの意図しないステート変更数がUDINT配列で返ります。意図しないEtherCATステートの変更とは、EtherCATマスタがリクエストしていない変更のことです。たとえば、EtherCATスレーブが自発的にOP状態からSAFEOP状態に切り替わる場合などです。

VAR_INPUT

```
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 lea
st nSlave *4 Bytes *)
    bExecute        :    BOOL; (*Function Block execution is triggered by a rising edge at this inpu
t*)
    tTimeout        :    TIME; (*States the time before the function is cancelled.*)
END_VAR
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pAddrBuf: 個々のスレーブの意図しない状態の変更回数をライトするUDINT配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.2 FB_EcGetAllSlaveAddr

ファンクションブロックFB_EcGetAllSlaveAddr は、マスタに接続されているすべてのスレーブのアドレスをリードできます。コールが成功すると、パラメータpAddrBufに渡されたバッファにすべてのスレーブのアドレスをUINT配列として返します。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pAddrBuf: 個々のスレーブのEtherCATアドレスをライトするUINT配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

STでの実装例:

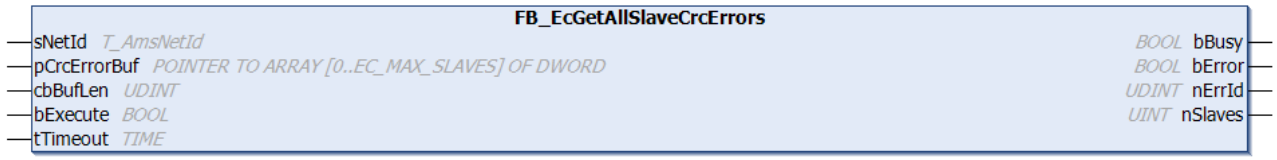
```
PROGRAM TEST_GetAllSlaveAddresses
VAR
    fbGetAllSlaveAddr : FB_EcGetAllSlaveAddr;
```

```
    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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.3 FB_EcGetAllCrcErrors



ファンクションブロックFB_EcGetAllSlaveCrcErrorsは、マスタに接続されているすべてのスレーブのCRCエラーカウンタをリードします。スレーブの個々のポートでのCRCエラーは合計されます。

スレーブの個々のポート(A、BおよびC)のCRCエラーをリードするためには、FB_EcGetSlaveCrcError [▶ 35]ファンクションブロックをコールする必要があります。

スレーブの個々のポート(A、B、CおよびD)のCRCエラーをリードするためには、FB_EcGetSlaveCrcErrorEx [▶ 36]ファンクションブロックをコールする必要があります。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pCrcErrorBuf: CRCエラーカウンタをライトするDWORD配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
```

```
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.4 FB_EcGetAllSlavePresentStateChanges

ファンクションブロックFB_EcGetAllSlavePresentStateChangesを使用して、マスタに接続されているすべてのスレーブの状態「slave is present」から「INIT_NO_COMM」へのEtherCATステートの変更をリードできます。コールが成功すると、パラメータpBufAddrに送信されたバッファは、すべてのスレーブの状態の変更回数をUDINT配列として含みます。状態「slave is present」から「INIT_NO_COMM」へのEtherCATステートの変更は、スレーブへの接続が中断したことを意味します。たとえば、EtherCATケーブルの切断の場合などです。

VAR_INPUT

```
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 INIT_NO_COMM to Present are copied to.*)
    cbBufLen        :    UDINT; (*Size of the buffer pAddrBuf. The size of the buffer must be at lea
st nSlave *4 Bytes *)
```

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pAddrBuf: 個々のスレーブの「INIT_NO_COMM」から「slave is present」への状態の変更回数をライトするUDINT配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.5 FB_EcGetConfSlaves

ファンクションブロックFB_EcGetConfSlavesを使用して、EtherCATマスタのオブジェクトディレクトリから構成されたスレーブのリストをリードできます。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pArrEcConfSlaveInfo: 個々の構成されたスレーブのデータをライトするST_EcSlaveConfigData [▶ 104]型構造体配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

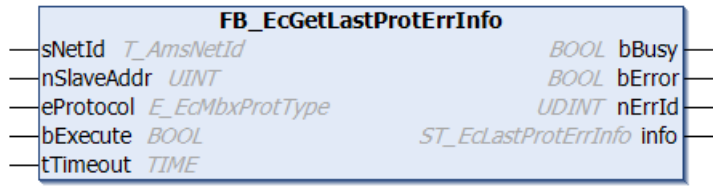
nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: 構成されたスレーブの数を返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.6 FB_EcGetLastProtErrInfo



ファンクションブロックFB_EcGetLastProtErrInfoを使用して、最新のメールボックスのプロトコルエラーに関連した追加エラー情報をリードできます。エラーを解放するメールボックスコマンドが、最後のエラーをその都度リセットします。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: エラー情報をリードするEtherCATスレーブの固定アドレス。

eProtocol: EtherCATのメールボックスのプロトコルタイプ [▶ 102]。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

info: 追加のエラー情報 [▶ 103]の構造体。

STでの例:

bGetの立ち上がりで、最新のメールボックスのプロトコルエラーに関連した追加エラー情報のリードをトリガします。

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.7 FB_EcGetMasterDevState

ファンクションブロックFB_EcGetMasterDevStateを使用して、EtherCATマスタの現在の状態をリードできます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

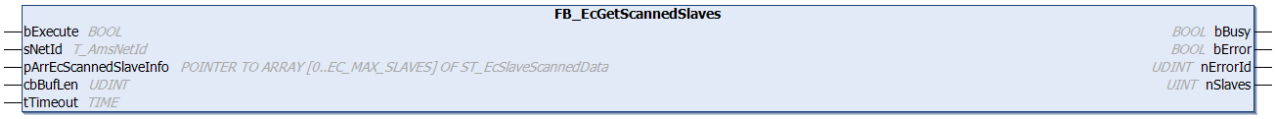
nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

nDevState: マスタデバイスの現在の状態。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.8 FB_EcGetScannedSlaves



ファンクションブロックFB_EcGetScannedSlavesを使用して、EtherCATマスタオブジェクトディレクトリから現在使用可能な(スキャン済みの)スレーブのリストをリードできます。この目的のために、EtherCATスレーブのEEPROMをリードする間に、オンラインスキャンが実行されます。接続されたスレーブの数によっては、スキャンプロセスに時間がかかります。

VAR_INPUT

```
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
```

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pArrEcScannedSlaveInfo: スキャンされた各スレーブのデータをライトするST_EcSlaveScannedData [▶ 105]型構造体配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: スキャンされたスレーブの数を返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.9 FB_EcGetSlaveCount

ファンクションブロックFB_EcGetSlaveCountを使用して、マスタに接続されているスレーブの数を特定できます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
```

```
    nErrId  : UDINT;
    nSlaves : UINT;
END_VAR
```

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

nSlaves: マスタに接続されているスレーブ数。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.10 FB_EcGetSlaveCrcError

ファンクションブロックFB_EcGetSlaveCrcErrorは、スレーブの個々のポート(A、BおよびC)のCRCエラーカウンタをリードします。コールが成功すると、ST_EcCrcError型の出力変数crcErrorには、リクエストしたCRCエラーカウンタが含まれます。

ファンクションブロックFB_EcGetSlaveCrcErrorは、最大3つまでのポート(例: EK1100)をもつスレーブのときのみ使用できます。最大4つまでのポート(例:EK1122)をもつスレーブにはファンクションブロックFB_EcGetSlaveCrcErrorExを使用します。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: CRCエラーカウンタをリードするEtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

crcError: 個々のポートのCRCエラー [▶ 103]カウンタ。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.11 FB_EcGetSlaveCrcErrorEx

ファンクションブロックFB_EcGetSlaveCrcErrorExは、スレーブの個々のポート(A、D、BおよびC)のCRCエラーカウンタをリードします。コールが成功すると、ST_EcCrcErrorEx型の出力変数crcErrorにはリクエストしたCRCエラーカウンタが返ります。

ファンクションブロックFB_EcGetSlaveCrcErrorExは、最大4つまでのポート(例:EK1122)をもつスレーブで使用できます。ファンクションブロックFB_EcGetSlaveCrcErrorは、最大3つまでのポート(例: EK1100)をもつスレーブに対してのみ使用できます。

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列 (T_AmsNetID型)。

nSlaveAddr: CRCエラーカウンタをリードするEtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

CrcError: 個々のポートのCRCエラーカウンタ。(ST_EcCrcErrorEx [▶ 103])

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.12 FB_EcGetSlaveIdentity

ファンクションブロックFB_EcGetSlaveIdentityを使用して、個々のEtherCATスレーブデバイスのCANopen Identityをリードできます。コールが成功すると、ST_EcSlaveIdentity型の出力変数identityには、リクエストされたID情報が含まれます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: EtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

identity: EtherCATデバイスのCANopenのIdentity [▶ 105]。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.13 FB_EcGetSlaveTopologyInfo

ファンクションブロックFB_EcGetSlaveTopolgyInfoを使用して、トポロジ情報を判定できます。

VAR_INPUT

```
VAR_INPUT
    sNetId       :    T_AmsNetId; (*AmsNetId of the EtherCAT master device*)
    pAddrBuf     :    POINTER TO ARRAY [0..EC_MAX_SLAVES] OF ST_TopologyDataEx; (*Contains the addre
ss 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
nSlave * 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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pAddrBuf: トポロジデータを返すST_TopologyDataEx [▶ 109]型構造体配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

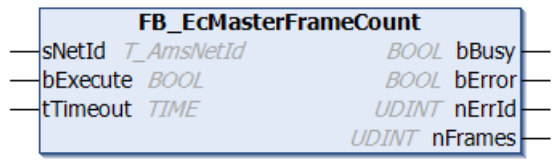
nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.14 FB_EcMasterFrameCount



ファンクションブロックFB_EcMasterFrameCountを使用して、マスタが構成したEtherCATフレームの数を判定できます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

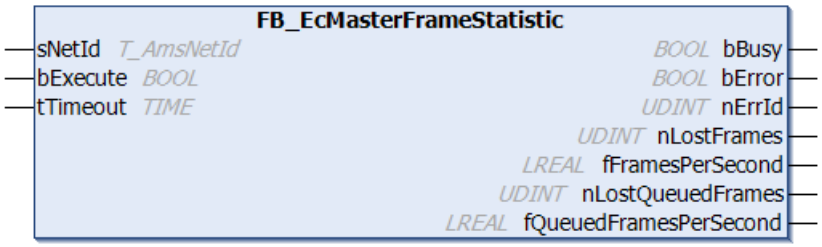
nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

nFrames: EtherCATフレームの数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.15 FB_EcMasterFrameStatistic



ファンクションブロックFB_EcMasterFrameStatisticを使用して、EtherCATマスタのフレーム統計をリードできます。周期および非周期(キュー)フレームには、区別されています。非周期フレームは、初期化またはEtherCATスレーブへのパラメータアクセスのために使用されます。フレームはマスタに戻るのに失敗するか、無効になる場合には、ロストしたとみなされます。
ロストフレーム数(すなわち、ロストまたは無効な周期フレーム)、秒当たりの周期フレーム数、ロストキューフレーム数(すなわち、ロストまたは無効な非周期フレーム)および秒当たりのキューフレーム数は、ファンクションブロックの出力で提供されます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

nLostFrames: ロストまたは無効な周期フレームの現在の数を返します。

fFramesPerSecond: 秒当たりの周期フレームの現在の数を返します。

nLostQueuedFrames: ロストまたは無効なキュー(非周期)フレームの現在の数を返します。

fQueuedFramesPerSecond: 秒当たりのキュー(非周期)フレームの現在の数を返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.16 FB_EcMasterFrameStatisticClearCRC

