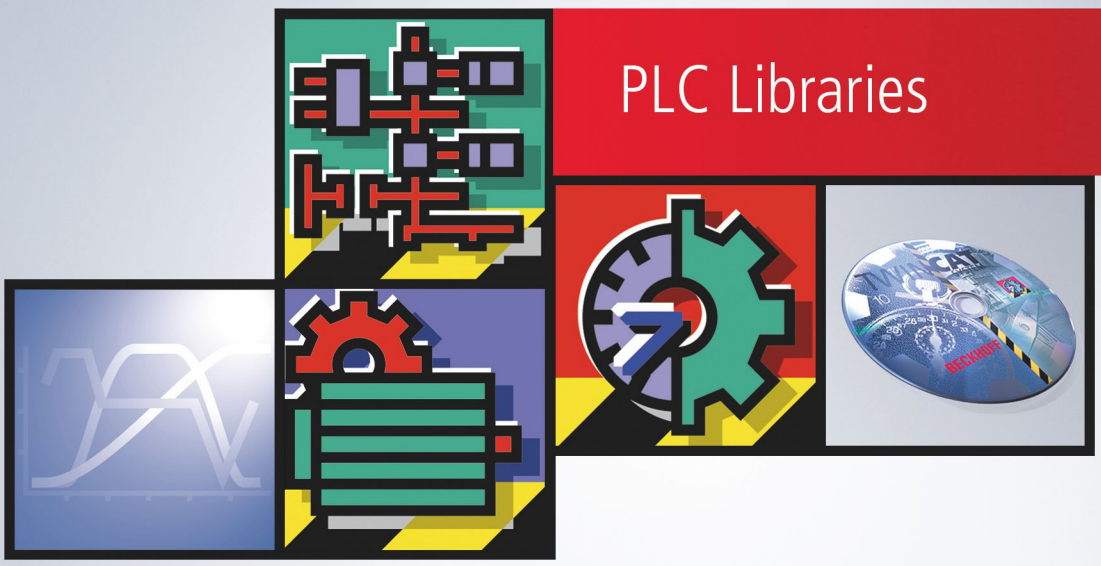
BECKHOFF New Automation Technology

Handbuch | DE

TX1200

TwinCAT 2 | PLC-Bibliothek: TcEnOcean



31.08.2023 | Version: 1.3

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

Disclaimer

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

Marken

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

Patente

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

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

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

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

Haftungsausschluss

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

Qualifikation des Personals

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

Signalwörter

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

Warnungen vor Personenschäden

⚠ GEFAHR

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

⚠ WARNUNG

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

⚠ VORSICHT

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

Warnung vor Umwelt- oder Sachschäden

HINWEIS

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

Information zum Umgang mit dem Produkt

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

1.3 Hinweise zur Informationssicherheit

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

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

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

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

2 Einleitung

Die TcEnOcean-Bibliothek ist eine TwinCAT SPS Bibliothek zum Datenaustausch mit EnOcean Geräten.

Diese Bibliothek ist nur in Verbindung mit einer EnOcean-Busklemme KL6021-0023 oder einer EnOcean-Masterklemme KL6581 einzusetzen.

3 Zielgruppen

Für den Nutzer dieser Bibliothek werden folgende Grundkenntnisse vorausgesetzt:

- TwinCAT PLC Control
- TwinCAT System Manager
- PC und Netzwerkkenntnisse
- Aufbau und Eigenschaften der Beckhoff Embedded-PC und deren Busklemmensystem
- Technologie von EnOcean Geräten
- Einschlägige Sicherheitsvorschriften der technischen Gebäudeausrüstung

Diese Softwarebibliothek ist für Gebäudeautomation-Systempartner der Beckhoff Automation GmbH & Co. KG. Die Systempartner sind tätig in dem Bereich Gebäudeautomation und beschäftigen sich mit Errichtung, Inbetriebsetzung, Erweiterung, Wartung und Service von mess-, steuer- und regelungstechnischen Anlagen der technischen Gebäudeausrüstung.

4 EnOcean

Technik

Die EnOcean-Funktechnik ermöglicht ein weit reichendes Signal mit geringen Mengen Umgebungsenergie. Mit 50 µWs kann ein serienmäßig EnOcean-Funkmodul ohne weiteres ein Signal über eine Distanz von 300 m (im Freifeld) übertragen. Die Signaldauer für ein EnOcean-Telegramm beträgt ca. 1 Tausendstel Sekunde.

- Lizenzfreies 868 MHz Frequenzband mit 1 % duty cycle
- Mehrfach-Telegrammaussendung mit Checksumme
- Kurze Telegramme (ca. 1 ms) führen zu geringer Kollisionswahrscheinlichkeit
- Hohe Reichweite: 30 m im Gebäude oder 300 m im Freifeld
- Repeater verfügbar für Erweiterungen
- Uni- und bidirektionale Kommunikation
- Hohe Datenübertragungsraten von 125 kbit/s
- Geringer "Daten-Overhead"
- ASK-Modulation
- Funkprotokoll ist definiert und in Modulen integriert
- Sensorprofile festgelegt und von Nutzern eingehalten
- Eindeutige Sende-ID (32 Bit)
- Keine Interferenz mit DECT-, WLAN-, PMR-Systemen etc.
- Systemdesign verifiziert in Industrieller Umgebung

Protokollaufbau

Protokoll	Beschreibung	Länge
ORG	Telegramm Typ	1 Byte
DB_3	Daten Byte 3	1 Byte
DB_2	Daten Byte 2	1 Byte
DB_1	Daten Byte 1	1 Byte
DB_0	Daten Byte 0	1 Byte
ID_3	Transmitter ID Byte 3	1 Byte
ID_2	Transmitter ID Byte 2	1 Byte
ID_1	Transmitter ID Byte 1	1 Byte
ID_0	Transmitter ID Byte 0	1 Byte
STATUS	Informationsstatus	1 Byte

4.1 Reichweitenplanung

Beachten Sie bei der Platzierung der EnOcean Geräte die Empfehlungen der EnOcean Allianz (siehe www.enocean.de). Das Einhalten der Empfehlungen unterstützt eine optimale Reichweite und Störunanfälligkeit.

Dämpfung von verschiedenen Materialien

Material	Dämpfung
Holz, Gips, Glas unbeschichtet (ohne Metall)	0...10 %
Backstein, Pressspanplatten	5...35 %
Beton mit Armierung aus Eisen	10...90 %
Metall, Aluminiumkaschierung	90..100 %

Reichweite

Material	Reichweite
Sichtverbindung	Typ 30 m in Gängen, bis zu 100 m in Hallen
Rigipswände/Holz	Typ 30 m, durch maximal 5 Wände
Ziegelwände/Gasbeton	Typ 20 m, durch maximal 3 Wände
Stahlbetonwände/-Decken	Typ 10 m, durch maximal 1 Wand/Decke

Platzierung des KL6583-Moduls

Die EnOcean-Sender und -Empfänger KL6583-0000 beinhaltet Sender, Empfänger sowie die Antenne.

Abstände

Der Abstand zu einer Stahlbetondecke sollte mindestens 50 cm betragen, der zu einer Wand 10 cm.

Das KL6583 Modul nicht auf einer Metall Platte verschrauben oder anbringen!

Umweltbedingungen

Des Weiteren sind die Umweltbedingungen einzuhalten:

- Luftfeuchtigkeit maximal 95 % ohne Betauung
- Umgebungstemperatur 0...55 °C

4.2 Zulassung von EnOcean-Funk-Technologie

Überprüfen Sie die Zulässigkeit des Betriebs in Ihrem Land

- **KL6583-0000**: **Europäische Union** und **Schweiz**
- **KL6583-0100**: **USA** und **Kanada**

KL6583-0100 für USA und Kanada

Enthält IC: 5731A-TCM320C

Enthält FCC ID: SZV-TCM320C

Das betroffene Gerät entspricht Teil 15 der FCC Regeln.
Der Betrieb unterliegt den folgenden Bedingungen:
(i.) dieses Gerät darf keine schädlichen Störungen verursachen und
(ii.) dieses Gerät muss alle empfangenen Störungen aufnehmen,
auch Störungen, die den Betrieb beeinträchtigen.

5 Integration in TwinCAT

5.1 KL6581 - Verknüpfung mit dem System Manager

1. Binden Sie das SPS-Programm ein und klicken Sie mit der rechten Maustaste auf die Datenstruktur.

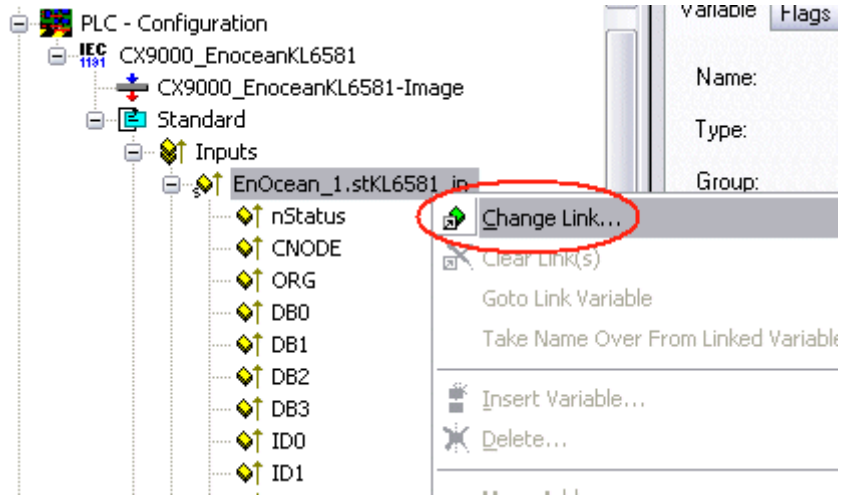


Bild 1

2. Wählen Sie "All Types" und "Continuous" an (siehe Bild 2).

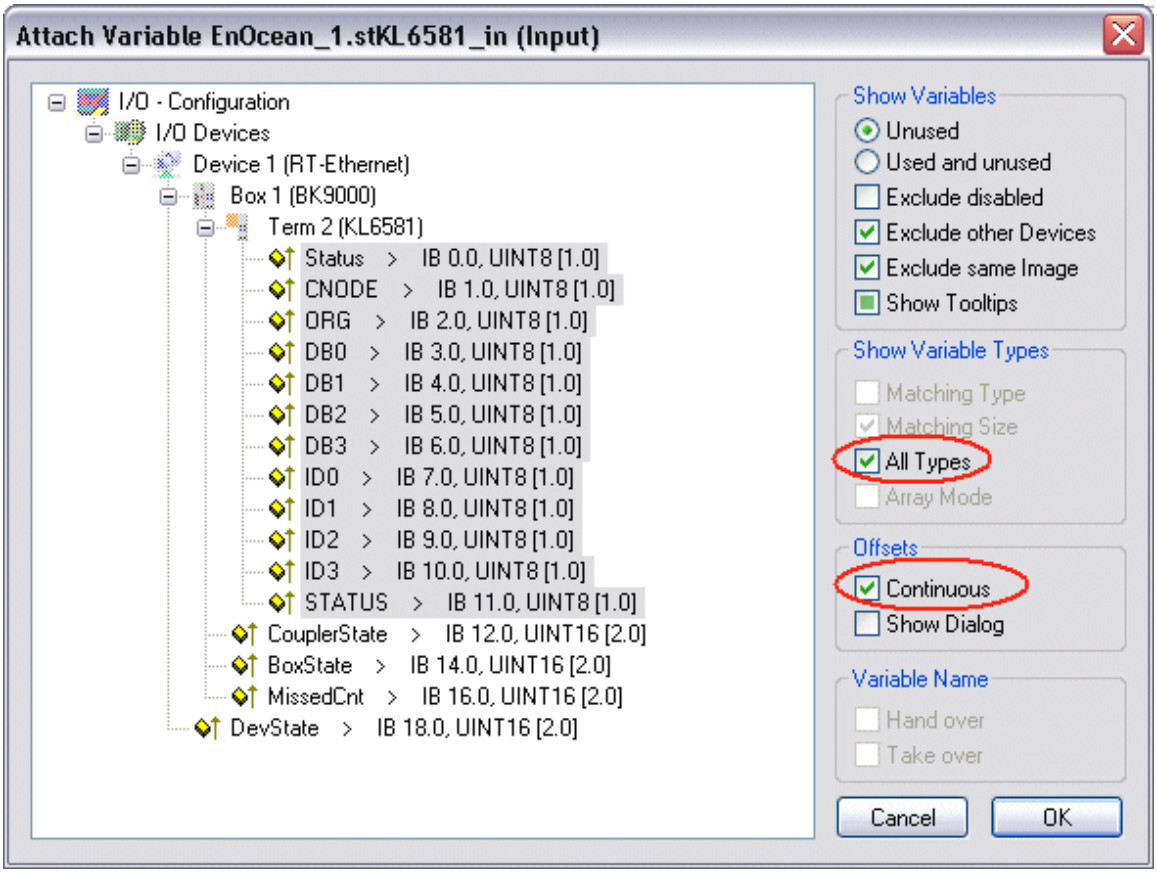


Bild 2

3. Klicken Sie mit der Maus auf die erste Variable der EnOcean-Masterklemme KL6581 "Status" an. Drücken Sie dann die >SHIFT< Taste und halten Sie diese gedrückt.

4. Gehen Sie mir dem Mauszeiger auf die letzte Variable der KL6581 "STATUS" und klicken Sie wiederum die linke Maustaste.
5. Jetzt lassen Sie die >SHIFT< Taste wieder los. Alle Daten der Klemme müssen jetzt markiert sein (siehe Bild 2).
6. Anschließend den OK Button betätigen.
7. Kontrollieren Sie die Verknüpfungen
   Gehen Sie dazu auf die KL6581 und öffnen Sie diese. Alle Daten der Klemme müssen jetzt mit einem kleinen Pfeil markiert sein (siehe Bild 3). Ist dies der Fall, fahren Sie genauso mit den Ausgängen fort.

