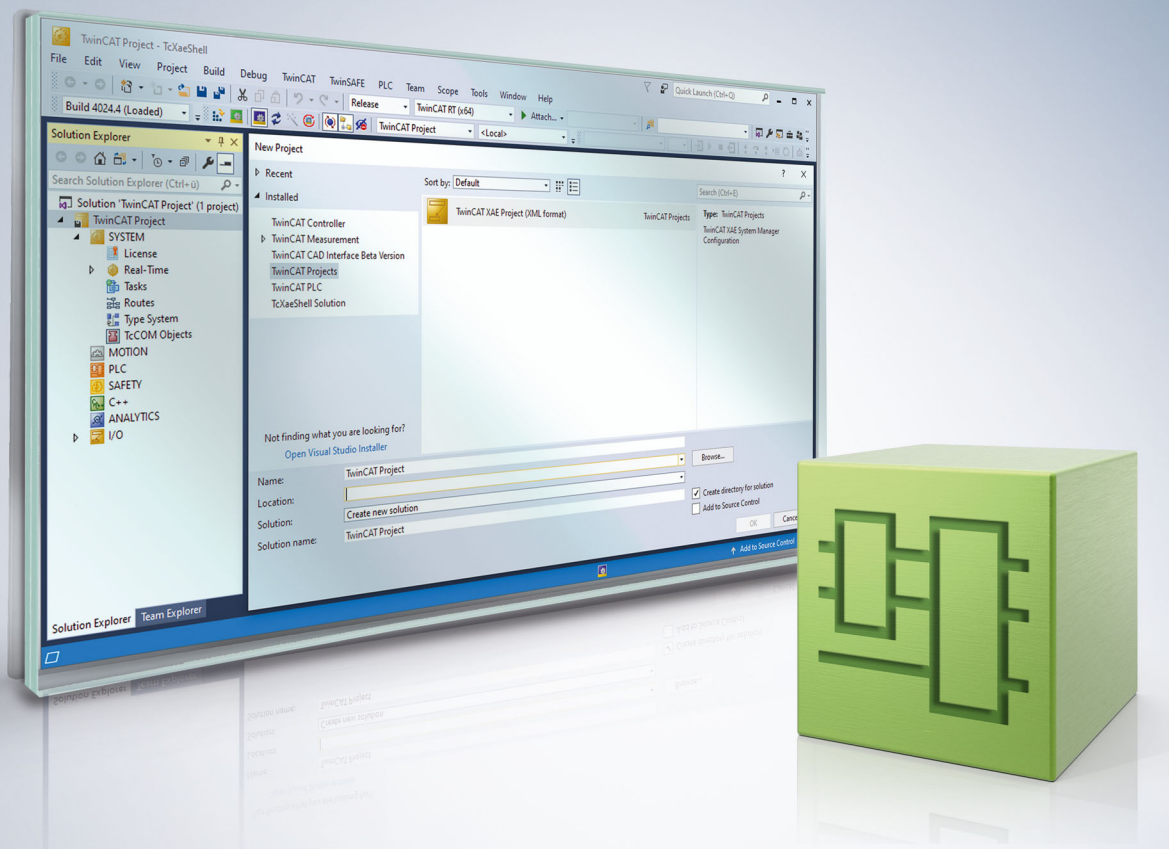
BECKHOFF New Automation Technology

Manual | EN

TE1000

TwinCAT 3 | PLC Libary: Tc2_EnOcean



2023-08-31 | Version: 1.6.1

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 For your safety 6
1.3 Notes on information security 7
2 Introduction 8
3 EnOcean 9
3.1 Range planning 9
3.2 Approval of EnOcean wireless technology 10
4 Programming 11
4.1 POUs 12
4.1.1 KL6021-0023 14
4.1.2 KL6581 22
4.2 DUTs 34
4.2.1 KL6021-0023 35
4.2.2 KL6581 37
4.3 Integration into TwinCAT 43
4.3.1 KL6581 with CX5120 43
5 Appendix 46
5.1 Support and Service 46

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Introduction

The Tc2_EnOcean library is a TwinCAT PLC library for data exchange with EnOcean devices. The library should only be used in conjunction with a KL6021-0023 or KL6581.

The user of this library requires basic knowledge of the following:

- TwinCAT XAE
- PC and network knowledge
- Structure and properties of the Beckhoff Embedded PC and its Bus Terminal system
- Technology of EnOcean devices
- Relevant safety regulations for building technical equipment

This software library is intended for building automation system partners of Beckhoff Automation GmbH & Co. KG. The system partners operate in the field of building automation and are concerned with the installation, commissioning, expansion, maintenance and service of measurement, control and regulating systems for the technical equipment of buildings.