ファンクションブロックFB_EcMasterFrameStatisticClearCRCを使用して、すべてのEtherCATスレーブのCRCエラーカウンタをクリアできます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.17 FB_EcMasterFrameStatisticClearFrames

ファンクションブロックFB_EcMasterFrameStatisticClearFramesを使用して、ロストフレームカウンタをクリアできます。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.18 FB_EcMasterFrameStatisticClearTxRxErr

ファンクションブロックFB_EcMasterFrameStatisticClearTxRxErrを使用して、ネットワークカードのミニポートドライバのエラーカウンタをクリアできます。

VAR_INPUT

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

sNetId: CPU (PC)のAMSネットワークIDの文字列。(T_AMSNetId型)

nEcMasterDevID: EtherCATマスタのデバイスID

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

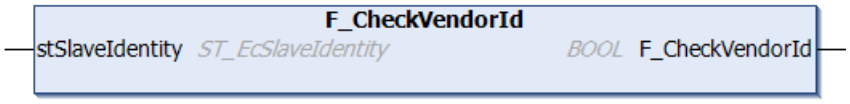
bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

5.19 F_CheckVendorId



ファンクションF_CheckVendorIdはVendorIDがベッコフの場合にTRUEを返し、そうでない場合はFALSEを返します。

VAR_INPUT

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

stSlaveIdentity: FB_EcGetSlaveIdentity [▶ 37]でリードしたスレーブID。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6 EtherCATステートマシン

6.1 FB_EcGetAllSlaveStates

ファンクションブロックFB_EcGetAllSlaveStatesは、マスタに接続されているすべてのスレーブEtherCATステートとリンクステータスをリードします。コールが成功すると、パラメータpStateBufに渡されたバッファはリクエストされたステータス情報をST_EcSlaveState型構造体配列として返します。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

pStateBuf:スレーブステートをライトするST_EcSlaveStates [▶ 106]型構造体配列のアドレス。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。エラー1798 (0x706)は、バッファアドレスのヌルポインタを示します。エラー1797 (0x705)は、バッファサイズが十分でないことを示します。

nSlaves: マスタに接続されているスレーブ数。

STでの実装例:

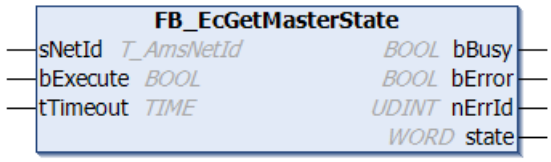
```
PROGRAM TEST_GetAllSlaveStates
VAR
    fbGetAllSlaveStates : FB_EcGetAllSlaveStates;
    sNetId              : T_AmsNetId := '172.16.2.131.2.1';
    bExecute            : BOOL;
    devStates           : ARRAY[0..255] OF ST_EcSlaveState;
```

```
    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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

6.2 FB_EcGetMasterState



ファンクションブロックFB_EcGetMasterStateを使用して、マスタのEtherCATステートをリードできます。コールが成功すると、WORD型のState出力変数にはリクエストされたステータス情報を返します。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

state: マスタの現在のEtherCATステート。設定可能値は下記のようになります。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	マスタはINIT状態です。

定数	値	説明
EC_DEVICE_STATE_PREOP	0x02	マスタはPre-operational状態です。
EC_DEVICE_STATE_SAFEOP	0x04	マスタはSafe-operational状態です。
EC_DEVICE_STATE_OP	0x08	マスタはOperational状態です。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6.3 FB_EcGetSlaveState

ファンクションブロックFB_EcGetSlaveStateは、個々のEtherCATスレーブのEtherCATステートとリンクステータスをリードします。コールが成功すると、ST_EcSlaveState型構造体であるstate出力変数に、リクエストされたステータス情報を返します。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: EtherCATステートをリードするEtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

state: スレーブのEtherCATステートとリンクステータスを含む構造体。(ST_EcSlaveState [▶ 106]型)

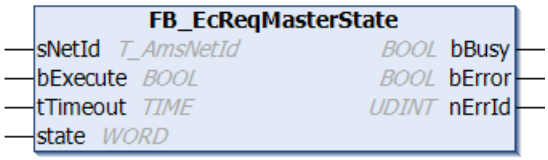
STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6.4 FB_EcReqMasterState



このファンクションブロックを使用して、マスタデバイスのEtherCATステートをリクエストでき、セットできます。リクエストするEtherCATステートは、state変数で送信します。ファンクションブロックはEtherCATステートをリクエスト後に動作を終了します。ファンクションブロックFB_EcSetMasterStateと異なり、新しいステートをセットするまで待機しません。

以下も参照してください。FB_EcSetMasterState [▶ 51]

VAR_INPUT

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    bExecute : BOOL;
```

```
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
    state    : WORD;
END_VAR
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

state: マスタからリクエストするEtherCATステート。設定可能なState値は以下のようになります。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	マスタからのINIT状態のリクエスト
EC_DEVICE_STATE_PREOP	0x02	マスタからのPre-operational状態のリクエスト
EC_DEVICE_STATE_SAFEOP	0x04	マスタからのSafe-operational状態のリクエスト
EC_DEVICE_STATE_OP	0x08	マスタからのOperational状態のリクエスト

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6.5 FB_EcReqSlaveState

このファンクションブロックを使用して、スレーブを指定したEtherCATステートにセットできます。リクエストするEtherCATステートは、state変数で送信します。このファンクションブロックは、ステートを変更するためにコマンドを送信後に動作を終了します。ファンクションブロックFB_EcSetSlaveStateとは異なり、EtherCATスレーブが新しいステートに移行するまで待機しません。

以下も参照してください。FB_EcSetSlaveState [▶ 52]

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: EtherCATステートをセットするEtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

state: スレーブにセットするEtherCATステート。設定可能なState値は以下のようになります。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	スレーブをINIT状態にセット
EC_DEVICE_STATE_PREOP	0x02	スレーブをPre-operational状態にセット
EC_DEVICE_STATE_BOOTSTRAP	0x03	スレーブをBootstrap状態にセット。この状態は、ファームウェアのダウンロードに使用されます。
EC_DEVICE_STATE_SAFEOP	0x04	スレーブをSafe-operational状態にセット
EC_DEVICE_STATE_OP	0x08	スレーブをOperational状態にセット
EC_DEVICE_STATE_ERROR	0x10	ステータスバイトのエラービットがEtherCATスレーブ (state.deviceState & EC_DEVICE_STATE_ERROR = TRUE)にセットされている場合、エラービットはEC_DEVICE_STATE_ERRORをセットすればリセットできます。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6.6 FB_EcSetMasterState

このファンクションブロックを使用して、マスタデバイスのEtherCATステートをリクエストでき、セットできます。リクエストするEtherCATステートは、reqState変数で送信します。ファンクションブロックはEtherCATステートをリクエスト後、ファンクションブロックFB_EcReqMasterStateとは異なり、新しいステートをセットするか、tTimeoutの最大時間が超過するまで動作を継続します。現在の状態がcurrState変数で出力されます。

以下も参照してください。FB_EcReqMasterState [▶ 48]

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

reqState: マスタからリクエストするEtherCATステート。reqStateの設定可能値は以下のようになります。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	マスタからのINIT状態のリクエスト
EC_DEVICE_STATE_PREOP	0x02	マスタからのPre-operational状態のリクエスト
EC_DEVICE_STATE_SAFEOP	0x04	マスタからのSafe-operational状態のリクエスト
EC_DEVICE_STATE_OP	0x08	マスタからのOperational状態のリクエスト

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

currState: マスタの現在のEtherCATステート。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

6.7 FB_EcSetSlaveState

このファンクションブロックを使用して、スレーブを指定したEtherCATステートにセットできます。リクエストするEtherCATステートは、reqState変数で送信します。ファンクションブロックはステートを変更するためにコマンドを送信後、ファンクションブロックFB_EcRegSlaveStateとは異なり、EtherCATスレーブが新しいステートに移行するか、tTimeoutの最大時間が超過するまで動作を継続します。現在の状態がcurrState変数で出力されます。

以下も参照してください。FB_EcReqSlaveState [▶ 50]

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: EtherCATステートをセットするEtherCATスレーブの固定アドレス。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

reqState: スレーブにセットされるEtherCATステート。reqStateの設定可能値は以下のようになります。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	スレーブをINIT状態にセット
EC_DEVICE_STATE_PREOP	0x02	スレーブをPre-operational状態にセット
EC_DEVICE_STATE_BOOTSTRAP	0x03	スレーブをBootstrap状態にセット。この状態は、ファームウェアのダウンロードに使用されます。
EC_DEVICE_STATE_SAFEOP	0x04	スレーブをSafe-operational状態にセット
EC_DEVICE_STATE_OP	0x08	スレーブをOperational状態にセット
EC_DEVICE_STATE_ERROR	0x10	ステータスバイトのエラービットがEtherCATスレーブ (currState.deviceState AND EC_DEVICE_STATE_ERROR = TRUE)にセットされている場合、エラービットはEC_DEVICE_STATE_ERRORをセットするとリセットできます。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

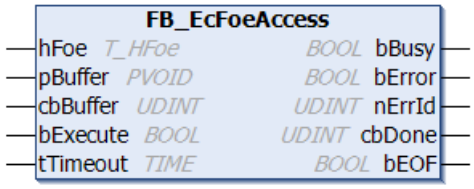
currState: スレーブの現在のEtherCATステート [▶ 106]。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

7 FoEインターフェイス

7.1 FB_EcFoeAccess



このファンクションブロックは、「File access over EtherCAT」メールボックスプロトコルの通信ポート経由でデータをライト、またはリードできます。

VAR_INPUT

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

hFoe: 「File access over EtherCAT」ハンドル [▶ 112]。

pBuffer: データをリードする(リードアクセス)バッファのアドレス、またはライトするデータ(ライトアクセス)が入るバッファのアドレスです。単一の変数、配列、構造体をバッファにすることができ、そのアドレスはADR演算子で指定します。

cbBuffer: ライト、またはリードするデータバイト数。。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

cbDone: 正常にライト、またはリードされた最新のデータバイト数。

bEOF: ファイルの終端。リードアクセス中にファイルの終端に達した場合に、この変数はTRUEになります。ライトアクセスの場合は、この変数は意味がありません。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

7.2 FB_EcFoeClose

ファンクションブロックFB_EcFoeCloseは、「File access over EtherCAT」メールボックスプロトコル用の通信ポートを閉じます。

VAR_INPUT

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

hFoe: 「File access over EtherCAT」ハンドル [▶ 112]。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

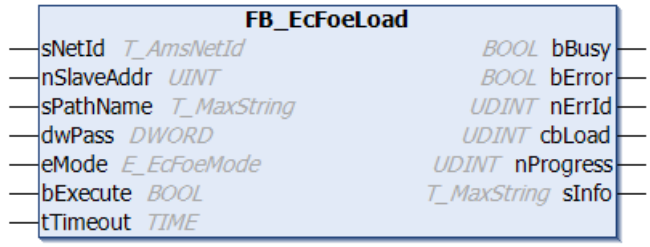
bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

7.3 FB_EcFoeLoad



ファンクションブロックFB_EcFoeLoadを使用して、「File access over EtherCAT」メールボックスプロトコル経由でEtherCATデバイスからファイルをダウンロード、またはEtherCATデバイスへファイルをアップロードできます。

パスは、コンピュータ内のローカルファイルシステムだけを指定できます。ネットワークパスはここでは使用できません。FoEプロトコル経由でファイルをアップロードまたはダウンロードするために、ファンクションブロックは自動的にEtherCATデバイスをBOOTSTRAPモードにリセットします。最後に、ファンクションブロックはデバイスをオリジナルの状態にリセットしようとします。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: ファイルをアップロード、またはダウンロードするEtherCATスレーブの固定アドレス。

sPathName: ライト、またはリードするファイルのパスとファイル名(例: 「C:¥FOE_Test¥EL6751¥ECATFW__EL6751_C6_V0030.efw」)

dwPass: パスワード(デフォルト: 0)。

eMode: 「File access over EtherCAT」アクセスモード [▶ 102] (デフォルト: ライトアクセス)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間(デフォルト: 200 s)。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

cbLoad: 正常にライトまたはリードしたデータバイト数。

nProgress: ライトアクセスの進捗状態(範囲: 0～100%)。この変数は、現在、リードアクセスに使用されません。リードの場合、常に0です。

sInfo: 文字列による追加のコマンド情報(予約済み)。

STでの例:

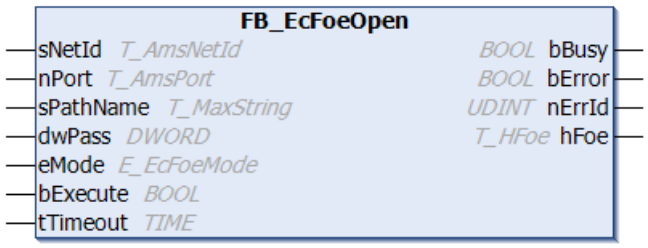
bLoad変数の立ち上がりエッジで、「File access over EtherCAT」メールボックスプロトコル経由でファームウェアのダウンロードをトリガします。

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

7.4 FB_EcFoeOpen



このファンクションブロックを使用して、「File access over EtherCAT」メールボックスプロトコルの通信ポートをオープンします。

VAR_INPUT