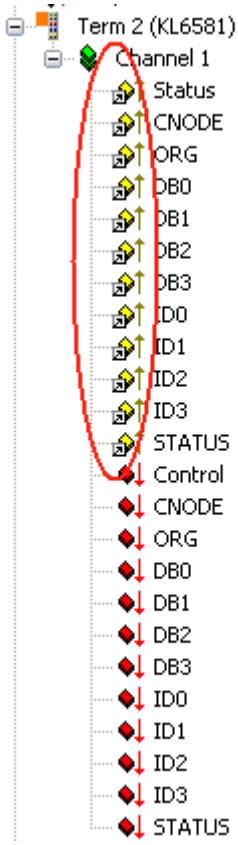


Bild 3

5.2 KL6021-0023 - Verknüpfung mit dem System Manager

1. Binden Sie das SPS-Programm ein und klicken Sie mit der rechten Maustaste auf die Datenstruktur.

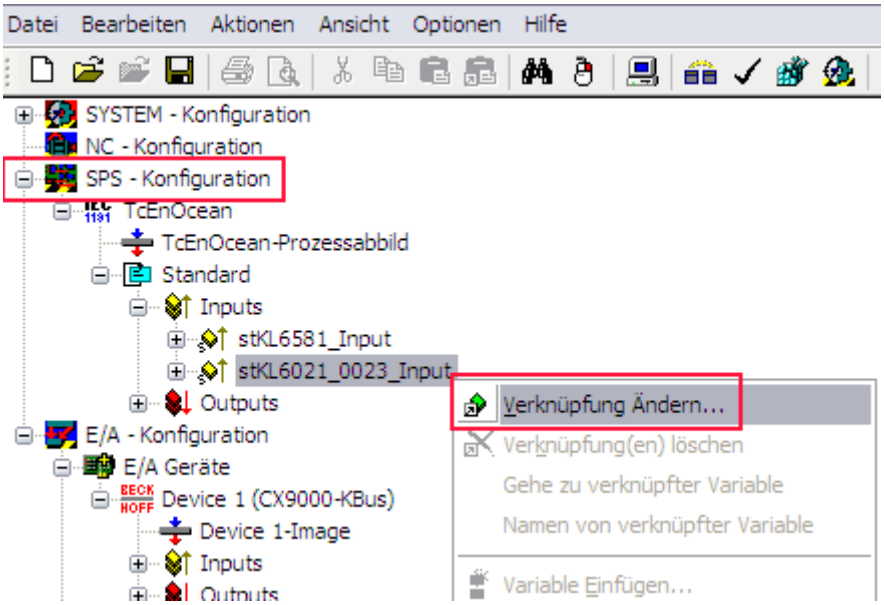


Bild 1

2. Wählen Sie "Alle Typen" und "Kontinuierlich" an (siehe Bild 2).

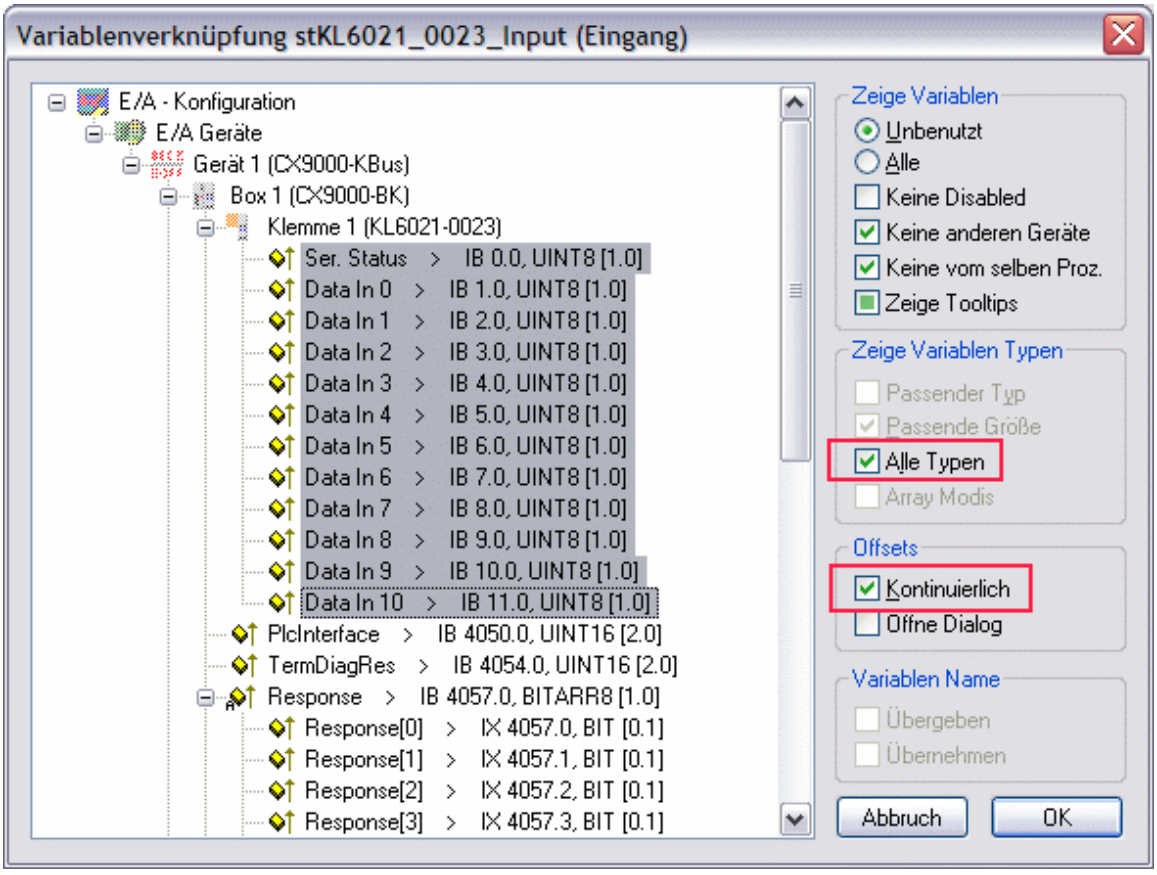


Bild 2

3. Klicken Sie mit der Maus auf die erste Variable der EnOcean-Busklemme KL6021-0023 "Status" an. Drücken Sie dann die >SHIFT< Taste und halten Sie diese gedrückt.
4. Gehen Sie mir dem Mauszeiger auf die letzte Variable der KL6021-0023 "STATUS" und klicken Sie wiederum die linke Maustaste.
5. Jetzt lassen Sie die >SHIFT< Taste wieder los. Alle Daten der Klemme müssen jetzt markiert sein (siehe Bild 2).

6. Anschließend den OK Button betätigen.
7. Kontrollieren Sie die Verknüpfungen
   Gehen Sie dazu auf die KL6021-0023 und öffnen Sie diese. Alle Daten der Klemme müssen jetzt mit einem kleinen Pfeil markiert sein (siehe Bild 3). Ist dies der Fall, fahren Sie genauso mit den Ausgängen fort.

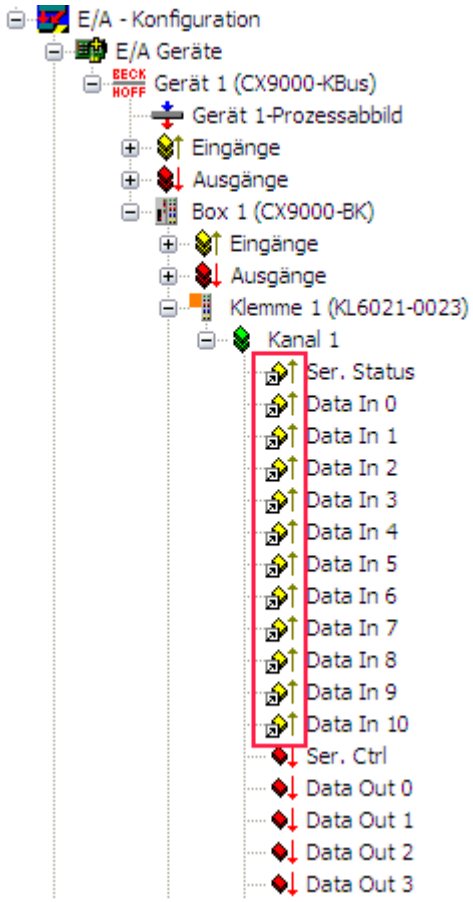


Bild 3

5.3 Integration in TwinCAT (CX9020)

Dieses Beispiel beschreibt, wie ein einfaches SPS-Programm für EnOcean in TwinCAT geschrieben werden kann und wie es mit der Hardware verknüpft wird. Es sollen die vier Tastersignale eines EnOcean Funkschaltmoduls empfangen werden.

https://infosys.beckhoff.com/content/1031/tcplclibenocean/Resources/11985702027.zip https://infosys.beckhoff.com/content/1031/tcplclibenocean/Resources/11985702027.zip

Hardware

Einrichtung der Komponenten

Es wird folgende Hardware benötigt:

- 1x Embedded-PC CX9020
- 1x EnOcean-Masterklemme KL6581
- 1x EnOcean-Sender und -Empfänger KL6583-0000
- 1x Endklemme KL9010

Richten Sie die Hardware sowie die EnOcean-Komponenten wie in den entsprechenden Dokumentationen beschrieben ein.

Dieses Beispiel geht davon aus, dass die Id vom Funkschaltmodul bekannt ist.

Software

Erstellung des SPS-Programms

Erstellen Sie ein neues SPS-Projekt für PC-basierte Systeme (ARM) und fügen die Bibliothek *TcEnOcean.lib* hinzu.

Erzeugen Sie als Nächstes die folgenden globalen Variablen:

```
VAR_GLOBAL
    stKL6581Input  AT %I* : KL6581_Input;
    stKL6581Output AT %Q* : KL6581_Output;
    stKL6581              : STR_KL6581;
END_VAR
```

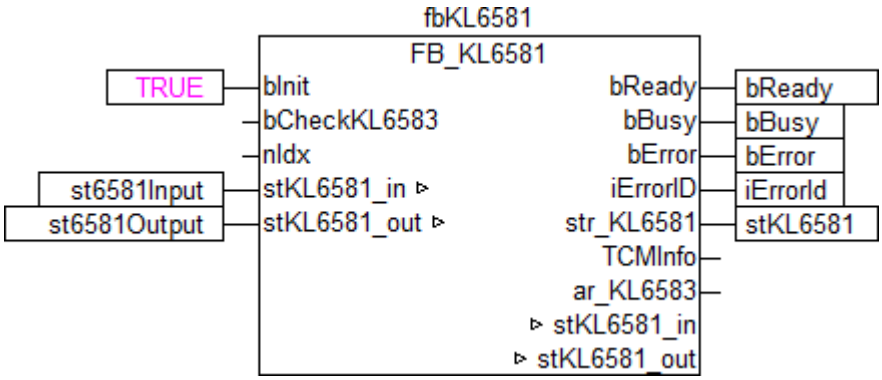
stKL6581Input: Eingangsvariable [▶ 46] für die EnOcean-Klemme.

stKL6581Output: Ausgangsvariable [▶ 47] für die EnOcean-Klemme.

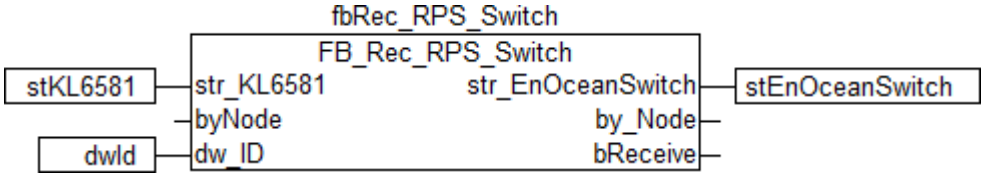
stKL6581: Wird für die Kommunikation [▶ 48] mit EnOcean benötigt.

Alle Bausteine bei EnOcean müssen in einer Task ausgeführt werden.

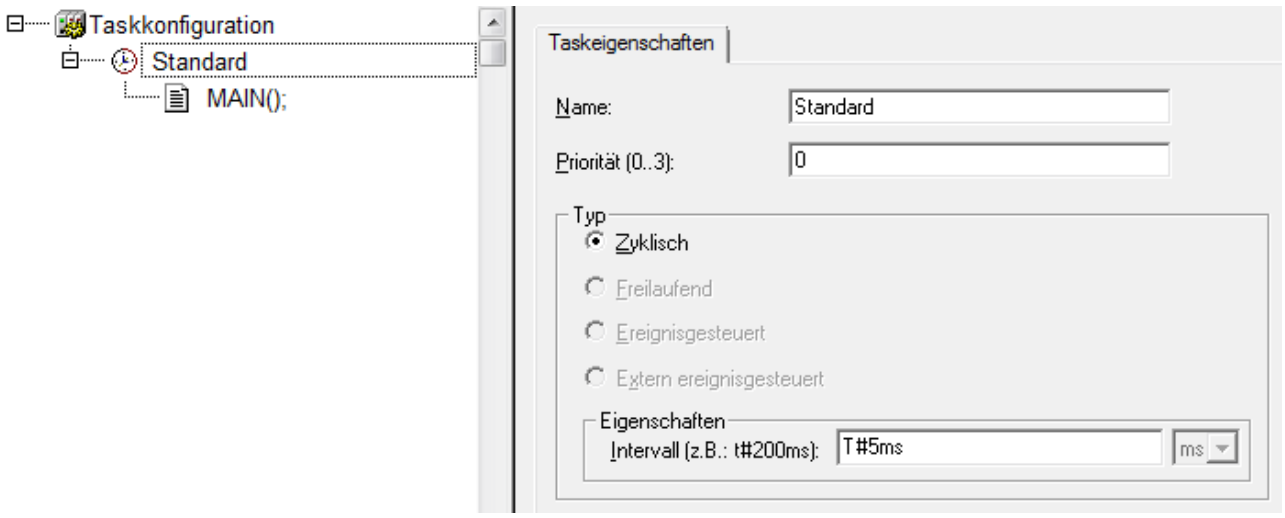
Legen Sie daher ein MAIN-Programm (CFC) an in dem die Bausteine FB_KL6581() [▶ 24] und FB_Rec_RPS_Switch() [▶ 26] aufgerufen werden. Achten Sie beim Kommunikationsbaustein darauf, mit *stKL6581Input*, *stKL6581Output* und *stKL6581* zu verknüpfen.



Der Eingang *dw_ID* des Empfangsbausteins wird mit der lokalen Variable *dwId* (Id vom Funkschaltmodul) verknüpft und *str_KL6581* mit der globalen Variable *stKL6581*.



Gehen Sie in die Taskkonfiguration und geben Sie der Task eine niedrigere Intervall-Zeit. Genauere Informationen dazu finden Sie in der Beschreibung des Bausteins FB_KL6581() [▶ 24].

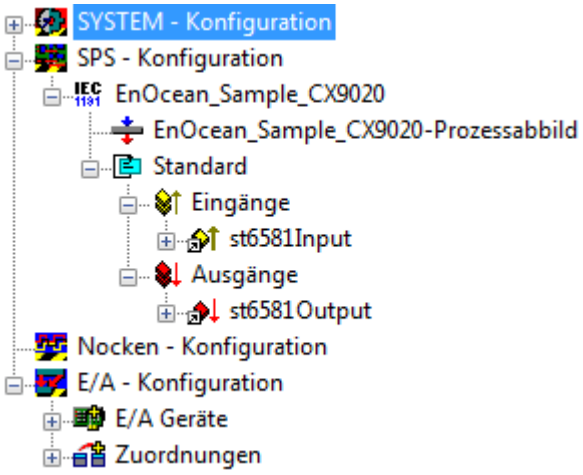


Laden Sie das Projekt als Bootprojekt auf den CX und speichern Sie es ab.

Konfiguration im System Manager

Legen Sie ein neues TwinCAT System-Manager-Projekt an, wählen Sie als Zielsystem den CX und lassen Sie nach dessen Hardware suchen.

Fügen Sie das oben angelegte SPS-Programm unter SPS-Konfiguration hinzu.



Verknüpfen Sie die globalen Variablen des SPS-Programms nun mit den Ein- und Ausgängen der Busklemmen, erzeugen Sie die Zuordnungen und aktivieren Sie die Konfiguration. Starten Sie dann das Gerät im Run-Modus.
Ihr CX ist jetzt einsatzbereit.

Nach Betätigen der Taster am Funkschaltmodul können die empfangenen Funksignale in der Struktur *stEnOceanSwitch* eingesehen werden.