The Tc2_EnOcean library is usable on all hardware platforms that support TwinCAT 3.1 or higher.

Hardware documentation for KL6021-0023 and KL6581 in the Beckhoff Information System.

3 EnOcean

The EnOcean radio technology makes a far-reaching signal with low quantities of ambient energy possible. With 50 µWs, a standard EnOcean radio module can easily transmit a signal over a distance of 300 m (in the free field). The signal period for an EnOcean telegram is approx. 1 thousandth of second.

- Licence-free 868 MHz frequency band with 1% duty cycle
- Multiple telegram transmission with checksum
- Short telegrams (approx. 1 ms) lead to a small probability of collision
- Long range: 30 m inside buildings or 300 m in the free field
- Repeater available for extensions
- Unidirectional and bidirectional communication
- High data transmission rates of 125 kbit/s
- Low 'data overhead'
- ASK modulation
- Radio protocol is defined and integrated in modules
- Sensor profiles specified and adhered to by users
- Unique transmission ID (32-bit)
- No interference with DECT, WLAN, PMR systems etc.
- System design verified in industrial environment

Protocol structure

Protocol	Description	Length
ORG	Telegram type	1 byte
DB_3	Data byte 3	1 byte
DB_2	Data byte 2	1 byte
DB_1	Data byte 1	1 byte
DB_0	Data 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	Information status	1 byte

3.1 Range planning

Please follow the recommendations of the EnOcean Alliance (see www.enocean.org) when placing the EnOcean devices. Adherence to the recommendations is conducive to an optimum range and high noise immunity.

Attenuation of different materials

Material	Attenuation
Wood, plaster, uncoated glass (without metal)	0...10 %
Brick, chipboard	5...35 %
Concrete with iron reinforcement	10...90 %
Metal, aluminum cladding	90..100 %

Range

Material	Range
Line of sight	Typically 30 m in corridors, up to 100 m in halls
Plasterboard walls/wood	Typically 30 m, through max. 5 walls
Brick walls/aerated concrete	Typically 20 m, through max. 3 walls
Reinforced concrete walls/ceilings	Typically 10 m, through max. 1 wall/ceiling

Placement of the KL6583 module

The KL6583 module contains transmitter, receiver and antenna.

Distances

The distance to a reinforced concrete ceiling should be at least 50 cm and to a wall 10 cm.

Do not attach or screw the KL6583 module to a metal plate!

Environmental conditions

Furthermore, the environmental conditions are to be adhered to:

- Maximum air humidity 95% without condensation
- Ambient temperature 0 - 55°C

3.2 Approval of EnOcean wireless technology

Check the admissibility of the operation in your country

The KL6583 EnOcean transceivers can be operated in following countries without registration or fee: **KL6583-0000: European Union** and **Switzerland**

KL6583-0100: USA and **Canada**

Permission for use in other countries must be clarified explicitly!

KL6583-0100 for USA and Canada

Includes IC: 5731A-TCM320C

Includes FCC ID: SZV-TCM320C

The affected device complies with part 15 of the FCC rules.
The operation is subject to the following conditions:

- this device must not cause adverse interference, and
- this device must absorb all received interference, including interference that would impair the operation.

4 Programming

POUs/KL6021-0023

POUs	Description
FB_EnOceanReceive [▶ 14]	Communication with KL6021-0023

POUs/KL6021-0023/Read

POUs	Description
FB_EnOceanPTM100 [▶ 15]	Receives the signals of a PTM100 module
FB_EnOceanPTM200 [▶ 16]	Receives the signals of a PTM200 module
FB_EnOceanSTM100 [▶ 18]	Receives the signals of a STM100 module (obsolete)
FB_EnOceanSTM100Generic [▶ 19]	Receives the signals of a STM100 module
FB_EnOceanSTM250 [▶ 21]	Receives the signals of a STM250 module

POUs/KL6581

Function blocks	Description
FB_KL6581 [▶ 22]	Communication with a KL6581

POUs/KL6581/Read

POUs	Description
FB_Rec_Generic [▶ 24]	Receives all types of EnOcean telegrams
FB_Rec_1BS [▶ 24]	Receives data with ORG telegram 6.Typical EnOcean device: Window contact.
FB_Rec_RPS_Switch [▶ 25]	Receives data with ORG telegram 5. Typical EnOcean device: Buttons.
FB_Rec_RPS_Window_Handle [▶ 26]	Receives data with ORG telegram 5.Typical EnOcean device: Window handle.