```
VAR_INPUT
    sNetId    : T_AmsNetId;
    nPort     : UINT;
```

```
    sPathName : T_MaxString;
    dwPass    : DWORD;
    eMode     : E_EcFoeMode;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nPort: EtherCATデバイスの固定アドレス。

sPathName: パス名 (例: 「c:¥TwinCAT¥FOE¥Data.fwp」)。デフォルトでは、ファイル名のみ (拡張子なし) がFoEプロトコルのファイルパスにファイル名に使用されます(この例では: 「Data」)。ライブラリバージョン3.3.12.0以降から、ファイル名の拡張子を含むファイル名も使用できます(ここの例では: 「Data.fwp」)。

FB_EcFoeOpenファンクションブロックのすべてのインスタンスに対して、ファイル名拡張子の使用をグローバルなBoolean変数
Tc2_EtherCAT.bEcFoeOpenFileNameWithFileExt
で有効、無効にできます。デフォルトでは、変数の値はFALSE (ファイル名拡張子なし)です。値をTRUEにセットした場合、ファイル名拡張子は有効になり使用できます。
FoEファンクションブロックは、元々、ファームウェア更新のためにファイル名拡張子なしで使用されていたことに注意してください。ファームウェアを更新したい場合、グローバル変数がオリジナルのデフォルト値、すなわちFALSEであることを確認する必要があります。

dwPass: パスワード。

eMode: アクセスモード [▶ 102](ライト/リードアクセス)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

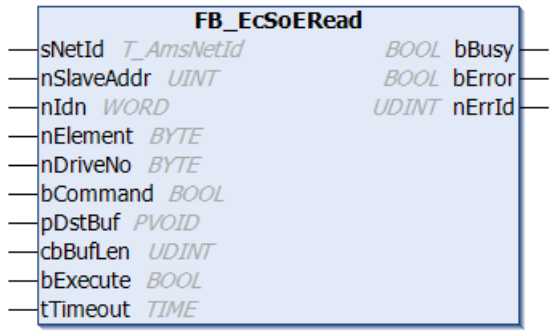
hFoe: 「File access over EtherCAT」ハンドル [▶ 112]。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

8 SoEインターフェイス

8.1 FB_EcSoeRead



ファンクションブロックFB_EcSoeReadを使用して、「Servo drive profile over EtherCAT (SoE)」プロトコルでドライブパラメータをリードできます。このためには、スレーブのメールボックス通信と、SoEプロトコルをサポートしていることが必要です。リードするドライブパラメータは、パラメータnIdn (識別番号)、nElementおよびnDriveNoで指定します。

VAR_INPUT

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

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SoEリードコマンドを送信するEtherCATスレーブの固定アドレス。

nIdn: リードするパラメータの識別番号。

nElement: リードするパラメータのエレメント番号。以下の値が許可されます。

値	説明
0x01	データステータス
0x02	名前(リードのみ)
0x04	アトリビュート
0x08	単位
0x10	最小
0x20	最大
0x40	値
0x80	デフォルト

nDriveNo: ドライブ番号。

bCommand: 内部コマンドの実行を使用する場合、このパラメータをセットしてください。

pDstBuf: 受信バッファのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

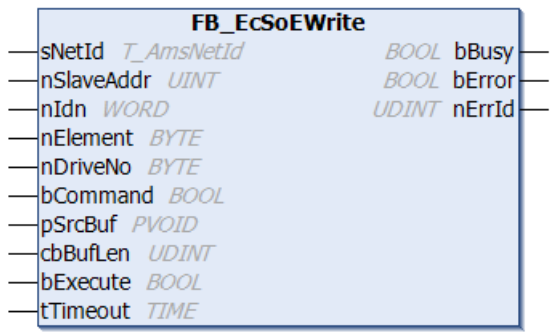
STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

8.2 FB_EcSoeWrite



ファンクションブロックFB_EcSoeWriteを使用して、「Servo drive profile over EtherCAT (SoE)」プロトコルでドライブパラメータをライトできます。このためには、スレーブのメールボックス通信と、SoEプロトコルをサポートしていることが必要です。ライトするドライブパラメータは、パラメータnIdn（識別番号）、nElementおよびnDriveNoで指定します。

VAR_INPUT

```
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
```

sNetId: EtherCATマスタデバイスのAMSネットワークIDの文字列。(T_AmsNetId型)

nSlaveAddr: SoE Writeコマンドを送信するEtherCATスレーブの固定アドレス。

nIdn: ライトするパラメータの識別番号。

nElement: ライトするパラメータのエレメント番号。以下の値が許可されます。

値	説明
0x01	データステータス
0x02	名前(リードのみ)
0x04	アトリビュート
0x08	単位
0x10	最小
0x20	最大
0x40	値
0x80	デフォルト

nDriveNo: ドライブ番号。

bCommand: 内部コマンドの実行を使用する場合、このパラメータをセットする必要があります。

pSrcBuf: 送信バッファのアドレス(ポインタ)。

cbBufLen: 送信するデータの数(バイト単位)

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

nErrId: bError出力がセットされている場合、最後に実行されたコマンドと関連したADSエラーコードを返します。

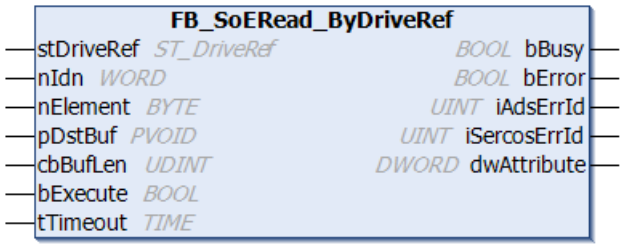
STでの実装例:

```
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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

8.3 FB_SoERead_ByDriveRef



ファンクションブロックFB_SoeRead_ByRefを使用して、「Servo drive profile over EtherCAT (SoE)」プロトコルでドライブパラメータをリードできます。このためには、スレーブのメールボックス通信と、SoEプロトコルをサポートしていることが必要です。リードするドライブパラメータは、パラメータnIdn (識別番号)、nElementおよびstDriveRefで指定します。

Tc2_EtherCATライブラリからのグローバル変数bSeqReadDrvAttrAndValue := TRUEを使用して、アトリビュートと値のアクセスを強制的にシーケンシャルにできます。この変数のデフォルト値はFALSEです。AX5xxxシリーズのデバイスは、アトリビュートと値にパラレルまたはシーケンシャルアクセスを有効にできます。サードパーティ製のデバイスの場合、全体的に数サイクル分アクセスが遅くなるので、アトリビュートと値へのアクセスを分離する必要があるかもしれません。

VAR_INPUT

```
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
```

stDriveRef: ドライブへの参照は、System ManagerのPLCに直接リンクできます。この目的のために、ST_PlcDriveRefのインスタンスを使用し、バイト列のNEtIDを文字列に変換する必要があります。(ST_DriveRef型)

nIdn: リードするパラメータの識別番号。

nElement: リードするパラメータのエレメント番号。以下の値が許可されます。

値	説明
0x01	データステータス
0x02	名前(リードのみ)
0x04	アトリビュート
0x08	単位
0x10	最小
0x20	最大
0x40	値
0x80	デフォルト

pDstBuf: リードバッファへのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

iAdsErrId: bError出力がセットされている場合、最後に実行されたコマンドの ADSエラーコードを返します。

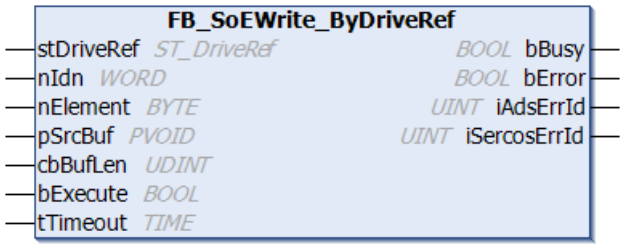
iSercosErrId: bError出力がセットされている場合、最後に実行されたコマンドのSercosエラーを返します。

dwAttribute: Sercosパラメータのアトリビュートを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

8.4 FB_SoEWrite_ByDriveRef



ファンクションブロックFB_SoeWrite_ByRefを使用して、「Servo drive profile over EtherCAT (SoE)」プロトコルでドライブパラメータをライトできます。このためには、スレーブのメールボックス通信と、SoEプロトコルをサポートしていることが必要です。ライトするドライブパラメータは、パラメータnIdn (識別番号)、nElementおよびstDriveRefで指定します。

Tc2_EtherCATライブラリからのグローバル変数bSeqReadDrvAttrAndValue := TRUEを使用して、アトリビュートと値のアクセスを強制的にシーケンシャルにできます。この変数のデフォルト値はFALSEです。AX5xxxシリーズのデバイスは、アトリビュートと値へのパラレルおよびシーケンシャルアクセスを有効にできます。サードパーティ製のデバイスの場合、全体的に数サイクル分アクセスが遅くなるので、アトリビュートと値へのアクセスを分離する必要があるかもしれません。

VAR_INPUT