5.4 Integration in TwinCAT (BC9191)

Dieses Beispiel beschreibt, wie ein einfaches SPS-Programm für EnOcean in TwinCAT geschrieben werden kann und wie es mit der Hardware verknüpft wird. Es sollen die vier Tastersignale eines EnOcean Funkschaltmoduls empfangen werden.

https://infosys.beckhoff.com/content/1031/tcplclibenocean/Resources/11985703435.zip https://infosys.beckhoff.com/content/1031/tcplclibenocean/Resources/11985703435.zip

Hardware

Einrichtung der Komponenten

Es wird folgende Hardware benötigt:

- 1x Busklemmen Controller BC9191
- 1x EnOcean-Sender und -Empfänger KL6583-0000

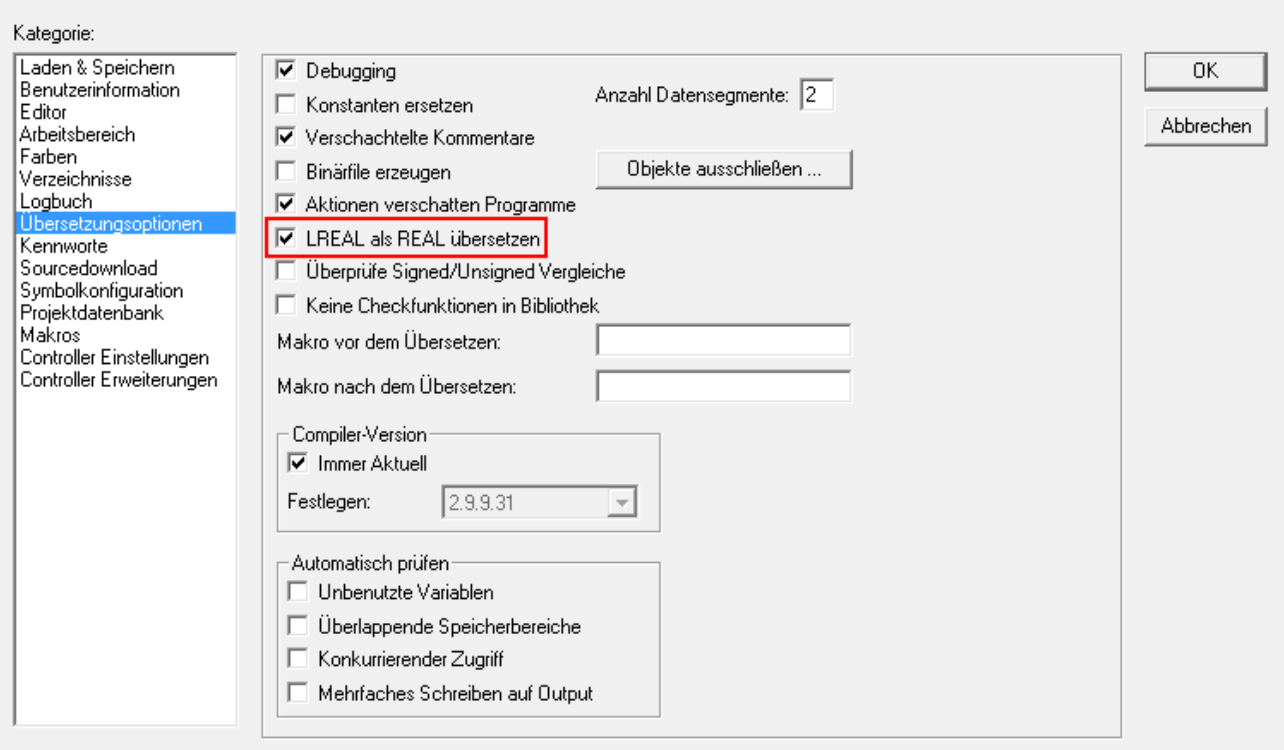
Richten Sie die Hardware sowie die EnOcean-Komponenten wie in den entsprechenden Dokumentationen beschrieben ein.

Dieses Beispiel geht davon aus, dass die Id vom Funkschaltmodul bekannt ist.

Software

Erstellung des SPS-Programms

Erstellen Sie ein neues SPS-Projekt für BC-basierte Systeme (BCxx50 über AMS) und fügen die Bibliotheken *TcEnOcean.lbx* und *TcSystemBCxx50.lbx* hinzu. Gehen Sie danach im Menü auf *Projekt → Optionen... → Übersetzungsoptionen* und wählen *LREAL als REAL übersetzen* an.



Erzeugen Sie als Nächstes die folgenden globalen Variablen:

```
VAR_GLOBAL
    stKL6581Input  AT %I* : KL6581_Input;
    stKL6581Output AT %Q* : KL6581_Output;
    stKL6581              : STR_KL6581;
END_VAR
```

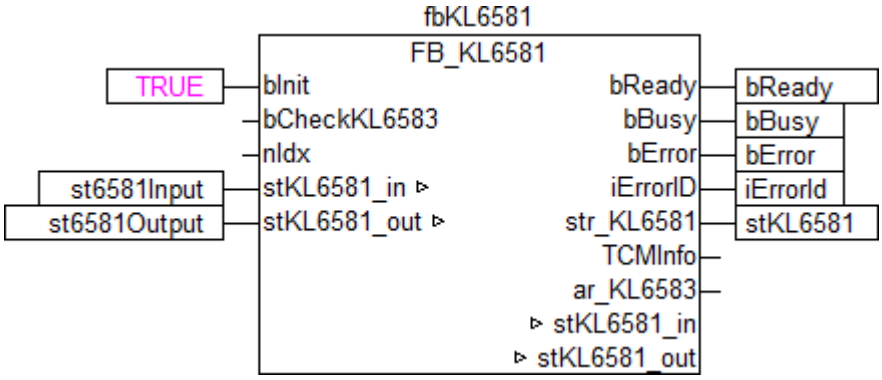
stKL6581Input: Eingangsvariable [▶ 46] für EnOcean.

stKL6581Output: Ausgangsvariable [▶ 47] für EnOcean.

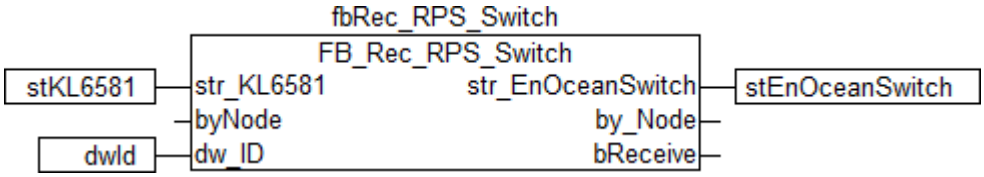
stKL6581: Wird für die Kommunikation [▶ 48] mit EnOcean benötigt.

Alle Bausteine bei EnOcean müssen in einer Task ausgeführt werden.

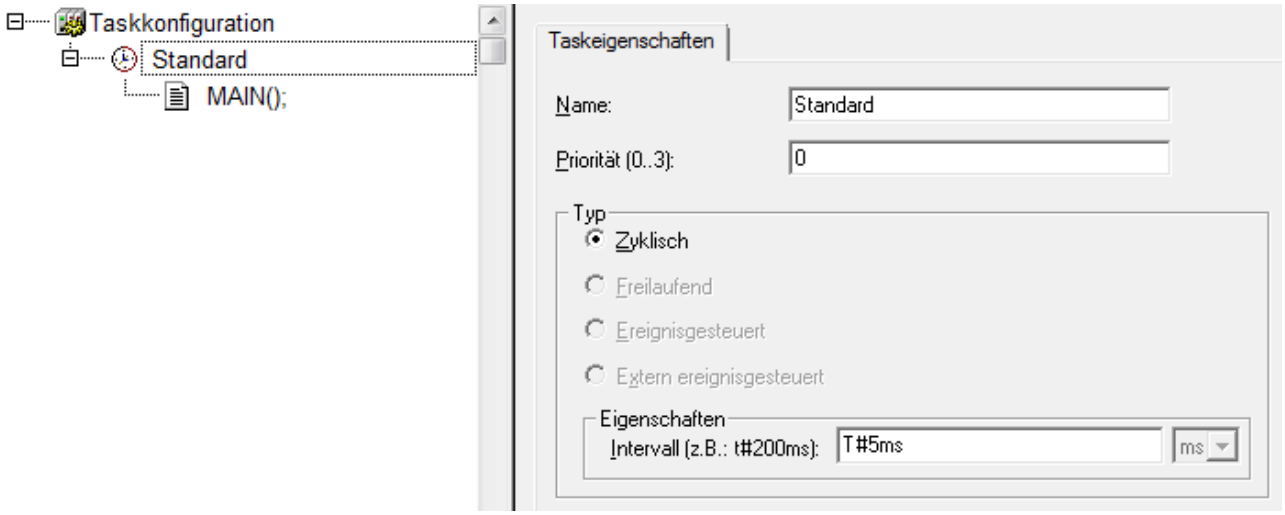
Legen Sie daher ein MAIN-Programm (CFC) an in dem die Bausteine FB_KL6581() [▶ 24] und FB_Rec_RPS_Switch() [▶ 26] aufgerufen werden. Achten Sie beim Kommunikationsbaustein darauf, mit *stKL6581Input*, *stKL6581Output* und *stKL6581* zu verknüpfen.



Der Eingang *dw_ID* des Empfangsbausteins wird mit der lokalen Variable *dwId* (Id vom Funkschaltmodul) verknüpft und *str_KL6581* mit der globalen Variable *stKL6581*.



Gehen Sie in die Taskkonfiguration und geben Sie der Task eine niedrigere Intervall-Zeit. Genauere Informationen dazu finden Sie in der Beschreibung des Bausteins FB_KL6581() [▶ 24].

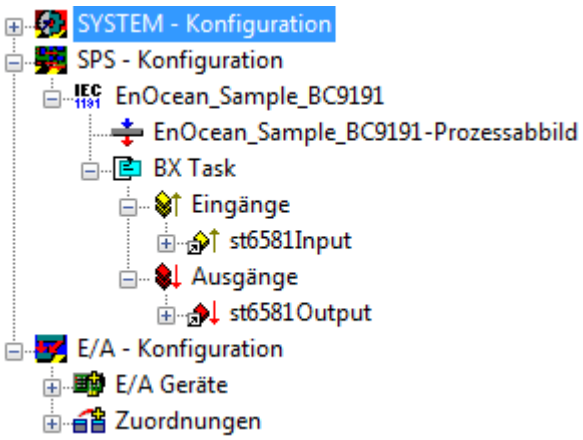


Laden Sie das Projekt als Bootprojekt auf den BC und speichern Sie es ab.

Konfiguration im System Manager

Legen Sie ein neues System-Manager-Projekt an, wählen Sie als Zielsystem den BC und lassen Sie nach dessen Hardware suchen.

Fügen Sie das oben angelegte SPS-Programm unter SPS-Konfiguration hinzu.



Verknüpfen Sie die globalen Variablen des SPS-Programms nun mit den Ein- und Ausgängen der EnOcean-Masterklemme KL6581 vom BC9191, erzeugen Sie die Zuordnungen und aktivieren Sie die Konfiguration. Starten Sie dann das Gerät im Run-Modus.

Ihr BC ist jetzt einsatzbereit.

Nach Betätigen der Taster am Funkschaltmodul können die empfangenen Funksignale in der Struktur *stEnOceanSwitch* eingesehen werden.

6 Programmierung

Inhalt

KL6581

Bausteine	Beschreibung
FB_KL6581 [▶ 24]	Kommunikation mit einer EnOcean-Masterklemme KL6581

Function

Bausteine	Beschreibung
F_Byte_to_Temp [▶ 33]	Diese Funktion wandelt einen Byte-Rohwert in eine REAL-Variable um.
F_Byte_to_TurnSwitch [▶ 34]	Diese Funktion wandelt einen Byte Rohwert in ein Bool Array um.

Other

Bausteine	Beschreibung
FB_EnOcean_Search [▶ 31]	Baustein erkennt alle EnOcean-Teilnehmer in seiner Reichweite und zeigt diese an.
FB_Rec_Teach_In [▶ 32]	Dieser Baustein zeigt an wenn in einem EnOcean-Telegramm das LRN Bit gesetzt ist unabhängig seiner EnOcean-ID.
FB_Rec_Teach_In_Ex [▶ 32]	Dieser Baustein zeigt an wenn in einem EnOcean-Telegramm das LRN Bit gesetzt ist unabhängig seiner EnOcean-ID.

Read

Bausteine	Beschreibung
FB_Rec_1BS [▶ 26]	Empfängt Daten mit ORG-Telegramm 6.Typisches EnOcean-Gerät: Fensterkontakt.
FB_Rec_Generic [▶ 25]	Empfängt alle Arten von EnOcean-Telegrammen.
FB_Rec_RPS_Switch [▶ 26]	Empfängt Daten mit ORG-Telegramm 5. Typisches EnOcean-Gerät: Taster.
FB_Rec_RPS_Window_Handle [▶ 27]	Empfängt Daten mit ORG-Telegramm 5. Typisches EnOcean-Gerät: Fenstergriff.

Send

Bausteine	Beschreibung
FB_Send_Generic [▶ 28]	Sendet beliebige EnOcean-Telegramme.
FB_Send_4BS [▶ 29]	Sendet EnOcean-Telegramme im 4BS Format.
FB_Send_RPS_Switch [▶ 29]	Sendet EnOcean-Telegramme im Format eines Tasters.
FB_Send_RPS_SwitchAuto [▶ 30]	Sendet EnOcean-Telegramme im Format eines Tasters.

Enums

Datentypen	Beschreibung
E_EnOcean_Org [▶ 45]	Typ des EnOcean Telegramms.
E_KL6581_Err [▶ 45]	Fehlermeldungen.

Structs

Datentypen	Beschreibung
KL6581_Input [▶ 46]	Prozessabbild der Eingänge der EnOcean-Masterklemme KL6581
KL6581_Output [▶ 47]	Prozessabbild der Ausgänge der EnOcean-Masterklemme KL6581
AR_EnOceanWindow [▶ 49]	Zustand des Fensters.
STR_EnOceanSwitch [▶ 47]	Zustand der Taster.
STR_KL6581 [▶ 48]	Interne Struktur.
STR_Teach [▶ 48]	Datenstruktur Hersteller ID, Typ und Funktion.
STR_Teach_In [▶ 49]	Datenstruktur Hersteller ID, Typ und Profil.
STREnOceanTurnSwitch [▶ 49]	Stellung des Drehschalters am Raumbediengerät.

KL6021-0023

Bausteine	Beschreibung
FB_EnOceanReceive [▶ 35]	Kommunikation mit einer EnOcean-Busklemme KL6021-0023

Read

Bausteine	Beschreibung
FB_EnOceanPTM100 [▶ 36]	Empfängt die Signale eines PTM100-Moduls.
FB_EnOceanPTM200 [▶ 38]	Empfängt die Signale eines PTM200-Moduls.
FB_EnOceanSTM100 [▶ 40]	Empfängt die Signale eines STM100-Moduls (veraltet).
FB_EnOceanSTM100Generic [▶ 42]	Empfängt die Signale eines STM100-Moduls.
FB_EnOceanSTM250 [▶ 44]	Empfängt die Signale eines STM250-Moduls.

Enums

Datentypen	Beschreibung
E_EnOceanRotarySwitch [▶ 49]	Stellung des Drehschalters am Raumbediengerät.
E_EnOceanSensorType [▶ 50]	Sensorentyp.

Structs

Datentypen	Beschreibung
ST_EnOceanInData [▶ 50]	Prozessabbild der Eingänge der EnOcean-Busklemme KL6021-0023
ST_EnOceanOutData [▶ 50]	Prozessabbild der Ausgänge der EnOcean-Busklemme KL6021-0023
ST_EnOceanReceivedData [▶ 51]	Interne Struktur.

6.1 Allgemeine Informationen

Installation

Ab Version TwinCAT 2.11 Build 2229 (R3 und x64 Engineering) werden die Bibliotheken "TcEnOcean.lib/.lb6/.lbx" standardmäßig mitinstalliert.

Name der Bibliothek

Diese Bibliothek ersetzt die "TcKL6581.lib/.lb6./.lbx". Es hat sich nur der Name der Bibliothek geändert. Die Bausteine sind kompatibel.

Hardware Dokumentation im Beckhoff Information System: KL6021-0023, KL6023 - EnOcean-Busklemmen

Hardware Dokumentation im Beckhoff Information System: KL6581, KL6583 - EnOcean-Busklemmen

Weitere erforderliche Bibliotheken

Für PC-Systeme (x86) und Embedded-PCs (CXxxxx):

- Standard.lib
- TcBase.lib
- TcSystem.lib

Für Busklemmen-Controller der Serie BCxx00:

- Standard.lb6
- PlcHelperBC.lb6

Für Busklemmen-Controller der Serie BCxx50, BCxx20 und BC9191:

- Standard.lbx
- TcBaseBCxx50.lbx
- TcSystemBCxx50.lbx

Für Busklemmen-Controller der Serie BXxx00:

- Standard.lbx
- TcBaseBX.lbx
- TcSystemBX.lbx

Speicherauslastung

Durch Einbinden der Bibliothek wird bereits SPS-Programmspeicher verbraucht. Abhängig vom Applikationsprogramm kann daher der verbleibende Speicher nicht ausreichend sein.

6.2 KL6581

Bausteine	Beschreibung
FB_KL6581 [▶ 24]	Kommunikation mit einer EnOcean-Masterklemme KL6581

Function

Bausteine	Beschreibung
F_Byte_to_Temp [▶ 33]	Diese Funktion wandelt einen Byte-Rohwert in eine REAL-Variable um.
F_Byte_to_TurnSwitch [▶ 34]	Diese Funktion wandelt einen Byte Rohwert in ein Bool Array um.

Other

Bausteine	Beschreibung
FB_EnOcean_Search [▶ 31]	Baustein erkennt alle EnOcean-Teilnehmer in seiner Reichweite und zeigt diese an.
FB_Rec_Teach_In [▶ 32]	Dieser Baustein zeigt an wenn in einem EnOcean-Telegramm das LRN Bit gesetzt ist unabhängig seiner EnOcean-ID.
FB_Rec_Teach_In_Ex [▶ 32]	Dieser Baustein zeigt an wenn in einem EnOcean-Telegramm das LRN Bit gesetzt ist unabhängig seiner EnOcean-ID.

Read

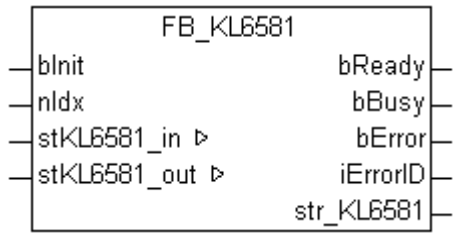
Bausteine	Beschreibung
FB_Rec_1BS [▶ 26]	Empfängt Daten mit ORG-Telegramm 6.Typisches EnOcean-Gerät: Fensterkontakt.
FB_Rec_Generic [▶ 25]	Empfängt alle Arten von EnOcean-Telegrammen.

Bausteine	Beschreibung
FB_Rec_RPS_Switch [▶ 26]	Empfängt Daten mit ORG-Telegramm 5. Typisches EnOcean-Gerät: Taster.
FB_Rec_RPS_Window_Handle [▶ 27]	Empfängt Daten mit ORG-Telegramm 5. Typisches EnOcean-Gerät: Fenstergriff.

Send

Bausteine	Beschreibung
FB_Send_Generic [▶ 28]	Sendet beliebige EnOcean-Telegramme.
FB_Send_4BS [▶ 29]	Sendet EnOcean-Telegramme im 4BS Format.
FB_Send_RPS_Switch [▶ 29]	Sendet EnOcean-Telegramme im Format eines Tasters.
FB_Send_RPS_SwitchAuto [▶ 30]	Sendet EnOcean-Telegramme im Format eines Tasters.

6.2.1 FB_KL6581



Dieser Funktionsbaustein übernimmt die Kommunikation mit der EnOcean-Masterklemme KL6581. Über diesen Baustein wird die KL6581 konfiguriert und der Datenaustausch mit dem EnOcean-Netzwerk gestartet.

Einschränkungen

- Nur ein Aufruf pro Instanz
- Aufruf muss einmal pro PLC-Zyklus erfolgen
- Instanz muss in derselben PLC-Task aufgerufen werden, wie die ihm zugeordneten Sende- und Empfangsbausteine
- Maximal 64 Instanzen pro PLC-Projekt zulässig

VAR_INPUT

```
bInit          : BOOL;
nIdx           : USINT := 1;
```

bInit: Aktiviert den Baustein, der als erstes die KL6581 konfiguriert und anschließend in den Datenaustausch versetzt, solange am Eingang ein TRUE ansteht.

nIdx: Die idx Nummer muss beim Einsatz von mehr als einer Busklemme pro SPS-Programm für jede KL6581 eindeutig sein (gültige Werte: 1...64).

VAR_OUTPUT

```
bReady         : BOOL;
bBusy          : BOOL;
bError         : BOOL;
iErrorID       : E_KL6581_Err;
str_KL6581     : STR_KL6581;
```

bReady: Der Baustein ist bereit Daten zu senden und zu empfangen.

bBusy: Der Baustein ist aktiv.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorID* beschrieben.

iErrorID: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

str_KL6581: Datenstruktur die mit den Sende- und Empfangsbausteinen verbunden wird (siehe STR_KL6581 [▶ 48]).

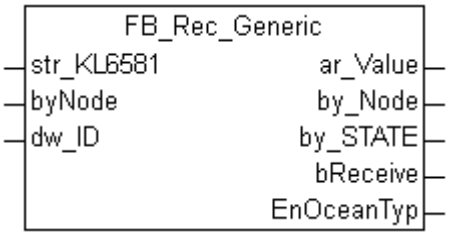
VAR_IN_OUT

```
stKL6581_in       : KL6581_Input;
stKL6581_out      : KL6581_Output;
```

stKL6581_in: Wird mit den Eingangsadressen der KL6581 im System Manager verknüpft (siehe KL6581_Input [▶ 46]).

stKL6581_out: Wird mit den Ausgangsadressen der KL6581 im System Manager verknüpft (siehe KL6581_Output [▶ 47]).

6.2.2 FB_Rec_Generic



Dieser Funktionsbaustein empfängt alle Daten, die über EnOcean empfangen wurden. Dieser Baustein kann für alle Arten von EnOcean-Telegrammen verwendet werden.

Die Daten muss der Anwender selbst interpretieren. Dazu ist die Dokumentation des Herstellers des sendenden EnOcean-Gerätes notwendig.

VAR_INPUT

```
str_KL6581   : STR_KL6581;
byNode       : BYTE;
dw_ID        : DWORD;
```

str_KL6581: Datenstruktur [▶ 48] die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

dw_ID: EnOcean-ID die empfangen werden soll.

VAR_OUTPUT

```
ar_Value     : ARRAY [0..3] OF BYTE;
by_Node      : BYTE;
by_STATE     : BYTE;
bReceive     : BOOL := TRUE;
EnOceanTyp   : E_EnOcean_Org;
```

ar_Value: 4 Byte EnOcean-Daten.

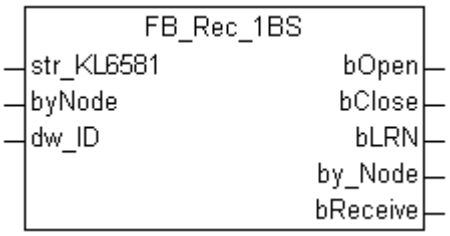
by_Node: Node Nummer der KL6583 die das EnOcean-Telegramm empfangen hat.

by_STATE: EnOcean STATUS Feld.

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

EnOceanTyp: EnOcean ORG Feld.

6.2.3 FB_Rec_1BS



Dieser Funktionsbaustein empfängt Daten, die über EnOcean empfangen wurden. Dieser Baustein wird zum Beispiel zur Anbindung von Fensterkontakten verwendet.

(ORG FIELD 6)

VAR_INPUT

```
str_KL6581   : STR_KL6581;
byNode       : BYTE;
dw_ID        : DWORD;
```

str_KL6581: Datenstruktur [▶ 48] die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

dw_ID: EnOcean-ID die empfangen werden soll.

VAR_OUTPUT

```
bOpen        : BOOL;
bClose       : BOOL;
bLRN         : BOOL;
by_Node      : BYTE;
bReceive     : BOOL := TRUE;
```

bOpen: Kontakt offen.

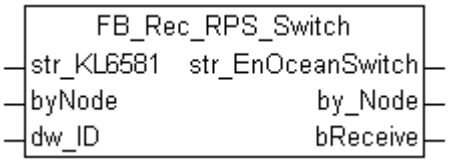
bClose: Kontakt geschlossen.

bLRN: LRN Taste gedrückt.

by_Node: Node Nummer der KL6583 die das EnOcean Telegramm empfangen hat.

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

6.2.4 FB_Rec_RPS_Switch



Dieser Funktionsbaustein empfängt Daten eines Schalters, die über EnOcean empfangen wurden. Der Baustein gibt die Daten in einer Datenstruktur aus.

ORG Field 5

VAR_INPUT

```
str_KL6581      : STR_KL6581;
byNode          : BYTE;
dw_ID           : DWORD;
```

str_KL6581: Datenstruktur [▶ 48] die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

dw_ID: EnOcean-ID die empfangen werden soll.

VAR_OUTPUT

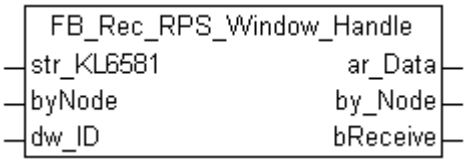
```
str_EnOceanSwitch  : STR_EnOceanSwitch;
by_Node            : BYTE;
bReceive           : BOOL := TRUE;
```

str_EnOceanSwitch: Daten des Schalters (siehe STR_EnOceanSwitch [▶ 47]).

by_Node: Node Nummer der KL6583 die das EnOcean Telegramm empfangen hat.

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

6.2.5 FB_Rec_RPS_Window_Handle



Dieser Funktionsbaustein empfängt Daten eines Fenstergriffes (WINDOW HANDLE), die über EnOcean empfangen wurden. Der Baustein gibt die Daten in einer Datenstruktur aus.

ORG Field 5

VAR_INPUT

```
str_KL6581  : STR_KL6581;
byNode      : BYTE;
dw_ID       : DWORD;
```

str_KL6581: Datenstruktur [▶ 48] die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

dw_ID: EnOcean-ID die empfangen werden soll.

VAR_OUTPUT

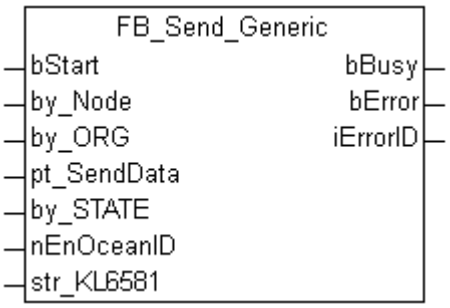
```
ar_Data     : AR_EnOceanWindow;
by_Node     : BYTE;
bReceive    : BOOL := TRUE;
```

dw_ID: Daten vom Fenstergriff (siehe AR_EnOceanWindow [▶ 49]).

by_Node: Node Nummer der KL6583 die das EnOcean-Telegramm empfangen hat.

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

6.2.6 FB_Send_Generic



Dieser Funktionsbaustein sendet Daten über EnOcean. Die Art und der Dateninhalt sind beliebig. Mit diesem Baustein können alle Arten von EnOcean-Daten-Telegrammen versendet werden.

VAR_INPUT

```
bStart        : BOOL;
by_Node       : BYTE;
by_ORG        : E_EnOcean_Org;
pt_SendData   : DWORD;
by_STATE      : BYTE;
nEnOceanID    : BYTE;
str_KL6581    : STR_KL6581;
```

bStart: Positive Flanke sendet die Daten.

by_Node: Adresse der EnOcean-Sender und -Empfänger KL6583-0000 an die das Telegramm gesendet werden soll (gültige Werte: 1...8).

by_ORG: ORG Field des EnOcean Telegramms.

pt_SendData: Pointer auf die Daten die gesendet werden sollen, mit ADR wird die Pointeradresse ermittelt. Der Pointer auf eine 4 Byte Variable zeigen.

by_STATE: EnOcean STATE, kann vom TCM Modul verändert werden.

nEnOceanID: Virtuelle EnOcean-ID, auf die reale EnOcean-ID wird ein Wert von 0...127 auf addiert (gültige Werte: 0...127).

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

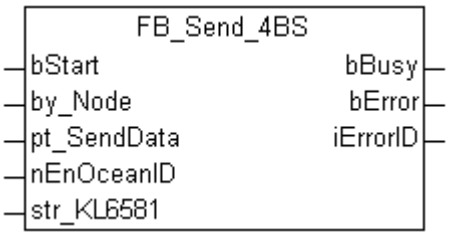
```
bBusy         : BOOL;
bError        : BOOL;
iErrorID      : E_KL6581_Err;
```

bBusy: Baustein ist aktiv, es können noch keine neuen Daten gesendet werden.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorID* beschrieben.

iErrorID: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

6.2.7 FB_Send_4BS



Dieser Funktionsbaustein sendet Daten über EnOcean. Das ORG Field ist fest auf 7 eingestellt.

VAR_INPUT

```
bStart       : BOOL;
by_Node      : BYTE;
pt_SendData  : DWORD;
nEnOceanID   : BYTE;
str_KL6581   : STR_KL6581;
```

bStart: Positive Flanke sendet die Daten.

by_Node: Adresse der EnOcean-Sender und -Empfänger KL6583-0000 an die das Telegramm gesendet werden soll (gültige Werte: 1...8).

pt_SendData: Pointer auf die Daten die gesendet werden sollen, mit ADR wird die Pointeradresse ermittelt. Der Pointer auf eine 4 Byte Variable zeigen.

nEnOceanID: Virtuelle EnOcean-ID, auf die reale EnOcean-ID wird ein Wert von 0...127 auf addiert (gültige Werte: 0...127).