```
VAR_INPUT
    stDriveRef : ST_DriveRef; (* contains sNetID of EcMaster, nSlaveAddr of EcDrive, nDriveNo of EcDrive, 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
```

stDriveRef: ドライブへの参照は、System ManagerのPLCに直接リンクできます。この目的のために、ST_PlcDriveRefのインスタンスを使用し、バイト列のNEtIDを文字列に変換する必要があります。(ST_DriveRef型)

nIdn: リードするパラメータの識別番号。

nElement: リードするパラメータのエレメント番号。以下の値が許可されます。

値	説明
0x01	データステータス
0x02	名前(リードのみ)
0x04	アトリビュート
0x08	単位
0x10	最小
0x20	最大
0x40	値
0x80	デフォルト

pSrcBuf: 送信バッファのアドレス(ポインタ)。

cbBufLen: リードデータ用に最大使用可能バッファサイズ(バイト)。

bExecute: ファンクションブロックは、この入力の立ち上がりエッジによって有効になります。

tTimeout: ファンクションブロックの実行に許容する最大時間。

VAR_OUTPUT

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

bBusy: ファンクションブロックの実行中に確認レスポンスを受信するまでセットされたままの状態となります。

bError: コマンドの送信中にエラーが発生した場合、bBusy出力がリセットされた後でこの出力がセットされます。

iAdsErrId: bError出力がセットされている場合、最後に実行されたコマンドの ADSエラーコードを返します。

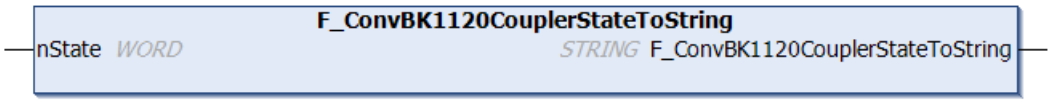
iSercosErrId: bError出力がセットされている場合、最後に実行されたコマンドのSercosエラーを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9 変換関数

9.1 F_ConvBK1120CouplerStateToString



ファンクションF_ConvBK1120CouplerStateToStringは、BK1120/BK1150/BK1250のカプラの状態を文字列として返します。nState = 0の場合は「No error」を返し、そうでない場合、たとえばnState = 1の場合は「K-bus error」を返します。複数のエラーが保留中の場合、各エラーはカンマで区切られます。

VAR_INPUT

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

nState: カプラの状態。System ManagerでBK1120/BK1250の入力からPLCへリンクできます。

```
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)'
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.2 F_ConvMasterDevStateToString

ファンクションF_ConvMasterDevStateToStringは、EtherCATマスタのデバイスステータスを文字列に変換します。

nState = 0の場合、「OK」が返されます。そうでない場合、「Not OK - Link error」が返されます。たとえば、nState = 1の場合です。複数のエラーが保留中の場合、各エラーはハイフォンで区切られます。

VAR_INPUT

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

nstate: EtherCATマスタのデバイスステータス。System Managerで、EtherCATマスタの入力からPLCへDevStateとしてリンクできます。

```
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'
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.3 F_ConvProductCodeToString

ファンクションF_ConvProductToStringは、製品コードを文字列として返します(例: 「EL6731-0000-0017」)。また、Tc2_EtherCATライブラリのバージョン3.3.8.0以降から、このファンクションは「EPP4374-0002-0018」および「ELM3704-0001-0016」のようなELMおよびEPPスレーブをサポートしています。

VAR_INPUT

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

stSlaveIdentity: FB_EcGetSlaveIdentity [▶ 37]でリードしたスレーブID。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.4 F_ConvSlaveStateToString

ファンクションF_ConvSlaveStateToStringは、EtherCATスレーブを文字列として返します。文字列への変換については、F_ConvStateToString [▶ 68]を参照してください。

VAR_INPUT

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

state: EtherCATスレーブステートの構造体(構成: deviceState : BYTE; linkState : BYTE;)

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.5 F_ConvSlaveStateToBits

ファンクションF_ConvSlaveStateToBitsは、 EtherCATスレーブステートを構造体TYPE ST_EcSlaveStateBits [▶ 107]として返します。

VAR_INPUT

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

stEcSlaveState: EtherCATスレーブステートの構造体(構成: deviceState : BYTE; linkState : BYTE;)

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.6 F_ConvSlaveStateToBitsEx

ファンクションF_ConvSlaveStateToBitsExは、EtherCATスレーブステートを構造体ST_EcSlaveStateBitsEx [▶ 108]として返します。

VAR_INPUT

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

stEcSlaveState: EtherCATスレーブステートの構造体(構成: deviceState : BYTE; linkState : BYTE;)

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

9.7 F_ConvStateToString

ファンクションF_ConvStateToStringは、EtherCATスレーブを文字列として返します。nState = 0の場合、「 」が返されます。そうでない場合「INIT」が返されます。たとえば、nState = 1の場合です。複数のメッセージが保留中の場合、各メッセージはスペースで区切られます。

VAR_INPUT

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

nstate: WORD型のEtherCATスレーブステート。

```
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'
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10 ディストリビュートクロック

10.1 DCTIME32

10.1.1 ConvertDcTimeToPos

このファンクションブロックは、 T_DCTIME32 [▶ 110]型の32ビットのディストリビュートクロックのシステムタイム変数を対応するNC軸の位置(すなわち、正確にこの時刻のNC軸の位置)に変換します。

VAR_INPUT

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

nAxisId: NC軸のID。

nSubIdx: この32ビットの入力変数には2つの異なる情報アイテムが含まれ、そのため2つの16ビット値に分割されます。

- 下位ワード(最下位の16ビット)には、軸のエンコーダサブエレメントのサブインデックスが含まれています。サブインデックスはゼロから増えてカウントされます。たとえば、エンコーダが1台の軸の場合、サブインデックスはゼロです。
- 上位ワード(最上位の16ビット)には、位置を計算する方法に影響を及ぼす制御ワード(ビットマスク)が含まれます(例: 内挿タイプまたは外挿タイプ)。ビットマスク値0x0001は、軸の設定された加速度が計算に含まれていることを意味します。

dcTime: 32ビットのディストリビュートクロックのシステムタイム変数。この入力の大きさは、計算により対応するNC軸の位置に変換されます。

注記

32ビット時刻は、あいまいさを避けるために現在のシステムタイムを中心にして± 2,147秒の狭い範囲でのみ使用することができます。ファンクションブロック内では、この前提条件をチェックできません。

VAR_OUTPUT

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

fPosition: dcTimeに対応するNC軸の位置を返します。このNC軸の位置は、スケーリングされてオフセットであり、たとえば度、またはミリメートルの物理単位を持ちます。

iErr: エラー0x4012 (軸IDは許可されません。または軸はシステム内に存在しません)などのエラーが発生した場合、エラー番号を返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.1.2 ConvertPosToDcTime

このファンクションブロックは、NC軸の位置をT_DCTIME32 [▶ 110]型の対応する32ビットディストリビュートクロックのシステムタイム変数に変換します(すなわち、このNC軸の位置に到達した正確な時刻、または到達する正確な時刻)。

VAR_INPUT

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

nAxisId: NC軸のID。

nSubIdx: この32ビットの入力の大きさは、2つの異なる情報アイテムから構成され、2つの16ビット値に分割されます。

- 下位ワード(最下位の16ビット)には、軸のエンコーダサブエレメントのサブインデックスが含まれています。サブインデックスはゼロから増えてカウントされます。たとえば、エンコーダが1台の軸の場合、サブインデックスはゼロです。
- 上位ワード(最上位の16ビット)には、位置を計算する方法に影響を及ぼす制御ワード(ビットマスク)が含まれます(例: 内挿タイプまたは外挿タイプ)。ビットマスク値0x0001は、軸の設定された加速度が計算に含まれていることを意味します。

fPosition: 対応する32ビットディストリビュートクロックシステムタイム変数に変換されるNC軸の位置。その位置に対応するディストリビュートクロックのシステムタイムが± 2,147秒の推定時刻の枠外にある場合、この変換はエラー番号付きで拒否されます。

VAR_OUTPUT

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

dcTime: 入力fPositionに相当する32ビットディストリビュートクロックのシステムタイム変数を返します。

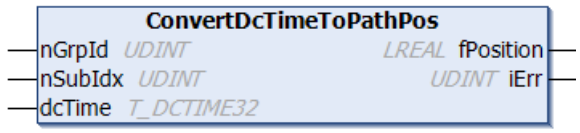
iErr: エラーが発生した場合、エラー番号を提供します。

- たとえば、エラー0x4012: 軸IDは許可されないか、またはシステムに存在しません。
- エラー0x4361:時間枠を超過(将来)、
- エラー0x4362: 時間枠を超過(過去)、
- エラー0x4363: 数学的に位置を検出できません。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.1.3 ConvertDcTimeToPathPos



このファンクションブロックは、 T_DCTIME32 [▶ 110]型の32ビットディストリビュートクロックのシステムタイム変数を現在のアクティブなNciプログラムの曲線上での相対的Nci経路長に変換します(すなわち、ファンクションブロックはタイミングに応じて、正または負の相対的な時間を返します)。

VAR_INPUT

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

nGrpId: 対応するNciチャンネルのグループID。

nSubIdx: この32ビットの入力変数には2つの異なる情報アイテムが含まれ、そのため2つの16ビット値に分割されます。

- 下位ワード(最下位の16ビット)には、軸のエンコーダサブエレメントのサブインデックスが含まれています。サブインデックスはゼロから増えてカウントされます。たとえば、エンコーダが1台の軸の場合、サブインデックスはゼロです。
- 上位ワード(最上位の16ビット)には、位置を計算する方法に影響を及ぼす制御ワード(ビットマスク)が含まれます(例: 内挿タイプまたは外挿タイプ)。
  ビットマスク値0x0001は、軸の設定された加速度が計算に含まれていることを意味します。
  ビットマスク0x0010は、計算は相対的で現在必須であることを意味します。それ以外の場合、コールはエラーで拒否されます。

dcTime: 32ビットのディストリビュートクロックのシステムタイム変数。この入力変数は、曲線上での対応する相対的なNci経路長に変換されます。

注記

32ビット時刻は、あいまいさを避けるために現在のシステムタイムを中心にして± 2,147秒の狭い範囲でのみ使用することができます。ファンクションブロック内では、この前提条件をチェックできません。

VAR_OUTPUT

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

fPosition: dcTimeに相当する曲線上での相対的Nci経路長を返します。

iErr: エラーの場合には、エラー番号を返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.1.4 ConvertPathPosToDcTime

このファンクションブロックは、相対的なNci経路長をT_DCTIME32 [▶ 110]型の対応する32ビットディストリビュートクロックのシステムタイム変数に変換します(すなわち、相対的なNci経路長に相当する、または相当した時刻)。

VAR_INPUT

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

nGrpId: 対応するNciチャンネルのグループID。<b />

nSubIdx: この32ビットの入力変数には2つの異なる情報アイテムが含まれ、そのため2つの16ビット値に分割されます。

- 下位ワード(最下位の16ビット)には、軸のエンコーダサブエレメントのサブインデックスが含まれています。サブインデックスはゼロから増えてカウントされます。たとえば、エンコーダが1台の軸の場合、サブインデックスはゼロです。
- 上位ワード(最上位の16ビット)には、位置を計算する方法に影響を及ぼす制御ワード(ビットマスク)が含まれます(例: 内挿タイプまたは外挿タイプ)。
  ビットマスク値0x0001は、軸の設定された加速度が計算に含まれていることを意味します。
  ビットマスク0x0010は、計算は相対的で現在必須であることを意味します。それ以外の場合、コールはエラーで拒否されます。

fPosition: 対応する32ビットディストリビュートクロックのシステムタイムに変換される相対的なNci経路長。
相対的なNci経路長に対応するディストリビュートクロックのシステムタイムが± 2,147秒の推定時刻の枠外にある場合、この変換はエラー番号付きで拒否されます。

VAR_OUTPUT

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

dcTime: 入力fPositionに相当する32ビットディストリビュートクロックのシステムタイム変数を返します。

iErr: エラーが発生した場合、エラー番号を提供します。

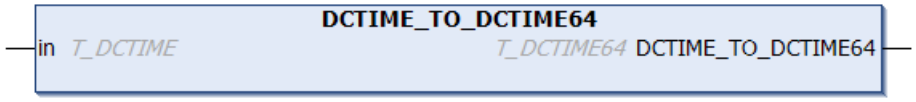
- エラー0x4361: 時間枠を超過(将来)、
- エラー0x4362: 時間枠を超過(過去)、
- エラー0x4363: 数学的に位置を検出できません。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2 DCTIME64

10.2.1 DCTIME_TO_DCTIME64



ファンクションは、T_DCTIME [▶ 111]型の ディストリビュートクロックのシステムタイム変数を T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。