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

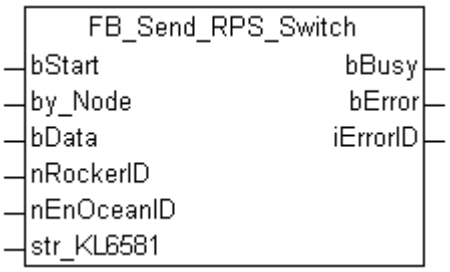
```
bBusy        : BOOL;
bError       : BOOL;
iErrorID     : E_KL6581_Err;
```

bBusy: Baustein ist aktiv, es können noch keine neuen Daten gesendet werden.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorID* beschrieben.

iErrorID: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

6.2.8 FB_Send_RPS_Switch



Dieser Baustein sendet EnOcean-Telegramme im Format eines Tasters. Mit der positiven Flanke von *bStart* wird der Wert von *bData* gesendet. Um einen Tastendruck zu simulieren muss der Baustein üblicherweise 2-mal gestartet werden, einmal mit *bData* = TRUE, und einmal mit *bData* = FALSE. Für eine einfachere Handhabung kann der Baustein FB_Send_RPS_SwitchAuto() [▶ 30] verwendet werden.

VAR_INPUT

```
bStart      : BOOL;
by_Node     : BYTE;
bData       : BOOL;
nRockerID   : INT;
nEnOceanID  : BYTE;
str_KL6581  : STR_KL6581;
```

bStart: Positive Flanke sendet die Daten.

by_Node: Adresse der EnOcean-Sender und -Empfänger KL6583-0000 an die das Telegramm gesendet werden soll (gültige Werte: 1...8).

bData: Wert der Übertragen werden soll.

nRockerID: Tasternummer, gültige Werte 0..3.

nEnOceanID: Virtuelle EnOcean-ID, auf die reale EnOcean-ID wird ein Wert von 0...127 auf addiert (gültige Werte: 0...127).

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

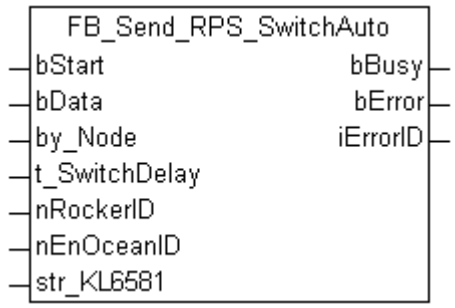
```
bBusy       : BOOL;
bError      : BOOL;
iErrorID    : E_KL6581_Err;
```

bBusy: Baustein ist aktiv, es können noch keine neuen Daten gesendet werden.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorID* beschrieben.

iErrorID: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

6.2.9 FB_Send_RPS_SwitchAuto



Dieser Baustein sendet EnOcean-Telegramme im Format eines Tasters. Mit der positiven Flanke von *bStart* wird der Wert von *bData* gesendet. Nach Ablauf der Zeit *t_SwitchDelay* wird das Signal "Taster loslassen" gesendet.

VAR_INPUT

```
bStart          : BOOL;
bData           : BOOL;
by_Node         : BYTE;
t_SwitchDelay   : TIME := T#100ms;
nRockerID       : INT;
nEnOceanID      : BYTE;
str_KL6581      : STR_KL6581;
```

bStart: Positive Flanke sendet die Daten.

bData: Wert der Übertragen werden soll.

by_Node: Adresse der EnOcean-Sender und -Empfänger KL6583-0000 an die das Telegramm gesendet werden soll (gültige Werte: 1...8).

t_SwitchDelay: Wie lange der Taster gedrückt werden muss.

nRockerID: Tasternummer, gültige Werte 0..3.

nEnOceanID: Virtuelle EnOcean-ID, auf die reale EnOcean-ID wird ein Wert von 0...127 auf addiert (gültige Werte: 0...127).

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

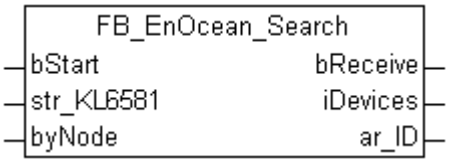
```
bBusy          : BOOL;
bError         : BOOL;
iErrorID       : E_KL6581_Err;
```

bBusy: Baustein ist aktiv, es können noch keine neuen Daten gesendet werden.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorID* beschrieben.

iErrorID: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

6.2.10 FB_EnOcean_Search



Dieser Funktionsbaustein zeigt alle EnOcean IDs an, die er empfangen hat und trägt diese in ein Empfangsarray ein *(ar_ID)*. Es können bis zu 256 EnOcean Teilnehmer erkannt werden. Wahlweise kann der Baustein auch für jede EnOcean-Sender und -Empfänger KL6583-0000 einzeln angelegt werden. Damit kann man erkennen ob ein EnOcean Teilnehmer von mehreren KL6583 empfangen wird.

VAR_INPUT

```
bStart      : BOOL;
str_KL6581  : STR_KL6581;
byNode      : BYTE;
```

bStart: Bei TRUE ist der Baustein aktiv, bei FALSE deaktiviert.

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen KL6583 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

VAR_OUTPUT

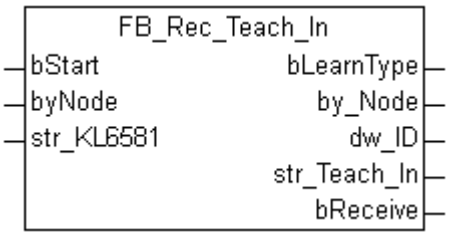
```
bReceive    : BOOL := TRUE;
iDevices    : INT;
ar_ID       : ARRAY [0..255] OF DWORD;
```

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

iDevices: Anzahl an gefundenen EnOcean-Teilnehmern.

ar_ID: EnOcean-IDs die gefunden wurden.

6.2.11 FB_Rec_Teach_In



Dieser Funktionsbaustein zeigt an, wenn bei einem EnOcean-Teilnehmer die Learn-Taste gedrückt wird. Sollte das Flag *bLearnType* gesetzt sein können weitere Informationen des EnOcean-Teilnehmers ausgelesen werden. Dies ist eine Funktion, die das EnOcean-Gerät liefern muss (die aber bisher von den wenigsten EnOcean-Geräten unterstützt wird).

VAR_INPUT

```
bStart        : BOOL;
byNode        : BYTE;
str_KL6581    : STR_KL6581;
```

bStart: Bei TRUE ist der Baustein aktiv, bei FALSE deaktiviert.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

```
bLearnType    : BOOL;
by_Node       : BYTE;
dw_ID         : DWORD;
str_Teach_In  : STR_Teach_In;
bReceive      : BOOL := TRUE;
```

bLearnType: Ist das Bit gesetzt, finden Sie weitere Daten in der Struktur *str_Teach_In*.

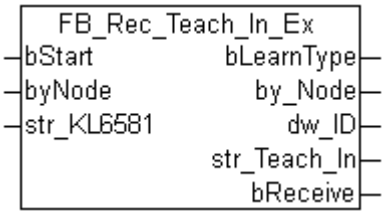
by_Node: Anzahl an gefundenen EnOcean Teilnehmern.

dw_ID: EnOcean IDs bei dem die Learn-Taste gedrückt wurde.

str_Teach_In: Datenstruktur - Hersteller ID, Typ und Profil (siehe STR_Teach_In [▶ 49]).

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

6.2.12 FB_Rec_Teach_In_Ex



Dieser Funktionsbaustein zeigt an, wenn bei einem EnOcean-Teilnehmer die Learn-Taste gedrückt wird. Sollte das Flag *bLearnType* gesetzt sein können weitere Informationen des EnOcean-Teilnehmers ausgelesen werden. Dies ist eine Funktion, die das EnOcean-Gerät liefern muss (die aber bisher von den wenigsten EnOcean-Geräten unterstützt wird). Zusätzlich zum FB_Rec_Teach_In() Funktionsblock wird noch geprüft, ob es sich um ein EEP Telegramm handelt.

VAR_INPUT

```
bStart        : BOOL;
byNode        : BYTE;
str_KL6581    : STR_KL6581;
```

bStart: Bei TRUE ist der Baustein aktiv, bei FALSE deaktiviert.

byNode: Filter - bei den Wert Null werden die EnOcean-Telegramme von allen EnOcean-Sender und -Empfänger KL6583-0000 empfangen. Wird ein Wert von 1...8 eingetragen, werden nur die Daten von der entsprechenden KL6583 empfangen.

str_KL6581: Datenstruktur die mit dem Baustein FB_KL6581() [▶ 24] verbunden wird (siehe STR_KL6581 [▶ 48]).

VAR_OUTPUT

```
bLearnType    : BOOL;
by_Node       : BYTE;
dw_ID         : DWORD;
str_Teach_In  : STR_Teach;
bReceive      : BOOL := TRUE;
```

bLearnType: Ist das Bit gesetzt, finden Sie weitere Daten in der Struktur *str_Teach_In*.

by_Node: Anzahl an gefundenen EnOcean Teilnehmern.

dw_ID: EnOcean IDs bei dem die Learn-Taste gedrückt wurde.

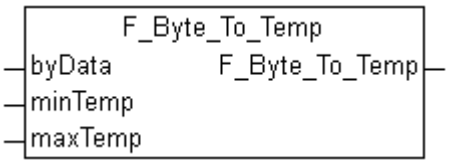
str_Teach_In: Datenstruktur - Hersteller ID, Typ und Funktion (siehe STR_Teach [▶ 48]).

bReceive: Bei empfangenem EnOcean Telegramm wird dieser Wert für einen Zyklus auf FALSE gesetzt.

Voraussetzungen

Entwicklungsumgebung	Zielsystem	erforderliche Bibliotheken
TwinCAT 2.11 R3/x64 ab Build 2251	PC/CX, BX oder BC	TcEnOcean-Bibliothek ab V2.0.6

6.2.13 F_Byte_to_Temp : REAL



Diese Funktion wandelt einen Byte-Rohwert in eine REAL-Variable um.

Bei EnOcean werden Temperaturdaten in einem bestimmten Format übertragen, das ein Byte groß ist. Diese Daten sind meist auf einen bestimmten Temperaturwert Skaliert.

Zum Beispiel wird ein Wert aus einem Wertebereich von 0...40 °C übertragen.

Man gibt bei der Funktion jetzt den minimalen und maximalen Wert der Daten an und übergibt der Funktion den Rohwert. Der Ausgang der Funktion gibt dann die Temperatur als REAL-Variable aus.

VAR_INPUT

```
byData       : BYTE;
minTemp      : REAL := 0;
maxTemp      : REAL := 40;
```

byData: Rohdaten.

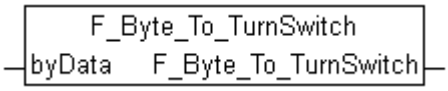
minTemp: Minimale Temperatur.

maxTemp: Maximale Temperatur.

F_Byte_To_Temp: Temperaturwert, skaliert über *minTemp* und *maxTemp*.

6.2.14 F_Byte_to_TurnSwitch : STREnOceanTurnSwitch

STREnOceanTurnSwitch [▶ 49]



Diese Funktion wandelt einen Byte Rohwert in ein Bool Array um, das als Datenstruktur vorliegt.

VAR_INPUT

```
byData      : BYTE;
```

byData: Rohdaten.

F_Byte_To_TurnSwitch: Datenstruktur (AUTO, 0, I, II, III).

Die Werte werden wie folgt interpretiert:

210. .255: F_Byte_to_TurnSwitch.bStageAuto := TRUE;

190 ..209: F_Byte_to_TurnSwitch.bStage_0 := TRUE;

165. .189: F_Byte_to_TurnSwitch.bStage_1 := TRUE;

145. .164: F_Byte_to_TurnSwitch.bStage_2 := TRUE;

0.. 144: F_Byte_to_TurnSwitch.bStage_3 := TRUE;

6.2.15 Fehlercodes der KL6581

Wert (hex)	Wert (dez)	Wert (enum)	Beschreibung
0x0000	0	NO_ERROR	Am Baustein liegt kein Fehler an.
0x000A	10	KL6581_WrongTerminal	Falsche Klemme angeschlossen.
0x0010	16	KL6581_WatchdogError	Zeitüberschreitung beim Initialisierungsvorgang des Bausteins FB_KL6581() [▶ 24].
0x0011	17	KL6581_NoComWithKL6581	Üblicherweise gibt es bei dieser Meldung keine Verbindung zur Klemme. Klemme im System Manager mit den Variablen verknüpft? Klemme falsch gesteckt? Alles bereinigen, alles Übersetzen und im System Manager neu eingelesen?
0x0012	18	KL6581_idx_number_not_OK	Die Eingangsvariable *nIdx* des Bausteins FB_KL6581() ist größer als 64.
0x0013	19	KL6581_Switch_to_Stopp	Die Klemme ist aus dem Datenaustausch mit der EnOcean-Sender und -Empfänger KL6583-0000
0x0014	20	KL6581_not_ready	Interne Meldung für die Funktionsblöcke, die an den FB_KL6581() angeschlossen sind.
0x0015	21	KL6581_No_KL6853_Found	Es ist keine KL6583 an der EnOcean-Masterklemme KL6581
0x0016	22	KL6581_TransmissionError	Daten konnten nicht gesendet werden, Adresse der KL6583 prüfen oder KL6583 nicht betriebsbereit.

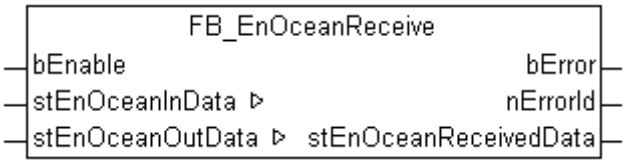
6.3 KL6021-0023

Bausteine	Beschreibung
FB_EnOceanReceive [▶ 35]	Kommunikation mit einer EnOcean-Busklemme KL6021-0023

Read

Bausteine	Beschreibung
FB_EnOceanPTM100 [▶ 36]	Empfängt die Signale eines PTM100-Moduls.
FB_EnOceanPTM200 [▶ 38]	Empfängt die Signale eines PTM200-Moduls.
FB_EnOceanSTM100 [▶ 40]	Empfängt die Signale eines STM100-Moduls (veraltet).
FB_EnOceanSTM100Generic [▶ 42]	Empfängt die Signale eines STM100-Moduls.
FB_EnOceanSTM250 [▶ 44]	Empfängt die Signale eines STM250-Moduls.

6.3.1 FB_EnOceanReceive



Der Funktionsbaustein FB_EnOceanReceive() ist ein Empfangsbaustein, der die von den EnOcean-Modulen gesendeten Telegramme in der Struktur *stEnOceanReceivedData* zur Verfügung stellt. Diese Struktur kann dann mit den Bausteinen FB_EnOceanPTM100() [▶ 36] und FB_EnOceanSTM100() [▶ 40]ausgewertet werden. In der Dokumentation dieser Bausteine sind auch Programmbeispiele aufgeführt, die die Funktionsweise näher erläutern.