FUNCTION DCTIME_TO_DCTIME64: T_DCTIME64

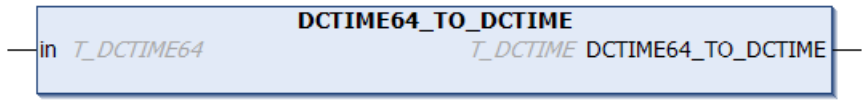
```
VAR_INPUT
    in : T_DCTIME;
END_VAR
```

in: 変換されるディストリビュートクロックのシステムタイム変数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.2 DCTIME64_TO_DCTIME



ファンクションは、T_DCTIME [▶ 111]型の ディストリビュートクロックのシステムタイム変数を T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。

FUNCTION DCTIME64_TO_DCTIME: T_DCTIME

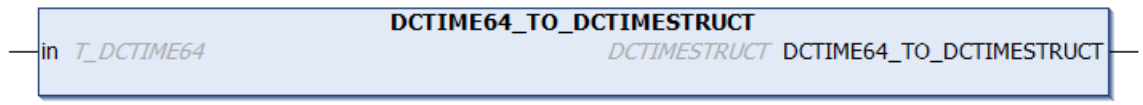
```
VAR_INPUT
    in : T_DCTIME64;
END_VAR
```

in: 変換されるディストリビュートクロックのシステムタイム変数。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.3 DCTIME64_TO_DCTIMESTRUCT



ファンクションは、T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数を DCTIMESTRUCT [▶ 110]型の構造体変数に変換します。

FUNCTION DCTIME64_TO_DCTIMESTRUCT

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

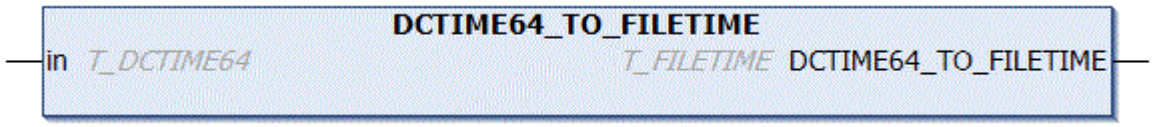
例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.4 DCTIME64_TO_FILETIME



ファンクションは、T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数をT_FILETIME型の64ビットのWindowsのファイル時刻変数に変換します。

FUNCTION DCTIME64_TO_FILETIME: T_FILETIME

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

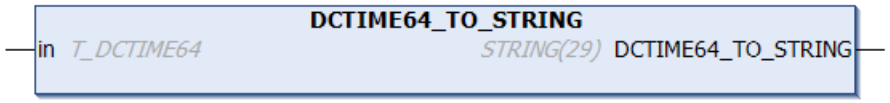
例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.5 DCTIME64_TO_STRING



ファンクションは、T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数を文字列に変換します。

変換で生成される文字列は、以下のような「YYYY-MM-DD-hh:mm:ss.nnnnnnnnn」のフォーマットになります。

- YYYY: 年
- MM: 月
- DD: 日
- hh: 時
- mm: 分
- ss: 秒
- nnnnnnnnn: ナノ秒

FUNCTION DCTIME64_TO_STRING: STRING (29)

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

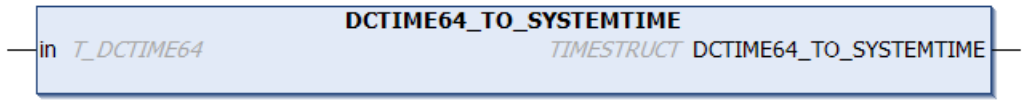
in: 変換されるディストリビュートクロックのシステムタイム変数。

例:
ファンクション F_GetCurDcTickTime64 [▶ 83]の説明を参照してください。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.6 DCTIME64_TO_SYSTEMTIME



ファンクションは、T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数をTIMESTRUCT型の構造体のWindowsのシステムタイム変数に変換します。

DCTIME64_TO_SYSTEMTIME: TIMESTRUCT

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

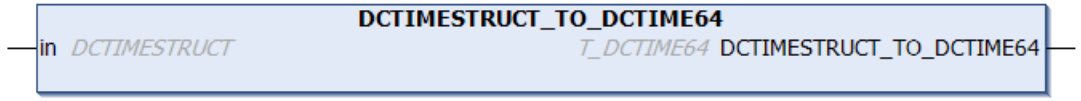
例:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT
END_VAR

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.7 DCTIMESTRUCT_TO_DCTIME64



ファンクションは、DCTIMESTRUCT [▶ 110]型の構造体変数を64ビットディストリビュートクロックのシステムタイム変数T_DCTIME64 [▶ 111]に変換します。構造体コンポーネントwDayofWeekは変換では無視されます。構造体コンポーネントwYearは2000以上で2584未満でなければいけません。構造体コンポーネントの値が無効な場合、ファンクションは値ゼロを返します。

FUNCTION DCTIMESTRUCT_TO_DCTIME64: T_DCTIME64

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

in: 変換される構造体変数

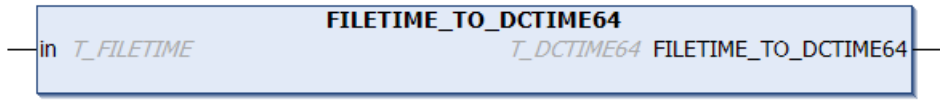
例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.8 FILETIME_TO_DCTIME64



ファンクションは、T_FILETIME型の64ビットのWindowsのファイル時刻変数をT_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。変換エラーの場合、ファンクションは値ゼロを返します。

FUNCTION FILETIME_TO_DCTIME64: T_DCTIME64

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

in: 変換されるWindowsのファイル時刻変数。

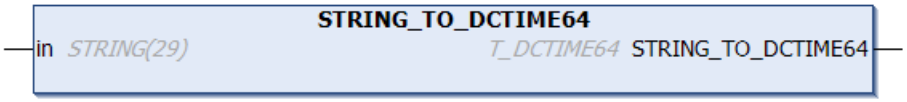
例:

```
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);
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.9 STRING_TO_DCTIME64



ファンクションは、文字列をT_DCTIME64 [▶ 111]型のディストリビュートクロックのシステムタイム変数に変換します。

FUNCTION STRING_TO_DCTIME64: T_DCTIME64

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

in: 変換される文字列。

文字列は、以下のような「‘YYYY-MM-DD-hh:mm:ss:nnnnnnnnn」‘フォーマットになります。

- YYYY: 年
- MM: 月
- DD: 日
- hh: 時
- mm: 分
- ss: 秒
- nnnnnnnnn: ナノ秒

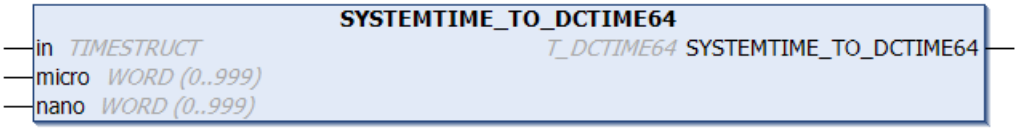
例:

ファンクションF_GetCurDcTickTime64 [▶ 83]の説明を参照してください。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.10 SYSTEMTIME_TO_DCTIME64



ファンクションは、TIMESTRUCT型の構造体のWindowsのシステムタイム変数をT_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。変換エラーの場合、ファンクションは値ゼロを返します。

FUNCTION SYSTEMTIME_TO_DCTIME64: T_DCTIME64

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

in: 変換されるWindowsのシステムタイム変数。

micro: マイクロ秒

nano: ナノ秒

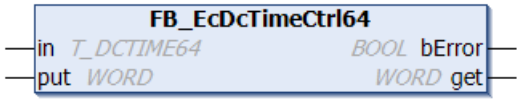
例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.2.11 FB_EcDcTimeCtrl64



このファンクションブロックを使用して、T_DCTIME64 [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数の年、月、日のような個々のコンポーネントをリードできます。ファンクションブロックには、いくつかのA_GETXYZアクションがあります。必要なアクションがコールされると、XYZコンポーネントの値は「get」出力変数で利用できます。「put」入力変数は現在使用されていません。

ファンクションブロックには、次のタスクがあります。

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

VAR_IN_OUT

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

in: TwinCATのディストリビュートクロックのシステムタイム変数

VAR_INPUT

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

put: 入力パラメータ(現在使用されていません)

VAR_OUTPUT

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

bError: アクションのコール中にエラーが発生した場合は、この出力がセットされます。

get: 出力パラメータ(年、月、日等)

STでの実装例:

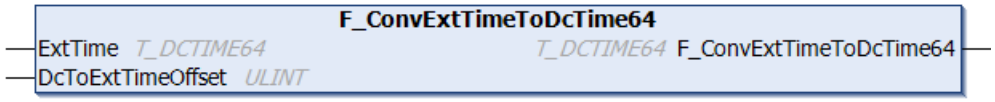
```
PROGRAM P_TEST
VAR
    dcStruct   : DCTIMESTRUCT;
    dcTime     : T_DCTIME64;
    fbCtrl     : FB_EcDcTimeCtrl;

    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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3 DCTIME64およびULINT

10.3.1 F_ConvExtTimeToDcTime64



ファンクション F_ConvExtTimeToDcTime64は、外部時刻をTwinCATディストリビュートクロックのシステムタイムに変換します。

FUNCTION F_ConvExtTimeToDcTime64: T_DCTIME64

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

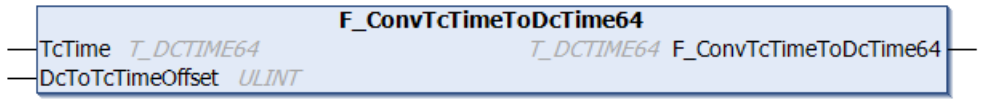
ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.2 F_ConvTcTimeToDcTime64



ファンクションF_ConvTcTimeToDcTime64は、TwinCATシステムタイムをTwinCATディストリビュートクロックのシステムタイムに変換します。

FUNCTION F_ConvTcTimeToDcTime64: T_DCTIME64

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

TcTime: TwinCATディストリビュートクロックのシステムタイムの形式でのTwinCATシステムタイム

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.3 F_ConvTcTimeToExtTime64

ファンクションF_ConvTcTimeToExtTime64は、TwinCATディストリビュートクロックのシステムタイムを外部時刻に変換します。

FUNCTION F_ConvTcTimeToExtTime64: T_DCTIME64

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

TcTime: ディストリビュートクロック形式でのTwinCATシステムタイム

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.4 F_GetActualDcTime64

このファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式で現在の時刻を返します(T_DCTIME64 [▶ 111])。

FUNCTION F_GetActaulDCTime: T_DCTIME64

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

STでの例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.5 F_GetCurDcTaskTime64

このファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME64 [▶ 111])でタスクの開始時刻(タスクが開始する時刻)を返します。ファンクションは常に、コールするタスクの開始時刻を返します。

FUNCTION F_GetCurDcTaskTime64: T_DCTIME64

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.6 F_GetCurDcTickTime64

ファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME64 [▶ 111])で現在(最後)のティックの時刻を返します。

FUNCTION F_GetCurDcTickTime64: T_DCTIME64

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

例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.7 F_GetCurExtTime64

ファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME64 [▶ 111]型)で外部時刻を返します。

FUNCTION F_GetCurExtTime64: T_DCTIME64

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

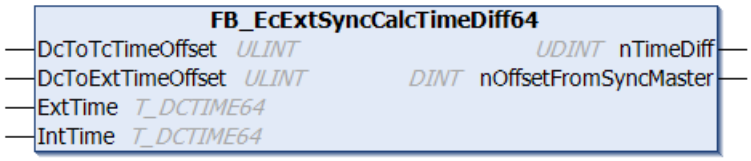
DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.8 FB_EcExtSyncCalcTimeDiff64