VAR_INPUT

```
bEnable            : BOOL := FALSE;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

VAR_OUTPUT

```
bError                : BOOL := FALSE;
nErrorId              : UDINT := 0;
stEnOceanReceivedData : ST_EnOceanReceivedData;
```

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

stEnOceanReceivedData: In dieser Struktur werden die empfangenen Daten abgelegt (siehe ST_EnOceanReceivedData [▶ 51]).

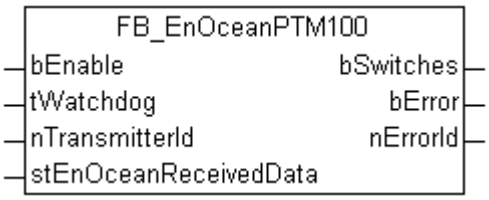
VAR_IN_OUT

```
stEnOceanInData       : ST_EnOceanInData;
stEnOceanOutData      : ST_EnOceanOutData;
```

stEnOceanInData: Wird mit den Eingangsadressen der EnOcean-Busklemme KL6021-0023 im System Manager verknüpft (siehe ST_EnOceanInData [▶ 50]).

stEnOceanOutData: Wird mit den Ausgangsadressen der EnOcean-Busklemme KL6021-0023 im System Manager verknüpft (siehe ST_EnOceanOutData [▶ 50]).

6.3.2 FB_EnOceanPTM100



Der Funktionsbaustein FB_EnOceanPTM100() gibt eine anwenderfreundliche Auswertung über den Zustand eines EnOcean PTM100-Moduls. Hierzu ist die Verwendung des Funktionsblocks FB_EnOceanReceive() [▶ 35] notwendig.

Im Unterschied zum PTM200- und PTM250-Modul, kann beim PTM100-Modul nur ein Taster gleichzeitig gedrückt werden. Des Weiteren unterstützt das PTM100-Modul acht, statt vier Taster.

Wichtig ist, dass zu jedem verwendeten Tasten-Modul eine neue Instanz dieses Bausteines angelegt werden muss.

VAR_INPUT

```
bEnable                 : BOOL := FALSE;
tWatchdog               : TIME;
nTransmitterId          : UDINT;
stEnOceanReceivedData   : ST_EnOceanReceivedData;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

tWatchdog: Überwachungszeit. Innerhalb dieser Zeit müssen neue Informationen über den Eingang *stEnOceanReceivedData* in diesen Baustein gelangen. Ist diese Zeit auf t#0s gesetzt, so ist die Watchdog-Funktion inaktiv.

nTransmitterId: ID des EnOcean-Modules, auf den der Baustein reagieren soll.

stEnOceanReceivedData: Informationen und notwendige Verbindung zum EnOcean-Empfangsbaustein FB_EnOceanReceive() [▶ 35]. Diese Informationen sind in einer Struktur des Typs ST_EnOceanReceivedData [▶ 51] hinterlegt.

VAR_OUTPUT

```
bSwitches           : ARRAY [0..7] OF BOOL;
bError              : BOOL := FALSE;
nErrorId            : UDINT := 0;
```

bSwitches: Dieses Feld von 8 boolschen Werten beschreibt die Zustände der 8 Taster auf dem Taster-Modul.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

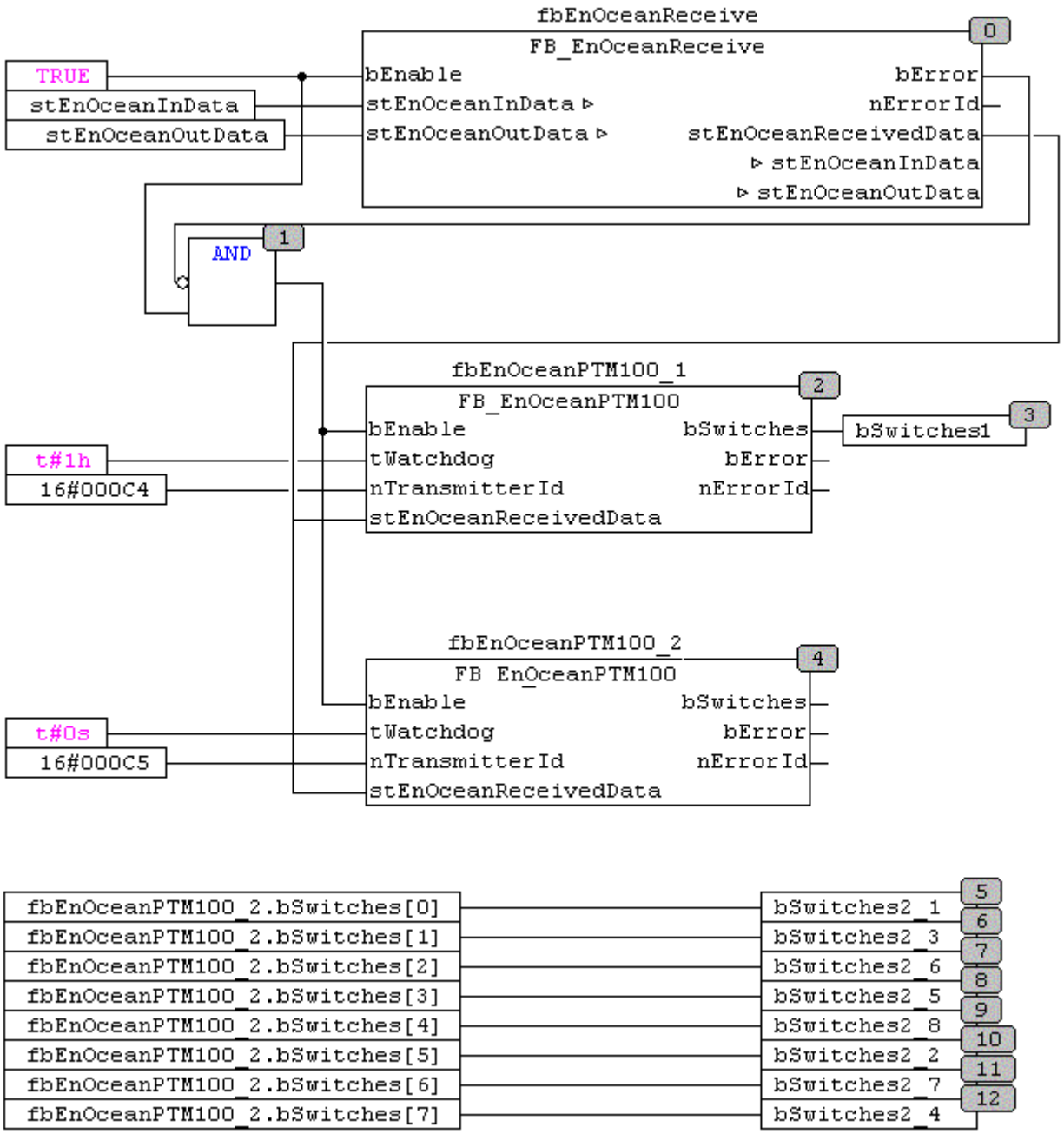
Anhand des folgenden Programmbeispiels soll die Funktionsweise des Bausteines näher beschrieben werden:

```
PROGRAM MAIN
VAR
  fbEnOceanReceive      : FB_EnOceanReceive;
  fbEnOceanPTM100_1     : FB_EnOceanPTM100;
  fbEnOceanPTM100_2     : FB_EnOceanPTM100;
  bSwitches1            : ARRAY [0..7] OF BOOL;
  bSwitches2_1          : BOOL;
  bSwitches2_2          : BOOL;
  bSwitches2_3          : BOOL;
  bSwitches2_4          : BOOL;
  bSwitches2_5          : BOOL;
  bSwitches2_6          : BOOL;
```

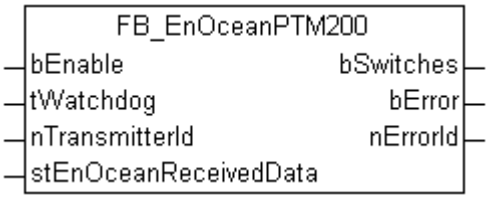
```
  bSwitches2_7         : BOOL;
  bSwitches2_8         : BOOL;
END_VAR

fbEnOceanReceive(bEnable := TRUE,
         stEnOceanInData := stEnOceanInData,
         stEnOceanOutData := stEnOceanOutData);

fbEnOceanPTM100_1(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C4,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);
          bSwitches1 := fbEnOceanPTM100_1.bSwitches;

fbEnOceanPTM100_2(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C5,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);

bSwitches2_1 := fbEnOceanPTM100_2.bSwitches[0];
bSwitches2_3 := fbEnOceanPTM100_2.bSwitches[1];
bSwitches2_6 := fbEnOceanPTM100_2.bSwitches[2];
bSwitches2_5 := fbEnOceanPTM100_2.bSwitches[3];
bSwitches2_8 := fbEnOceanPTM100_2.bSwitches[4];
bSwitches2_2 := fbEnOceanPTM100_2.bSwitches[5];
bSwitches2_7 := fbEnOceanPTM100_2.bSwitches[6];
bSwitches2_4 := fbEnOceanPTM100_2.bSwitches[7];
```

In diesem Beispielprogramm werden 2 Sendemodule (PTM100) abgefragt, ein Sendemodul mit der Transmitter-Id 16#C4 und ein anderes mit der Transmitter-Id 16#C5. Für beide Sendemodule wurde jeweils ein Funktionsbaustein FB_EnOceanPTM100() angelegt. Beide Funktionsbausteine erhalten ihre Informationen von einem vorangeschalteten Empfängerbaustein FB_EnOceanReceive() und sind nur dann aktiv (Eingang *bEnable*), wenn der Empfängerbaustein aktiv und nicht in Störung ist. Die Taster des ersten Sendemoduls werden zur weiteren Auswertung einem gleich großen bool´schen Array *bSwitches1* zugeordnet, während die Taster des zweiten Sendemodules einzelnen boolschen Variablen *bSwitches2_1* bis *bSwitches2_8* zugewiesen werden - beide Möglichkeiten sind denkbar.

Im freigrafischen Funktionsplanedito (CFC) würde dasselbe Beispiel folgendermaßen aussehen, wobei die Variablendeklaration die gleiche wie beim oben aufgeführten Beispiel ist:



6.3.3 FB_EnOceanPTM200



Der Funktionsbaustein FB_EnOceanPTM200() gibt eine anwenderfreundliche Auswertung über den Zustand eines EnOcean PTM200- oder PTM250-Moduls. Hierzu ist die Verwendung des Funktionsblocks FB_EnOceanReceive() [▶ 35] notwendig.

Im Unterschied zum PTM100-Modul, können beim PTM200-Module und beim PTM250-Modul bis zu zwei Taster gleichzeitig gedrückt werden. Des weiteren unterstützt das PTM200- und PTM250-Modul vier, statt acht Taster.

Wichtig ist, dass zu jedem verwendeten Tasten-Modul eine neue Instanz dieses Bausteines angelegt werden muss.

VAR_INPUT

```
bEnable                : BOOL := FALSE;
tWatchdog              : TIME;
nTransmitterId         : UDINT;
stEnOceanReceivedData  : ST_EnOceanReceivedData;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

tWatchdog: Überwachungszeit. Innerhalb dieser Zeit müssen neue Informationen über den Eingang *stEnOceanReceivedData* in diesen Baustein gelangen. Ist diese Zeit auf t#0s gesetzt, so ist die Watchdog-Funktion inaktiv.

nTransmitterId: ID des EnOcean-Modules, auf den der Baustein reagieren soll.

stEnOceanReceivedData: Informationen und notwendige Verbindung zum EnOcean-Empfangsbaustein FB_EnOceanReceive() [▶ 35]. Diese Informationen sind in einer Struktur des Typs ST_EnOceanReceivedData [▶ 51] hinterlegt.

VAR_OUTPUT

```
bSwitches          : ARRAY [0..3] OF BOOL;
bError             : BOOL := FALSE;
nErrorId           : UDINT := 0;
```

bSwitches: Dieses Feld von 4 boolschen Werten beschreibt die Zustände der 4 Taster auf dem Taster-Modul.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

Anhand des folgenden Programmbeispiels soll die Funktionsweise des Bausteines näher beschrieben werden:

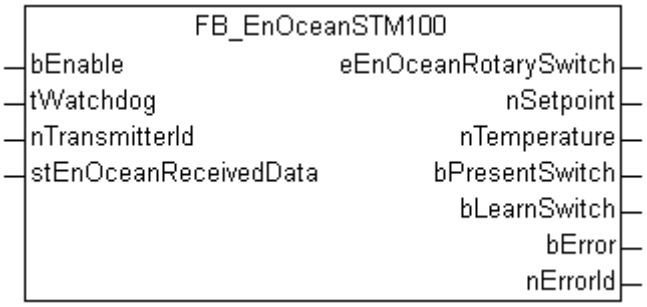
```
PROGRAM MAIN
VAR
  fbEnOceanReceive     : FB_EnOceanReceive;
  fbEnOceanPTM100_1    : FB_EnOceanPTM200;
  fbEnOceanPTM100_2    : FB_EnOceanPTM200;
  bSwitches1           : ARRAY [0..3] OF BOOL;
  bSwitches2_1         : BOOL;
  bSwitches2_2         : BOOL;
  bSwitches2_3         : BOOL;
  bSwitches2_4         : BOOL;
END_VAR

fbEnOceanReceive(bEnable := TRUE,
         stEnOceanInData := stEnOceanInData
         stEnOceanOutData := stEnOceanOutData);

fbEnOceanPTM200_1(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C6,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);
          bSwitches1 := fbEnOceanPTM200_1.bSwitches;

fbEnOceanPTM200_2(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C7,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);

bSwitches2_1 := fbEnOceanPTM200_2.bSwitches[0];
```

```
bSwitches2_2 := fbEnOceanPTM200_2.bSwitches[1];
bSwitches2_3 := fbEnOceanPTM200_2.bSwitches[2];
bSwitches2_4 := fbEnOceanPTM200_2.bSwitches[3];
```

In diesem Beispielprogramm werden 2 Sendemodule (PTM200 / PTM250) abgefragt, ein Sendemodul mit der Transmitter-Id 16#C6 und ein anderes mit der Transmitter-Id 16#C7. Für beide Sendemodule wurde jeweils ein Funktionsbaustein FB_EnOceanPTM200() angelegt. Beide Funktionsbausteine erhalten ihre Informationen von einem vorangeschalteten Empfängerbaustein FB_EnOceanReceive() [▶ 35] und sind nur dann aktiv (Eingang *bEnable*), wenn der Empfängerbaustein aktiv und nicht in Störung ist. Die Taster des ersten Sendemoduls werden zur weiteren Auswertung einem gleich großen boolschen Array *bSwitches1* zugeordnet, während die Taster des zweiten Sendemoduls einzelnen boolschen Variablen *bSwitches2_1* bis *bSwitches2_4* zugewiesen werden - beide Möglichkeiten sind denkbar.

Ein Bespielprogramm im freigrafischen Funktionsplaneditor (CFC) finden sie bei der Beschreibung zum FB_EnOceanPTM100() [▶ 36].

6.3.4 FB_EnOceanSTM100



Veraltet! Bei neuen Projekten sollte der Baustein FB_EnOceanSTM100Generic() [▶ 42] verwendet werden!

Der Funktionsbaustein FB_EnOceanSTM100() gibt eine anwenderfreundliche Auswertung über die Daten eines EnOcean STM100-Moduls. Hierzu ist die Verwendung des Funktionsblocks FB_EnOceanReceive() [▶ 35] notwendig.

Wichtig ist, dass zu jedem verwendeten STM100-Modul eine neue Instanz dieses Bausteines angelegt werden muss.

VAR_INPUT

```
bEnable                : BOOL := FALSE;
tWatchdog              : TIME;
nTransmitterId         : UDINT;
stEnOceanReceivedData  : ST_EnOceanReceivedData;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

tWatchdog: Überwachungszeit. Innerhalb dieser Zeit müssen neue Informationen über den Eingang *stEnOceanReceivedData* in diesen Baustein gelangen. Ist diese Zeit auf t#0s gesetzt, so ist die Watchdog-Funktion inaktiv.

nTransmitterId: ID des EnOcean-Modules, auf den der Baustein reagieren soll.

stEnOceanReceivedData: Informationen und notwendige Verbindung zum EnOcean-Empfangsbaustein FB_EnOceanReceive() [▶ 35]. Diese Informationen sind in einer Struktur des Typs ST_EnOceanReceivedData [▶ 51] hinterlegt.