ファンクションブロックFB_EcExtSyncCalcTimeDiff64は、タイムオフセットを考慮して外部時刻と内部時刻の間の差を計算します。

VAR_IN_OUT

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

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

IntTime: TwinCATディストリビュートクロックのシステムタイムの形式での内部時刻

VAR_OUTPUT

```
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
```

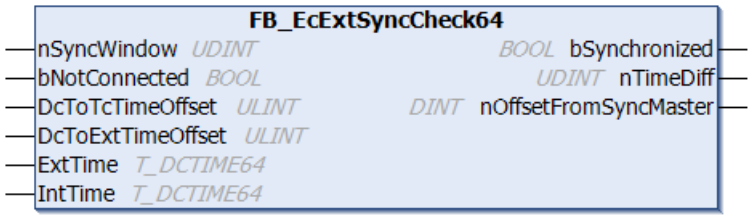
nTimeDiff: 差が32ビット未満の場合は、時間差を返します。差が32ビット以上の場合、16#FFFFFFFFを返します。

nOffsetFromSyncMaster: 差が32ビット以上で、内部時刻とDC時刻の間のオフセットが32ビット未満の場合、16#80000000を返します。
差が32ビット以上で、内部時刻とDC時刻との間のオフセットが32ビット以上の場合、16#7FFFFFFFを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.3.9 FB_EcExtSyncCheck64



ファンクションブロックFB_EcExtSyncCheck64は、内部クロックと外部クロックが同期しているかチェックします。ファンクションブロック FB_EcExtSyncCalcTimeDiff64 [▶ 84]を参照してください。

VAR_INPUT

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

nSyncWindow: 内部クロックと外部クロックが同期しているとみなされる時間範囲。

bNotConnected: TRUE = 外部クロックへの接続が中断。

VAR_IN_OUT

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

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

IntTime: TwinCATディストリビュートクロックのシステムタイムの形式での内部時刻

VAR_OUTPUT

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

bSynchronized: TRUE = 外部クロックと内部クロックが同期

nTimeDiff: 2つのクロック間の現在の時間差

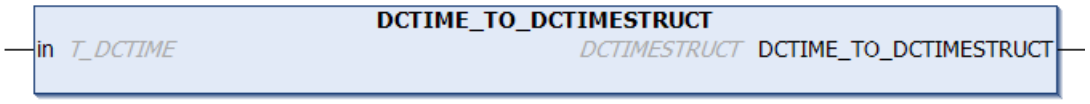
nOffsetFromSyncMaster: Sync Masterへのオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4 [旧DCTIME]

10.4.1 DCTIME_TO_DCTIMESTRUCT



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションDCTIME64_TO_DCTIMESTRUCT [▶ 74]を使用してください。

ファンクションは、T_DCTIME [▶ 111]型の 64ビットディストリビュートクロックのシステムタイム変数をDCTIMESTRUCT [▶ 110]型の構造体変数に変換します。

FUNCTION DCTIME_TO_DCTIMESTRUCT: DCTIMESTRUCT

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.2 DCTIME_TO_FILETIME

旧ファンクション

このファンクションは使用されていません。かわりに、ファンクション DCTIME64_TO_FILETIME [▶ 75]を使用してください。

ファンクションは、T_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数をT_FILETIME型の64ビットのWindowsのファイル時刻変数に変換します。

FUNCTION DCTIME_TO_FILETIME: T_FILETIME

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

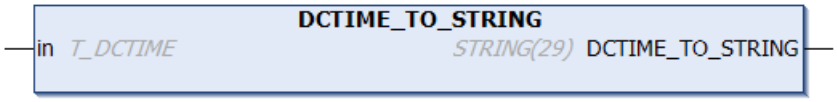
例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.3 DCTIME_TO_STRING



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションDCTIME64_TO_STRING [▶ 76]を使用してください。

ファンクションは、文字列をT_DCTIME [▶ 111]型のディストリビュートクロックのシステムタイム変数に変換します。

変換で生成される文字列は、以下のような「YYYY-MM-DD-hh:mm:ss.nnnnnnnnn」フォーマットになります。

- YYYY: 年
- MM: 月
- DD: 日
- hh: 時
- mm: 分

- ss: 秒
- nnnnnnnnn: ナノ秒

FUNCTION DCTIME_TO_STRING: STRING(29)

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

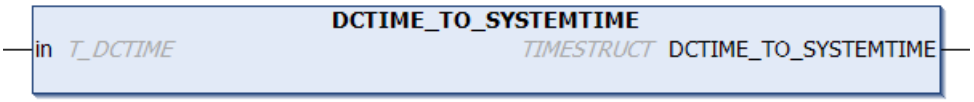
例:

ファンクションF_GetCurDcTickTime [▶ 96]の説明を参照してください。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.4 DCTIME_TO_SYSTEMTIME



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションDCTIME64_TO_SYSTEMTIME [▶ 76]を使用してください。

ファンクションは、T_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数をTIMESTRUCT型の構造体のWindowsのシステムタイム変数に変換します。

FUNCTION DCTIME_TO_SYSTEMTIME: TIMESTRUCT

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

in: 変換されるディストリビュートクロックのシステムタイム変数。

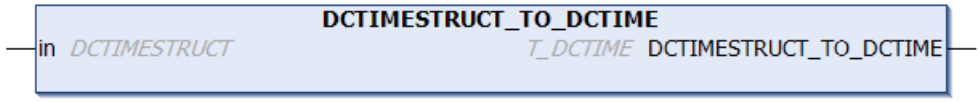
例:

```
PROGRAM P_TEST
VAR
    syst : TIMESTRUCT;
END_VAR

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.5 DCTIMESTRUCT_TO_DCTIME



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションDCTIMESTRUCT_TO_DCTIME64 [▶ 77]使用してください。

ファンクションは、DCTIMESTRUCT [▶ 110]型の構造体変数をT_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。
構造体コンポーネントwDayofWeekは、変換では無視されます。構造体コンポーネントwYearは2000以上で2584未満でなければいけません。構造体コンポーネントの値が無効な場合、ファンクションは値ゼロを返します。

FUNCTION DCTIMESTRUCT_TO_DCTIME: T_DCTIME

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

in: 変換される構造体変数。

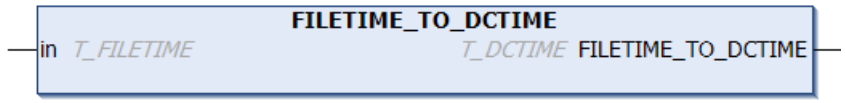
例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.6 FILETIME_TO_DCTIME



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションFILETIME_TO_DCTIME64 [▶ 77]を使用してください。

ファンクションは、T_FILETIME型の64ビットのWindowsのファイル時刻変数をT_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。変換エラーの場合、ファンクションは値ゼロを返します。

FUNCTION FILETIME_TO_DCTIME: T_DCTIME

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

in: 変換されるWindowsのファイル時刻変数。

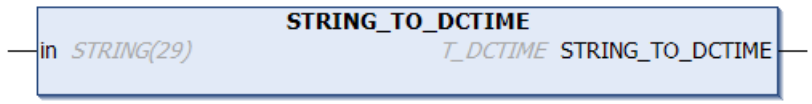
例:

```
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);
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.4.7 STRING_TO_DCTIME



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションSTRING_TO_DCTIME64 [▶ 78]を使用してください。

ファンクションは、文字列をT_DCTIME [▶ 111]型のディストリビュートクロックのシステムタイム変数に変換します。

FUNCTION STRING_TO_DCTIME: T_DCTIME

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

in: 変換される文字列。

文字列は、以下のような「YYYY-MM-DD-hh:mm:ss.nnnnnnnnn」フォーマットでなければいけません。

- YYYY: 年
- MM: 月
- DD: 日
- hh: 時
- mm: 分
- ss: 秒
- nnnnnnnnn: ナノ秒

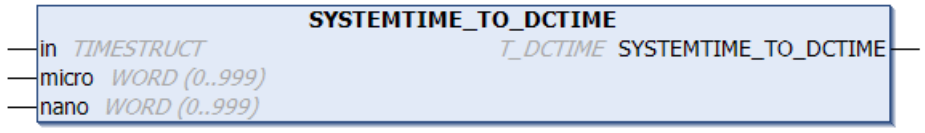
例:

ファンクションF_GetCurDcTickTime [▶ 96]の説明を参照してください。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

10.4.8 SYSTEMTIME_TO_DCTIME



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションSYSTEMTIME_TO_DCTIME64 [▶ 78]を使用してください。

ファンクションは、TIMESTRUCT型の構造体のWindowsのシステムタイム変数をT_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数に変換します。変換エラーの場合、ファンクションは値ゼロを返します。

FUNCTION SYSTEMTIME_TO_DCTIME: T_DCTIME

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

in: 変換されるWindowsのシステムタイム変数。

micro: マイクロ秒。

nano: ナノ秒。

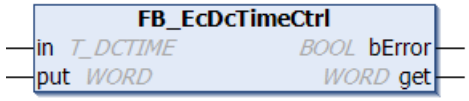
例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

10.4.9 FB_EcDcTimeCtrl



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションブロック FB_EcDcTimeCtrl64 [▶ 79]を使用してください。

このファンクションブロックを使用して、T_DCTIME [▶ 111]型の64ビットディストリビュートクロックのシステムタイム変数の年、月、日のような個々のコンポーネントをリードできます。ファンクションブロックには、いくつかのA_GetXYZアクションがあります。必要なアクションがコールされると、XYZコンポーネントの値は「get」出力変数で利用できます。「put」入力変数は現在使用されていません。

ファンクションブロックには、現在、以下のアクションがあります。

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

VAR_IN_OUT

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

in: TwinCATのディストリビュートクロックのシステムタイム変数

VAR_INPUT

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

put: 入力パラメータ(現在使用されていません)

VAR_OUTPUT

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

bError: アクションのコール中にエラーが発生した場合は、この出力がセットされます。

get: 出力パラメータ(年、月、日等)

STでの実装例:

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

    wYear      : WORD;
    wMonth     : WORD;
    wDay       : WORD;
    wDayOfWeek : WORD;
    wHour      : WORD;
```

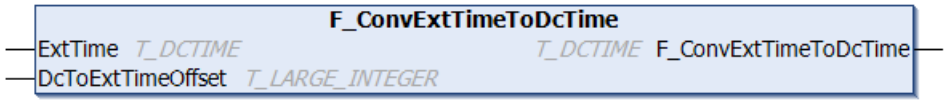
```
    wMinute    : WORD;
    wSecond    : WORD;
    wMilli     : WORD;
    wMicro     : WORD;
    wNano      : WORD;
END_VAR

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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5 [outdated DCTIME and T_LARGE_INTEGER]

10.5.1 F_ConvExtTimeToDcTime



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_ConvExtTimeToDcTime64 [▶ 81]を使用してください。

ファンクションF_ConvExtTimeToDcTimeは、外部時刻をTwinCATディストリビュートクロックのシステムタイムに変換します。

FUNCTION F_ConvExtTimeToDcTime: T_DCTIME

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

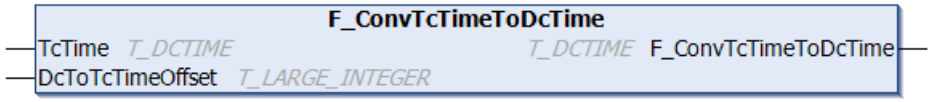
ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.2 F_ConvTcTimeToDcTime



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_ConvTcTimeToDcTime64 [▶ 81]を使用してください。

ファンクションF_ConvTcTimeToDcTime64は、TwinCATシステムタイムをTwinCATディストリビュートクロックのシステムタイムに変換します。

FUNCTION F_ConvTcTimeToDcTime: T_DCTIME

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

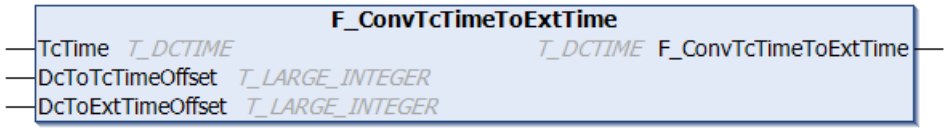
TcTime: TwinCATディストリビュートクロックのシステムタイムの形式でのTwinCATシステムタイム

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.3 F_ConvTcTimeToExtTime



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_ConvTcTimeToExtTime64 [▶ 82]を使用してください。

ファンクションF_ConcTcTimeToExtTimeは、TwinCATディストリビュートクロックのシステムタイムを外部時刻に変換します。

FUNCTION F_ConvTcTimeToExtTime: T_DCTIME

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

TcTime: ディストリビュートクロック形式でのTwinCATシステムタイム

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.4 F_GetActualDcTime

旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_GetActualDcTime64 [▶ 82]を使用してください。

このファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME [▶ 111])で現在の時刻を返します。

FUNCTION F_GetActualDcTime: T_DCTIME

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

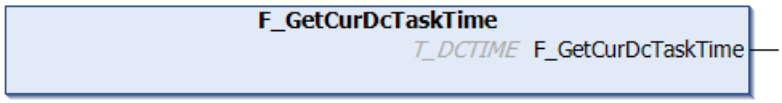
例:

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

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.5 F_GetCurDcTaskTime



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_GetCurDcTaskTime64 [▶ 83]を使用してください。

このファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME [▶ 111])でタスクの開始時刻(タスクが開始する時刻)を返します。ファンクションは常に、コールするタスクの開始時刻を返します。

FUNCTION F_GetCurDcTaskTime: T_DCTIME

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.6 F_GetCurDcTickTime

旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_GetCurDcTickTime64 [▶ 83]を使用してください。

ファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME [▶ 111])で現在(最後)のティックの時刻を返します。

FUNCTION F_GetCurDcTickTime: T_DCTIME

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

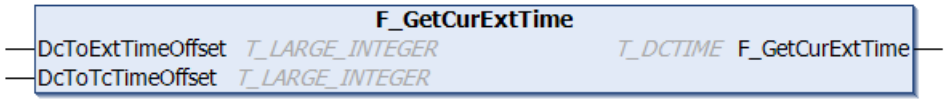
例:

```
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 );
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.7 F_GetCurExtTime



旧ファンクション

このファンクションは使用されていません。かわりに、ファンクションF_GetCurExtTime64 [▶ 84]を使用してください。

ファンクションは、TwinCATディストリビュートクロックのシステムタイムの形式(T_DCTIME [▶ 111])で外部時刻を返します。

FUNCTION F_GetCurExtTime: T_DCTIME

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

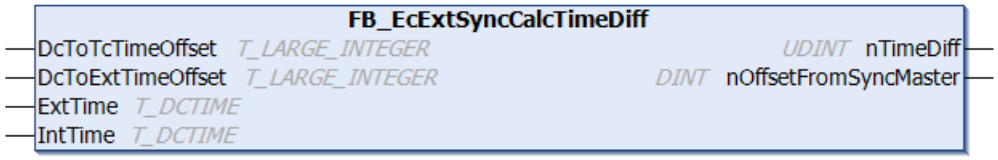
DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.8 FB_EcExtSyncCalcTimeDiff



旧ファンクションブロック

このファンクションブロックは使用されていません。かわりに、ファンクションブロックFB_EcExtSyncCalcTimeDiff64 [▶ 84]を使用してください。

ファンクションブロックFB_EcExtSyncCalcTimeDiffは、タイムオフセットを考慮して外部時刻と内部時刻の間の差を計算します。

VAR_IN_OUT

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

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

IntTime: TwinCATディストリビュートクロックのシステムタイムの形式での内部時刻

VAR_OUTPUT

```
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
```

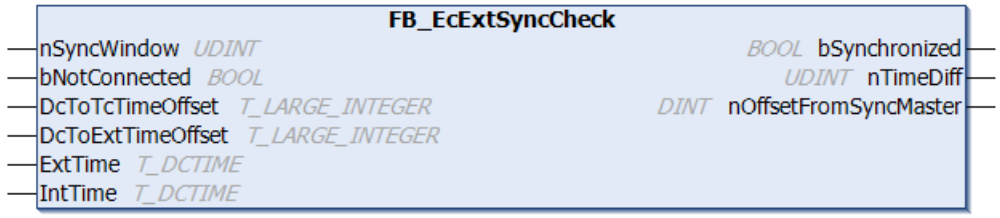
nTimeDiff: 差が32ビット未満の場合は、時間差を返します。差が32ビット以上の場合、16#FFFFFFFFを返します。

nOffsetFromSyncMaster: 差が32ビット以上で、内部時刻とDC時刻の間のオフセットが32ビット未満の場合、16#80000000を返します。
差が32ビット以上で、内部時刻とDC時刻との間のオフセットが32ビット以上の場合、16#7FFFFFFFを返します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

10.5.9 FB_EcExtSyncCheck



旧ファンクションブロック

このファンクションブロックは使用されていません。かわりに、ファンクションブロック FB_EcExtSyncCheck64 [▶ 85]を使用してください。

ファンクションブロックFB_EcExtSyncCheckは、内部クロックと外部クロックが同期しているかチェックします。ファンクションブロックFB_EcExtSyncCalcTimeDiff [▶ 97]を参照してください。

VAR_INPUT

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

nSyncWindow: 内部クロックと外部クロックが同期しているとみなされる時間範囲。

bNotConnected: TRUE = 外部クロックへの接続が中断。

VAR_IN_OUT

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

DcToTcTimeOffset: TwinCATディストリビュートクロックのシステムタイムとTwinCATシステムタイムの間のタイムオフセット

DcToExtTimeOffset: TwinCATディストリビュートクロックのシステムタイムと外部時刻の間のタイムオフセット

ExtTime: TwinCATディストリビュートクロックのシステムタイムの形式での外部時刻

IntTime: TwinCATディストリビュートクロックのシステムタイムの形式での内部時刻

VAR_OUTPUT

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

bSynchronized: TRUE = 外部クロックと内部クロックが同期

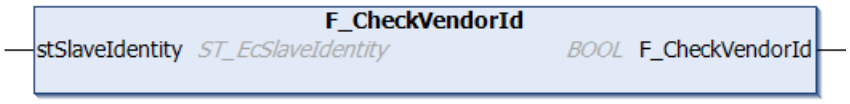
nTimeDiff: 2つのクロック間の現在の時間差

nOffsetFromSyncMaster: Sync Masterへのオフセット

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

11 F_CheckVendorId



ファンクションF_CheckVendorIdはVendorIDがベッコフの場合にTRUEを返し、そうでない場合はFALSEを返します。

VAR_INPUT

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

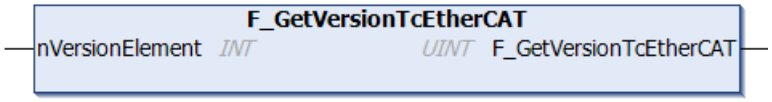
stSlaveIdentity: FB_EcGetSlaveIdentity [▶ 37]でリードしたスレーブID。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

12 [廃止]

12.1 F_GetVersionTcEtherCAT



旧ファンクション

このファンクションは使用されていません。かわりに、グローバル構造体インスタンス stLibVersion_Tc2_EtherCATを使用してください。

このファンクションを使用して、PLCライブラリのバージョン情報をリードできます。

FUNCTION F_GetVersionTcEtherCAT : UINT

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

nVersionElement: リードするバージョン要素です。使用可能なパラメータ:

- 1 : メジャー番号
- 2 : マイナー番号
- 3 : リビジョン番号

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13 データ型

13.1 E_EcAdressingType

EtherCATでのアドレス指定は、固定の設定されたアドレス(eAdressingType_Fixed)に基づく位置依存(eAdressingType_AutoInc)ものか、またはすべてのスレーブ(eAdressingType_Broadcast)に適用されるものかのいずれかです。

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.2 E_EcFoeMode

「File access over EtherCAT」メールボックスプロトコル用のアクセスモード。

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.3 E_EcMbxProtType

サポートされているEtherCATメールボックスプロトコルのタイプ。

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.4 ST_EcCrcError

EtherCATスレーブデバイスの個々のポート(A、BおよびC)のCRCエラーカウンタを含む構造体。

```
TYPE ST_EcCrcError :
STRUCT
    portA : UDINT;
    portB : UDINT;
    portC : UDINT;
END_STRUCT
END_TYPE
```

portA: ポートAのCRCエラーカウンタ

portB: ポートBのCRCエラーカウンタ

portC: ポートCのCRCエラーカウンタ

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.5 ST_EcCrcErrorEx

EtherCATスレーブデバイスの個々のポート(A、B、CおよびD)のCRCエラーカウンタを含む構造体。

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

portA: ポートAのCRCエラーカウンタ

portB: ポートBのCRCエラーカウンタ

portC: ポートCのCRCエラーカウンタ

portD: ポートDのCRCエラーカウンタ

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.6 ST_EcLastProtErrInfo

構造体ST_EcLastProtErrInfoには、最新のEtherCATメールボックスプロトコルエラーに関する追加エラー情報が含まれています。

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

ownAddr:自己のAMSアドレス(エラー情報を確認する通信デバイスのアドレス)。

orgAddr: エラーオリジネータのAMSアドレス (プロトコルエラーをトリガしたか、または引き起こした通信デバイスのアドレス)。

errCode: メールボックスプロトコルのエラー番号 [▶ 115] (SoE、CoE、FoEエラーコード)。

binDesc: バイナリデータとしての追加のエラー情報。追加のエラー情報はデバイス固有で、たとえば文字列またはバイナリデータを含めることができます。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.7 ST_EcMasterStatistic

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

nSysTime: システムタイム(単位マイクロ秒)

nCycFrameCnt: 周期EtherCATフレームの数

nCycFrameMissedCnt: ロスト周期EtherCATフレームの数

nQueuedFrameCnt: 非周期EtherCATフレームの数

nQueuedFrameMissedCnt: ロスト非周期EtherCATフレームの数

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.8 ST_EcSlaveConfigData

構造体ST_EcSlaveConfigDataには、 EtherCATスレーブデバイス用のEtherCATコンフィグレーションデータが含まれています。