VAR_OUTPUT

```
eEnOceanRotarySwitch  : E_EnOceanRotarySwitch;
nSetpoint             : INT;
nTemperature          : INT;
bPresentSwitch        : BOOL;
```

```
bLearnSwitch          : BOOL;
bError                : BOOL := FALSE;
nErrorId              : UDINT := 0;
```

eEnOceanRotarySwitch: Die Ausgabe an diesem Ausgang erfolgt nach der Definition des ENUM-Types E_EnOceanRotarySwitch [▶ 49] und beschreibt die Stellung des Drehschalters am Raumbediengerät.

nSetpoint: An dieser Ausgangsvariablen liegt der am Gerät eingestellte Sollwert an. Dieser kann Werte im Bereich von -100 bis +100 annehmen.

nTemperature: Hier wird die gemessene Temperatur in 1/10 °C ausgegeben mit einem Messbereich von 0 °C bis 40 °C. Bei ausgelöstem Watchdog vermutet der Baustein einen drahtbruchähnlichen Fehler und der Wert wird fest auf 850 °C gesetzt.

bPresentSwitch: Bei Aktivierung der Anwesenheitstaste am Raumbediengerät wird dieser Ausgang TRUE.

bLearnSwitch: Bei Aktivierung der Anlerntaste am Raumbediengerät wird dieser Ausgang TRUE.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

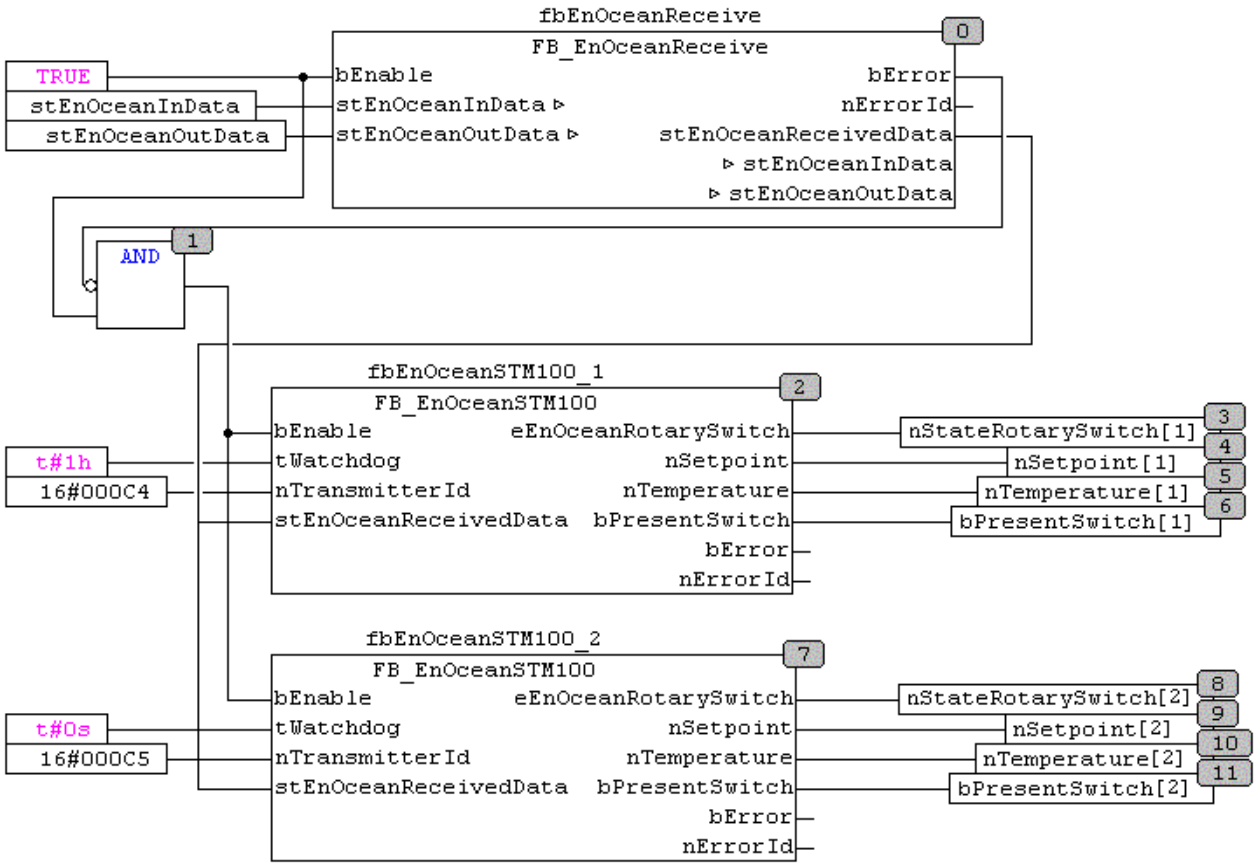
nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

Anhand des folgenden Programmbeispiels soll die Funktionsweise des Bausteines näher beschrieben werden:

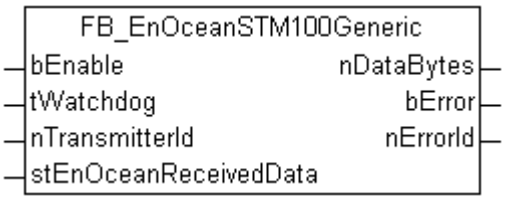
```
PROGRAM MAIN
VAR
  fbEnOceanReceive   : FB_EnOceanReceive;
  fbEnOceanSTM100_1  : FB_EnOceanSTM100;
  fbEnOceanSTM100_2  : FB_EnOceanSTM100;
  nTemperature       : ARRAY [1..2] OF INT;
  nSetpoint          : ARRAY [1..2] OF INT;
  nStateRotarySwitch : ARRAY [1..2] OF E_EnOceanRotarySwitch;
  bPresentSwitch     : ARRAY [1..2] OF BOOL;
END_VAR

fbEnOceanReceive(bEnable := TRUE,
         stEnOceanInData := stEnOceanInData,
         tEnOceanOutData := stEnOceanOutData);

fbEnOceanSTM100_1(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C4,
          tWatchdog := t#1h,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData
          nTemperature => Temperature[1],
          nSetpoint => nSetpoint[1] ,
          eEnOceanRotarySwitch => nStateRotarySwitch[1],
          bPresentSwitch => bPresentSwitch[1]);

fbEnOceanSTM100_2(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C5,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData,
          nTemperature => Temperature[2],
          nSetpoint => nSetpoint[2] ,
          eEnOceanRotarySwitch => nStateRotarySwitch[2],
          bPresentSwitch => bPresentSwitch[2]);
```

In diesem Beispielprogramm werden 2 Raumbediengeräte abgefragt, einer mit der Transmitter-Id 16#000000C4 und ein anderer mit der Transmitter-Id 16#000000C5. Für beide Module wurde jeweils ein Funktionsbaustein FB_EnOceanSTM100() angelegt. Beide Funktionsbausteine erhalten ihre Informationen von einem vorangeschalteten Empfängerbaustein FB_EnOceanReceive() und sind nur dann aktiv (Eingang *bEnable*), wenn der Empfängerbaustein aktiv und nicht in Störung ist. Das erste Gerät wird durch die Watchdog-Funktion überwacht, wobei innerhalb von 1 Stunde neue Werte an die Steuerung übertragen werden müssen, das zweite Gerät ist ohne Watchdog-Überwachung programmiert. Zur weiteren Auswertung sind die an den Funktionsbausteinen ausgegebenen Werten Merkern zugewiesen.

Im freigrafischen Funktionsplaneditor (CFC) würde dasselbe Beispiel folgendermaßen aussehen, wobei die Variablendeklaration die gleiche wie beim oben aufgeführten Beispiel ist:



6.3.5 FB_EnOceanSTM100Generic



Der Funktionsbaustein FB_EnOceanSTM100Generic() gibt eine anwenderfreundliche Auswertung über die Daten eines EnOcean STM100-Moduls. Hierzu ist die Verwendung des Funktionsblocks FB_EnOceanReceive() [▶ 35] notwendig.

Wichtig ist, dass zu jedem verwendeten STM100-Modul eine neue Instanz dieses Bausteines angelegt werden muss.

VAR_INPUT

```
bEnable                : BOOL := FALSE;
tWatchdog              : TIME;
nTransmitterId         : UDINT;
stEnOceanReceivedData  : ST_EnOceanReceivedData;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

tWatchdog: Überwachungszeit. Innerhalb dieser Zeit müssen neue Informationen über den Eingang *stEnOceanReceivedData* in diesen Baustein gelangen. Ist diese Zeit auf t#0s gesetzt, so ist die Watchdog-Funktion inaktiv.

nTransmitterId: ID des EnOcean-Modules, auf den der Baustein reagieren soll.

stEnOceanReceivedData: Informationen und notwendige Verbindung zum EnOcean-Empfangsbaustein FB_EnOceanReceive() [▶ 35]. Diese Informationen sind in einer Struktur des Typs ST_EnOceanReceivedData [▶ 51] hinterlegt.

VAR_OUTPUT

```
nDataBytes          : ARRAY [0..3] OF BYTE;
bError              : BOOL := FALSE;
nErrorId            : UDINT := 0;
```

nDataBytes: 4 Bytes großes Array mit dem Nutzdaten die das STM100-Modul versendet hat. Die Bedeutung der einzelnen Bytes ist herstellerabhängig.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

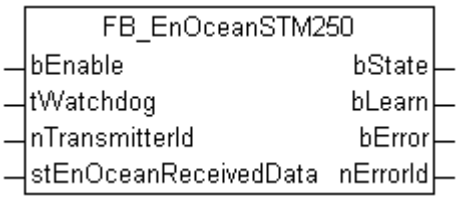
nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

Anhand des folgenden Programmbeispiels soll die Funktionsweise des Bausteines näher beschrieben werden:

```
PROGRAM MAIN
VAR
  fbEnOceanReceive    : FB_EnOceanReceive;
  fbEnOceanSTM100_1   : FB_EnOceanSTM100Generic;
  fbEnOceanSTM100_2   : FB_EnOceanSTM100Generic;
  nTemperature        : ARRAY [1..2] OF BYTE;
  nSetpoint           : ARRAY [1..2] OF BYTE;
  nStateRotarySwitch  : ARRAY [1..2] OF BYTE;
  nPresentSwitch      : ARRAY [1..2] OF BYTE;
END_VAR

fbEnOceanReceive(bEnable := TRUE,
         stEnOceanInData := stEnOceanInData,
         stEnOceanOutData := stEnOceanOutData);

fbEnOceanSTM100_1(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C4,
          tWatchdog := t#1h,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);
nTemperature[1] := fbEnOceanSTM100_1.nDataBytes[0];
nSetpoint[1] := fbEnOceanSTM100_1.nDataBytes[1];
nStateRotarySwitch[1 := fbEnOceanSTM100_1.nDataBytes[2];
nPresentSwitch[1] := fbEnOceanSTM100_1.nDataBytes[3];

fbEnOceanSTM100_2(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
          nTransmitterId := 16#000000C5,
          tWatchdog := t#0s,
          stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData);
nTemperature[2] := fbEnOceanSTM100_2.nDataBytes[0];
nSetpoint[2] := fbEnOceanSTM100_2.nDataBytes[1];
nStateRotarySwitch[2] := fbEnOceanSTM100_2.nDataBytes[2];
nPresentSwitch[2] := fbEnOceanSTM100_2.nDataBytes[3];
```

In diesem Beispielprogramm werden 2 EnOcean Sendemodule abgefragt, einer mit der Transmitter-Id 16#000000C4 und ein anderer mit der Transmitter-Id 16#000000C5. Für beide Transmitter wurde jeweils ein Funktionsbaustein FB_EnOceanSTM100Generic() angelegt. Beide Funktionsbausteine erhalten ihre Informationen von einem vorangeschalteten Empfängerbaustein FB_EnOceanReceive() [▶ 35] und sind nur dann aktiv (Eingang *bEnable*), wenn der Empfängerbaustein aktiv und nicht in Störung ist. Das erste Gerät wird durch die Watchdog-Funktion überwacht, wobei innerhalb von 1 Stunde neue Werte an die Steuerung übertragen werden müssen, das zweite Gerät ist ohne Watchdog-Überwachung programmiert. Zur weiteren Auswertung sind die an den Funktionsbausteinen ausgegebenen Werten Variablen zugewiesen. Für eine weitere Verwendung der Werte müssten diese noch in physikalische Größen skaliert werden. Wie die Umrechnung zu erfolgen hat, ist aus dem Datenblatt des Sensors zu entnehmen.

6.3.6 FB_EnOceanSTM250



Der Funktionsbaustein FB_EnOceanSTM250() gibt eine anwenderfreundliche Auswertung über die Daten eines EnOcean STM250-Moduls. Hierzu ist die Verwendung des Funktionsblocks FB_EnOceanReceive() [▶ 35] notwendig.

Wichtig ist, dass zu jedem verwendeten STM250-Modul eine neue Instanz dieses Bausteines angelegt werden muss.

VAR_INPUT

```
bEnable                : BOOL := FALSE;
tWatchdog              : TIME;
nTransmitterId         : UDINT;
stEnOceanReceivedData  : ST_EnOceanReceivedData;
```

bEnable: Ein positives Signal an diesem Eingang setzt den Baustein aktiv. Bei einem negativen Signal am *bEnable* Eingang wird im Baustein keine Funktion ausgeführt und alle Ausgänge werden auf 0 bzw. FALSE gesetzt.

tWatchdog: Überwachungszeit. Innerhalb dieser Zeit müssen neue Informationen über den Eingang *stEnOceanReceivedData* in diesen Baustein gelangen. Ist diese Zeit auf t#0s gesetzt, so ist die Watchdog-Funktion inaktiv.

nTransmitterId: ID des EnOcean-Modules, auf den der Baustein reagieren soll.

stEnOceanReceivedData: Informationen und notwendige Verbindung zum EnOcean-Empfangsbaustein FB_EnOceanReceive() [▶ 35]. Diese Informationen sind in einer Struktur des Typs ST_EnOceanReceivedData [▶ 51] hinterlegt.

VAR_OUTPUT

```
bState             : BOOL;
bLearn             : BOOL;
bError             : BOOL := FALSE;
nErrorId           : UDINT := 0;
```

bState: Bei Aktivierung des Reedkontakts am STM250-Modul wird dieser Ausgang TRUE (Kontakt geschlossen).

bLearn: Bei Aktivierung der Lerntaste am STM250-Modul wird dieser Ausgang FALSE.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *nErrorId* beschrieben.

nErrorId: Beschreibt die Art des Fehlers (siehe Fehlercodes [▶ 45]).

Anhand des folgenden Programmbeispiels soll die Funktionsweise des Bausteines näher beschrieben werden:

```
PROGRAM MAIN
VAR
  fbEnOceanReceive  : FB_EnOceanReceive;
  fbEnOceanSTM250   : FB_EnOceanSTM250;
  bState            : BOOL;
  bLearn            : BOOL;
END_VAR

fbEnOceanReceive(bEnable := TRUE,
         stEnOceanInData := stEnOceanInData,
         stEnOceanOutData := stEnOceanOutData);
```

```
fbEnOceanSTM250(bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceive.bEnable,
        nTransmitterId := 16#000008CA,
        tWatchdog := t#0s,
        stEnOceanReceivedData := fbEnOceanReceive.stEnOceanReceivedData,
        bState => bState,
        bLearn => bLearn);
```

In diesem Beispielprogramm wird ein STM250 Modul mit der Transmitter-Id 16#000008CA abgefragt. Hierzu wurde der Funktionsbaustein FB_EnOceanSTM250() angelegt. Dieser Funktionsbaustein erhält Informationen von einem vorangeschalteten Empfängerbaustein FB_EnOceanReceive() [▶ 35] und ist nur dann aktiv (Eingang *bEnable*), wenn der Empfängerbaustein aktiv und nicht in Störung ist. Zur weiteren Auswertung sind die an den Funktionsbaustein ausgegebenen Werten Variablen zugewiesen.

6.3.7 Fehlercodes der KL6021-0023

Wert (hex)	Beschreibung
0x0000	Kein Fehler.
0x0001	Prüfsummenfehler.
0x0002	Watchdogüberwachung.
0x0003	Pufferüberlauf (in der KL6023).
0x0004	Noch keine Daten vom Sensor empfangen.

6.4 Datentypen

6.4.1 E_EnOcean_Org

Typ des EnOcean Telegramms.