```
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
```

nEntries: 内部使用

nAddr: EtherCATスレーブのアドレス

sType: スレーブのEtherCATタイプ

sName: EtherCATスレーブの名前

nDevType: スレーブのEtherCATデバイスタイプ

stSlaveIdentity: EtherCATスレーブのID (ST_EcSlaveIdentity [▶ 105]を参照)

nMailboxOutSize: EtherCATスレーブのメールボックスのOutSize。

nMailboxInSize: EtherCATスレーブのメールボックスのInSize。

nLinkStatus: EtherCATスレーブのリンクステータス。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.9 ST_EcSlaveIdentity

構造体ST_EcSlaveIdentityには、EtherCATスレーブデバイスのEtherCAT IDデータが含まれています。

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

vendorId: スレーブデバイスのベンダーID。

productCode: スレーブデバイスの製品コード。

revisionNo: スレーブデバイスのリビジョン番号を示します。

serialNo: スレーブデバイスのシリアル番号を示します。

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.10 ST_EcSlaveScannedData

構造体ST_EcSlaveScannedDataには、スキャンされたEtherCATスレーブデバイスのEtherCATコンフィグレーションデータが含まれています。

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

nEntries: 内部使用

nAddr: EtherCATスレーブのアドレス

stSlaveIdentity: EtherCATスレーブのID (ST_EcSlaveIdentity [▶ 105]を参照)

ndlStatusReg: ESCレジスタ0110/0111$_{hex}$ または272/273$_{dec}$からのEtherCATスレーブのリンクステータススレーブが到達できない、またはオフラインの場合は、ステータス0が表示されます。ポート番号 <=> ソケット/Ebus接点割り当ては、それぞれのデバイスのマニュアルに記載されています。それ以外に記載されているのでなければ、ポート0はEL/ES端子または EPボックスRJ45 ソケットの左側Ebus接点で、ポート1は右側の外向きのEbus接点/RJ45 ソケットです。

ビットの意味は以下のようになります。

ビット	意味
1	内部使用
2	内部使用
3	内部使用
4	ポート0の物理リンク 0: リンクなし、1: リンクを検出
5	ポート1の物理リンク 0: リンクなし、1: リンクを検出
6	ポート2の物理リンク 0: リンクなし、1: リンクを検出
7	ポート3の物理リンク 0: リンクなし、1: リンクを検出
8	ポート0のループ 0: オープン、1: クローズ
9	ポート0の通信 0: 安定した通信なし、1: 通信確立
10	ポート1のループ 0: オープン、1: クローズ
11	ポート1の通信 0: 安定した通信なし、1: 通信確立
12	ポート2のループ 0: オープン、1: クローズ
13	ポート2の通信 0: 安定した通信なし、1: 通信確立
14	ポート3のループ 0: オープン、1: クローズ
15	ポート3の通信 0: 安定した通信なし、1: 通信確立

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.11 ST_EcSlaveState

構造体ST_EcSlaveStateには、EtherCATスレーブデバイスのEtherCATステートとリンクステータスが含まれています。

```
TYPE ST_EcSlaveState:
STRUCT
    deviceState :BYTE;
```

```
    linkState   :BYTE;
END_STRUCT
END_TYPE
```

deviceState: スレーブのEtherCATステート。ステータスは以下の値のどれかを採用することができます。

定数	値	説明
EC_DEVICE_STATE_INIT	0x01	INIT状態
EC_DEVICE_STATE_PREOP	0x02	Pre-operational状態
EC_DEVICE_STATE_BOOTSTRAP	0x03	Bootstrap状態
EC_DEVICE_STATE_SAFEOP	0x04	Safe-operational状態
EC_DEVICE_STATE_OP	0x08	Operational状態

さらに、以下のビットを設定できます。

定数	値	説明
EC_DEVICE_STATE_ERROR	0x10	EtherCATスレーブのステートマシンエラー
EC_DEVICE_STATE_INVALID_VPRS	0x20	無効なベンダーID、製品コード、リビジョン番号、シリアル番号
EC_DEVICE_STATE_INITCMD_ERROR	0x40	初期化コマンドの送信中のエラー
EC_DEVICE_STATE_DISABLED	0x80	スレーブは無効

linkState: EtherCATスレーブのリンクステータスリンクステータスは、以下のビットのORingから構成できます。

定数	値	説明
EC_LINK_STATE_OK	0x00	
EC_LINK_STATE_NOT_PRESENT	0x01	EtherCATスレーブとのEtherCAT通信なし
EC_LINK_STATE_LINK_WITHOUT_COMM	0x02	ポートXでのエラー(EC_LINK_STATE_PORT_A/B/C/Dにより指定)ポートにはリンクがありますが、このポートを通じて通信はできません。
EC_LINK_STATE_MISSING_LINK	0x04	ポートXの不明リンク(EC_LINK_STATE_PORT_A/B/C/Dにより指定)。
EC_LINK_STATE_ADDITIONAL_LINK	0x08	ポートXの追加リンク(EC_LINK_STATE_PORT_A/B/C/Dにより指定)。
EC_LINK_STATE_PORT_A	0x10	ポート0
EC_LINK_STATE_PORT_B	0x20	ポート1
EC_LINK_STATE_PORT_C	0x40	ポート2
EC_LINK_STATE_PORT_D	0x80	ポート3

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.12 ST_EcSlaveStateBits

構造体ST_EcSlaveStateBitsには、EtherCATスレーブデバイスのEtherCATステートとリンクステータスが含まれています。

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.13 ST_EcSlaveStateBitsEx

構造体ST_EcSlaveStateBitsExには、EtherCATスレーブデバイスのEtherCATステートとリンクステータスが含まれています。

```
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
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.14 ST_PortAddr

構造体ST_PortAddrには、EtherCATスレーブデバイスのEtherCATトポロジ情報が含まれています。EtherCATスレーブデバイスには、通常、2～4ポートがあります。

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

portA: 現在のEtherCATスレーブのポートAでの以前のEtherCATスレーブのアドレス

portB: 現在のEtherCATスレーブのポートBでのオプションの後続のEtherCATスレーブのアドレス

portC: 現在のEtherCATスレーブのポートCでのオプションの後続のEtherCATスレーブのアドレス

portD: 現在のEtherCATスレーブのポートDでのオプションの後続のEtherCATスレーブのアドレス

13.15 ST_TopologyDataEx

構造体ST_TopologyDataExには、EtherCATトポロジとホットコネクトグループの情報が含まれています。

```
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
```

nOwnPhysicalAddr: EtherCATスレーブデバイスの固定EtherCATアドレス

nOwnAutoIncAddr: EtherCATスレーブデバイスのオートインクリメントEtherCATアドレス

stPhysicalAddr: ポートA～DにつながっているEtherCATスレーブデバイスの固定アドレス情報

stAutoIncAddr: ポートA～DにつながっているEtherCATスレーブデバイスのオートインクリメントアドレス情報

aReserved1: 予約済み

nStatusBits:
nStatusBits.0 = TRUE: ディストリビュートクロックをサポート
nStatusBits.1 = TRUE: ディストリビュートクロックをサポート(64ビット)
nStatusBits.2 = TRUE: スレーブが存在
nStatusBits.3 = TRUE: スレーブはホットコネクトグループ
の開始ノード nStatusBits.4 = TRUE: スレーブはホットコネクトグループ内
nStatusBits.5 = TRUE: ホットコネクトがポートBで無効
nStatusBits.6 = TRUE: ホットコネクトがポートCで無効
nStatusBits.7 = TRUE: ホットコネクトがポートDで無効

nHCSlaveCountCfg: ホットコネクトグループデバイスの設定数

nHCSlaveCountAct: ホットコネクトグループデバイスの検出数

aReserved2: 予約済み

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.16 DCTIMESTRUCT

構造体TwinCATディストリビュートクロックのシステムタイムの形式。最小単位はナノ秒です。このデータ型は、**01.01.2000 (GMT)からのナノ秒数**を表しています。

```
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
```

wYear : 年: 2000 ~ 2584

wMonth : 月: 1 ~ 12 (1月 = 1、2月 = 2など)

wDayOfWeek : 曜日: 0 ~ 6 (日曜日 = 0、月曜日 = 1など)

wDay: 月の日: 1～31;

wHour: 時: 0 ~ 23;

wMinute: 分: 0 ~ 59;

wSecond: 秒: 0 ~ 59;

wMilliseconds: ミリ秒: 0～999;

wMicroseconds: マイクロ秒: 0～999;

wNanoseconds: ナノ秒: 0～999;

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.17 T_DCTIME32

32ビットTwinCATディストリビュートクロックのシステムタイムの形式。最小単位はナノ秒です。

この32ビットDCシステムタイムは、最下位の32ビットのみを使用した、フルアブソリュート64ビットDCシステムタイム(T_DCTIME [▶ 111])から形成されています。これは、アブソリュートユニーク時刻のプロパティが失われ、あいまいさを避けるために、現在のシステムタイムを中心として± 2,147秒の狭い時間枠で32ビット時刻のみを使用することを仮定しています。この仮定が該当する多くのアプリケーションがありま

す。
この仮定に違反する場合、解釈でエラーが発生する可能性があり、さらにこの時刻の処理でエラーが発生する可能性があります。

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.18 T_DCTIME64

64ビットTwinCATディストリビュートクロックのシステムタイムの形式。最小単位はナノ秒です。

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

便利なディストリビュートクロックのシステムタイム定数	説明
EC_DCTIME_DELTA_OFFSET64	01.01.1601と01.01.2000の間の100ナノ秒単位の数値これは、Windowsのファイル時刻とディストリビュートクロックのシステムタイムの間の差です。
EC_DCTIME_DATEDELTA_OFFSET	ゼロ年と2000年1月1日の間に経過した日数
EC_DCTIME_TICKSPERMSEC64	ミリ秒当たりのディストリビュートクロックのシステムタイムのナノ秒数
EC_DCTIME_TICKSPERSEC64	秒当たりのディストリビュートクロックのシステムタイムのナノ秒数
EC_DCTIME_TICKSPERDAY64	1日当たりのディストリビュートクロックのシステムタイムのナノ秒数

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

13.19 T_DCTIME

旧データ型

このデータ型は使用されていません。かわりに、データ型T_DCTIME64 [▶ 111]を使用してください。

このデータ型T_DCTIMEは、ディストリビュートクロックのシステムタイム(DC時刻と省略)をリニア64ビット整数値として表します。時刻は、1.1.2000 UTCからナノ秒で表されます。
データ型は2つの32ビットDWORD変数で表され、そのため容易にPLCで処理できます。演算(何回かの加算および減算)をTc2_Utilitiesライブラリのu64関数を使用して実行できます。

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

便利なディストリビュートクロックのシステムタイム定数	説明
EC_DCTIME_DELTA_OFFSET	01.01.1601と01.01.2000の間の100ナノ秒単位の数値これは、Windowsのファイル時刻とディストリビュートクロックのシステムタイムの間の差です。

便利なディストリビュートクロックのシステムタイム定数	説明
EC_DCTIME_DATEDELTA_OFFSET	ゼロ年と2000年1月1日の間に経過した日数
EC_DCTIME_TICKSPERMSEC	ミリ秒当たりのディストリビュートクロックのシステムタイムのナノ秒数
EC_DCTIME_TICKSPERSEC	秒当たりのディストリビュートクロックのシステムタイムのナノ秒数
EC_DCTIME_TICKSPERDAY	1日当たりのディストリビュートクロックのシステムタイムのナノ秒数

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

13.20 T_HFoe

「File access over EtherCAT」ハンドル。ハンドルを使用する前に、ファンクションブロック FB_EcFoeOpen [▶ 57]で一度、初期化する必要があります。この構造体型変数に直接ライトしてはいけません。

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

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

14 定数

14.1 グローバル定数

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;
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX (x86、x64、ARM)	Tc2_EtherCAT

14.2 ライブラリバージョン

すべてのライブラリに対して、固有のバージョン番号が付与されています。バージョンは、PLCライブラリリポジトリなどで確認できます。グローバル定数には、ライブラリバージョンに関する情報が含まれています。

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_EtherCAT : ST_LibVersion;
END_VAR
```

stLibVersion_Tc2_EtherCAT: Tc2_EtherCAT libraryのバージョン情報(ST_LibVersion型)

現在、保有しているバージョンが必要なバージョンかどうかチェックしてください。ファンクション F_CmpLibVersionを使用してください(Tc2_Systemライブラリで定義)。

TwinCAT 2で使用していたライブラリバージョンを比較する他のすべての選択肢は、TC3では使用できません。

14.3 EtherCATメールボックスプロトコルのエラーコード

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). *)
```

要件

開発環境	ターゲットシステム	必要なPLCライブラリ
TwinCAT v3.1.0	PCまたはCX（x86、x64、ARM）	Tc2_EtherCAT

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