```
TYPE E_EnOcean_Org :
(
  PTM_TELEGRAM    := 5,
  STM_1BYTE_TELEGRAM  := 6,
  STM_4BYTE_TELEGRAM  := 7,
  CTM_TELEGRAM    := 8,
  MODEM_TELEGRAM      := 16#A,
  MODEM_ACK_TELEGRAM  := 16#B,
)
END_TYPE
```

PTM_TELEGRAM: PTM Telegramm.

STM_1BYTE_TELEGRAM: 1 Byte Telegramm.

STM_4BYTE_TELEGRAM: 4 Byte Telegramm.

CTM_TELEGRAM: CTM Telegramm.

MODEM_TELEGRAM: Modem Telegramm.

MODEM_ACK_TELEGRAM: Modem Telegramm mit Bestätigung.

6.4.2 E_KL6581_Err

Fehlermeldungen.

```
TYPE E_KL6581_Err :
(
  NO_ERROR          := 16#0,
  KL6581_WrongTerminal      := 16#A,
  KL6581_WatchdogError      := 16#10,
  KL6581_NoComWithKL6581    := 16#11,
  KL6581_idx_number_not_OK  := 16#12,
  KL6581_Switch_to_Stopp    := 16#13,
  KL6581_not_ready      := 16#14,
  KL6581_No_KL6853_Found    := 16#15,
```

```
  KL6581_TransmissionError  := 16#16,
)
END_TYPE
```

NO_ERROR: Am Baustein liegt kein Fehler an.

KL6581_WrongTerminal: Falsche Klemme angeschlossen.

KL6581_WatchdogError: Zeitüberschreitung beim Initialisierungsvorgang des Bausteins FB_KL6581() [▶ 24].

KL6581_NoComWithKL6581: Üblicherweise gibt es bei dieser Meldung keine Verbindung zur Klemme. Klemme im System Manager mit den Variablen verknüpft? Klemme falsch gesteckt? Alles bereinigen, alles Übersetzen und im System Manager neu eingelesen?

KL6581_idx_number_not_OK: Die Eingangsvariable *nIdx* des Bausteins FB_KL6581() ist größer als 64.

KL6581_Switch_to_Stopp: Die Klemme ist aus dem Datenaustausch mit der EnOcean-Sender und -Empfänger KL6583-0000 gegangen, es wurden keine EnOcean Daten gesendet oder empfangen.

KL6581_not_ready: Interne Meldung für die Funktionsblöcke, die an den FB_KL6581() angeschlossen sind.

KL6581_No_KL6853_Found: Es ist keine KL6583 an der EnOcean-Masterklemme KL6581 angeschlossen oder die Kommunikation ist nicht vorhanden!

KL6581_TransmissionError: Daten konnten nicht gesendet werden, Adresse der KL6583 prüfen oder KL6583 nicht betriebsbereit.

6.4.3 KL6581_Input

Prozessabbild der Eingänge der EnOcean-Masterklemme KL6581.

Wird im System Manager mit der Klemme verknüpft.

```
TYPE KL6581_Input :
STRUCT
  nStatus  : BYTE;
  CNODE    : BYTE;
  ORG      : BYTE;
  DB0      : BYTE;
  DB1      : BYTE;
  DB2      : BYTE;
  DB3      : BYTE;
  ID0      : BYTE;
  ID1      : BYTE;
  ID2      : BYTE;
  ID3      : BYTE;
  STATUS   : BYTE;
END_STRUCT
END_TYPE
```

nStatus: Status Byte.

CNODE: Daten Byte.

ORG: Daten Byte.

DB0: Daten Byte.

DB1: Daten Byte.

DB2: Daten Byte.

DB3: Daten Byte.

ID0: Daten Byte.

ID1: Daten Byte.

ID2: Daten Byte.

ID3: Daten Byte.

STATUS: Daten Byte.

6.4.4 KL6581_Output

Prozessabbild der Ausgänge der EnOcean-Masterklemme KL6581.

Wird im System Manager mit der Klemme verknüpft.

```
TYPE KL6581_Output :
STRUCT
  nControl  : BYTE;
  CNODE     : BYTE;
  ORG       : BYTE;
  DB0       : BYTE;
  DB1       : BYTE;
  DB2       : BYTE;
  DB3       : BYTE;
  ID0       : BYTE;
  ID1       : BYTE;
  ID2       : BYTE;
  ID3       : BYTE;
  STATUS    : BYTE;
END_STRUCT
END_TYPE
```

nControl: Control Byte.

CNODE: Daten Byte.

ORG: Daten Byte.

DB0: Daten Byte.

DB1: Daten Byte.

DB2: Daten Byte.

DB3: Daten Byte.

ID0: Daten Byte.

ID1: Daten Byte.

ID2: Daten Byte.

ID3: Daten Byte.

STATUS: Daten Byte.

6.4.5 STR_EnOceanSwitch

Zustand der Taster.

```
TYPE STR_EnOceanSwitch :
STRUCT
  bT1_ON   : BOOL;
  bT1_OFF  : BOOL;
  bT2_ON   : BOOL;
  bT2_OFF  : BOOL;
  bT3_ON   : BOOL;
  bT3_OFF  : BOOL;
  bT4_ON   : BOOL;
  bT4_OFF  : BOOL;
END_STRUCT
END_TYPE
```

bT1_ON: Taster 1 an.

bT1_OFF: Taster 1 aus.

bT2_ON: Taster 2 an.

bT2_OFF: Taster 2 aus.

bT3_ON: Taster 3 an.

bT3_OFF: Taster 3 aus.

bT4_ON: Taster 4 an.

bT4_OFF: Taster 4 aus.

6.4.6 STR_KL6581

Interne Struktur

Über diese Struktur wird der Baustein FB_KL6581() [▶ 24] mit den Sende-Empfangsbausteinen verbunden.

```
TYPE STR_KL6581 :
STRUCT
  by_Status  : BYTE;
  by_Node    : BYTE;
  by_ORG     : BYTE;
  ar_DB      : ARRAY[0..3] OF BYTE;
  _Dummy     : BYTE;
  dw_ID      : DWORD;
  ptData     : DWORD;
  iErrorId   : E_KL6581_Err;
  by_STATE   : BYTE;
  bError     : BOOL;
  idx    : USINT;
END_STRUCT
END_TYPE
```

by_Status: Status.

by_Node: Node Nummer der EnOcean-Sender und -Empfänger KL6583-0000 die das EnOcean Telegramm empfangen hat.

by_ORG: Typ des EnOcean Telegramms.

ar_DB: Daten Bytes.

_Dummy: Platzhalter, ohne weitere Bedeutung.

dw_ID: Transmitter Id.

ptData: Pointer.

iErrorId: Beschreibt die Art des Fehlers (siehe E_KL6581_Err [▶ 45]).

by_STATE: State.

bError: Der Ausgang wird TRUE sobald ein Fehler auftritt. Dieser Fehler wird über die Variable *iErrorId* beschrieben.

idx: Index.

6.4.7 STR_Teach

Datenstruktur - Hersteller ID, Typ und Funktion.

```
TYPE STR_Teach :
STRUCT
  nManufacturerID  : WORD;
  nTYPE        : BYTE;
  nFunc        : BYTE;
END_STRUCT
END_TYPE
```

nManufacturerID: Hersteller Id.

nTYPE: Typ.

nFunc: Funktion.

6.4.8 STR_Teach_In

Datenstruktur - Hersteller ID, Typ und Profil.

```
TYPE STR_Teach_In :
STRUCT
  nManufacturerID  : WORD;
  nTYPE        : BYTE;
  nProfile     : BYTE;
END_STRUCT
END_TYPE
```

nManufacturerID: Hersteller Id.

nTYPE: Typ.

nProfile: Profil.

6.4.9 STREnOceanTurnSwitch

strEnOceanTurnSwitch beschreibt die Stellung des Drehschalters am Raumbediengerät.

```
TYPE STREnOceanTurnSwitch :
STRUCT
  bStageAuto  : BOOL;
  bStage_0    : BOOL;
  bStage_1    : BOOL;
  bStage_2    : BOOL;
  bStage_3    : BOOL;
END_STRUCT
END_TYPE
```

bStageAuto: Schalter in Stellung „Auto“.

bStage_0: Schalter in Stellung „0“.

bStage_1: Schalter in Stellung „1“.

bStage_2: Schalter in Stellung „2“.

bStage_3: Schalter in Stellung „3“.

6.4.10 AR_EnOceanWindow

Diese Struktur zeigt den Zustand des Fensters an.

```
TYPE AR_EnOceanWindow :
STRUCT
  bUp     : BOOL;
  bOpen   : BOOL;
  bClose  : BOOL;
END_STRUCT
END_TYPE
```

bUp: Das Fenster ist gekippt.

bOpen: Das Fenster ist offen.

bClose: Das Fenster ist geschlossen.

6.4.11 E_EnOceanRotarySwitch

E_EnOceanRotarySwitch beschreibt die Stellung des Drehschalters am Raumbediengerät.

```
TYPE E_EnOceanRotarySwitch :
(
  eEnOceanRotarySwitchStep0  := 0,
  eEnOceanRotarySwitchStep1  := 1,
  eEnOceanRotarySwitchStep2  := 2,
  eEnOceanRotarySwitchStep3  := 3,
```

```
  eEnOceanRotarySwitchAuto   := 4,
)
END_TYPE
```

eEnOceanRotarySwitchStep0: Schalter in Stellung „0“.

eEnOceanRotarySwitchStep1: Schalter in Stellung „1“.

eEnOceanRotarySwitchStep2: Schalter in Stellung „2“.

eEnOceanRotarySwitchStep3: Schalter in Stellung „3“.

eEnOceanRotarySwitchAuto: Schalter in Stellung „Auto“.

6.4.12 E_EnOceanSensorType

Sensorentyp.

```
TYPE E_EnOceanSensorType :
(
  eEnOceanSensorTypePTM       := 5,
  eEnOceanSensorTypeSTM1Byte  := 6,
  eEnOceanSensorTypeSTM4Byte  := 7,
  eEnOceanSensorTypeCTM       := 8,
)
END_TYPE
```

eEnOceanSensorTypePTM: PTM.

eEnOceanSensorTypeSTM1Byte: STM 1 Byte.

eEnOceanSensorTypeSTM4Byte: STM 4 Byte.

eEnOceanSensorTypeCTM: CTM.

6.4.13 ST_EnOceanInData

Prozessabbild der Eingänge der EnOcean-Busklemme KL6021-0023.

Wird im TwinCAT System Manager mit der Klemmen verknüpft.

```
TYPE ST_EnOceanInData :
STRUCT
  nStatus  : BYTE;
  nData    : ARRAY[0..10] OF BYTE;
END_STRUCT
END_TYPE
```

nStatus: Status Byte.

nData: 11 Bytes für die Eingangsdaten.

6.4.14 ST_EnOceanOutData

Prozessabbild der Ausgänge der EnOcean-Busklemme KL6021-0023.

Wird im System Manager mit der Klemmen verknüpft.

```
TYPE ST_EnOceanOutData :
STRUCT
  nCtrl  : BYTE;
  nData  : ARRAY[0..10] OF BYTE;
END_STRUCT
END_TYPE
```

nCtrl: Control Byte.

nData: 11 Bytes für die Ausgangsdaten.

6.4.15 ST_EnOceanReceivedData

Interne Struktur

Über diese Struktur wird der Baustein FB_EnOceanReceive() [▶ 35] mit den Empfangsbausteinen verbunden.

```
TYPE ST_EnOceanReceivedData :
STRUCT
  bReceived       : BOOL;
  nLength         : BYTE;
  eEnOceanSensorType  : E_EnOceanSensorType;
  nData           : ARRAY[0..3] OF BYTE;
  nStatus         : BYTE;
  nTransmitterId      : UDINT;
END_STRUCT
END_TYPE
```

bReceived: Daten empfangen.

nLength: Länge.

eEnOceanSensorType: Sensortyp (siehe E_EnOceanSensorType [▶ 50]).

nData: Daten Bytes.

nStatus: Status.

nTransmitterId: Transmitter Id.

7 Anhang

7.1 Beispiele

Voraussetzungen

Beispiel	Beschreibung
https://infosys.beckhoff.com/content/1031/tcplclibenocean/Resources/11985704843.zip	TwinCAT-PLC-Projekt für die KL6581.
Micropelt MVA002	Kommunikation mit dem Heizkörperstellantrieb MVA-002 der Firma micropelt.
HORA SmartDrive MX	Kommunikation mit dem Heizkörperstellantrieb SmartDrive MX der Firma HORA.

7.2 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unseren Internetseiten: https://www.beckhoff.de

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline:	+49(0)5246 963 157
Fax:	+49(0)5246 963 9157
E-Mail:	support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline:	+49(0)5246 963 460
Fax:	+49(0)5246 963 479
E-Mail:	service@beckhoff.com

Beckhoff Firmenzentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49(0)5246 963 0
Fax:	+49(0)5246 963 198
E-Mail:	info@beckhoff.com
Internet:	https://www.beckhoff.de

Mehr Informationen:
www.beckhoff.de/tx1200

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