POUs/KL6581/Send

POUs	Description
FB_Send_Generic [▶ 27]	All kinds of EnOcean data telegrams can be sent with this block.
FB_Send_4BS [▶ 28]	Sends EnOcean telegrams in the 4BS format.
FB_Send_RPS_Switch [▶ 28]	Sends EnOcean telegrams in the format of a button.
FB_Send_RPS_SwitchAuto [▶ 29]	This function block sends data such as those from a switch.

POUs/KL6581/Other

POUs	Description
FB_EnOcean_Search [▶ 30]	This function block recognizes all EnOcean devices within its range and displays them.
FB_Rec_Teach_In [▶ 31]	This function block indicates if the LRN bit in an EnOcean telegram is set, independent of its EnOcean ID.
FB_Rec_Teach_In_Ex [▶ 32]	This function block indicates pressing of the Learn button at an EnOcean device.

POUs/KL6581/Function

Function blocks	Description
F_Byte_To_Temp [▶ 33]	This function converts a byte raw value into a REAL variable.
F_Byte_To_TurnSwitch [▶ 33]	This function converts a raw byte value to a Boolean array.

DUTs/KL6021-0023/Hardware Types

Data types	Description
ST_EnOceanInData [▶ 36]	Process image of the KL6021-0023 inputs
ST_EnOceanOutData [▶ 37]	Process image of the KL6021-0023 outputs

DUTs/KL6021-0023

Data types	Description
E_EnOceanRotarySwitch [▶ 35]	State of the rotary-switch on the transmitting-module
E_EnOceanSensorType [▶ 35]	Sensor type
ST_EnOceanReceivedData [▶ 36]	Internal structure

DUTs/KL6581/hardware types

Data types	Description
KL6581_Input [▶ 39]	Process image of the KL6581 inputs
KL6581_Output [▶ 39]	Process image of the KL6581 outputs

DUTs/KL6581

Data types	Description
AR_EnOceanWindow [▶ 40]	State of the window
E_ENOCEAN_Org [▶ 38]	EnOcean telegram type
E_KL6581_Err [▶ 38]	Error messages
STR_EnOceanSwitch [▶ 41]	State of the buttons
STR_KL6581 [▶ 41]	Internal structure
STR_Teach_In [▶ 42]	Data structure manufacturer ID, type and profile
STREnOceanTurnSwitch [▶ 42]	Position of the rotary switch at the room control unit

4.1 POUs

KL6021-0023

POUs	Description
FB_EnOceanReceive [▶ 14]	Communication with KL6021-0023

KL6021-0023/Read

POUs	Description
FB_EnOceanPTM100 [▶ 15]	Receives the signals of a PTM100 module
FB_EnOceanPTM200 [▶ 16]	Receives the signals of a PTM200 module
FB_EnOceanSTM100 [▶ 18]	Receives the signals of a STM100 module (obsolete)
FB_EnOceanSTM100Generic [▶ 19]	Receives the signals of a STM100 module
FB_EnOceanSTM250 [▶ 21]	Receives the signals of a STM250 module

KL6581

POUs	Description
FB_KL6581 [▶ 22]	Main block for the communication with the KL6581 and the connected KL6583 modules

KL6581/Read

POUs	Description
FB_Rec_Generic [▶ 24]	Receives all types of EnOcean telegrams
FB_Rec_1BS [▶ 24]	Receives data with ORG telegram 6.Typical EnOcean device: Window contact.
FB_Rec_RPS_Switch [▶ 25]	Receives data with ORG telegram 5. Typical EnOcean device: Buttons.
FB_Rec_RPS_Window_Handle [▶ 26]	Receives data with ORG telegram 5.Typical EnOcean device: Window handle.

KL6581/Send

POUs	Description
FB_Send_Generic [▶ 27]	All kinds of EnOcean data telegrams can be sent with this block.
FB_Send_4BS [▶ 28]	Sends EnOcean telegrams in the 4BS format.
FB_Send_RPS_Switch [▶ 28]	Sends EnOcean telegrams in the format of a button.
FB_Send_RPS_SwitchAuto [▶ 29]	This function block sends data such as those from a switch.

KL6581/Other

Function blocks	Description
FB_EnOcean_Search [▶ 30]	This function block recognizes all EnOcean devices within its range and displays them.
FB_Rec_Teach_In [▶ 31]	This function block indicates if the LRN bit in an EnOcean telegram is set, independent of its EnOcean ID.
FB_Rec_Teach_In_Ex [▶ 32]	This function block indicates pressing of the Learn button at an EnOcean device.

KL6581/Function

Function blocks	Description
F_Byte_To_Temp [▶ 33]	This function converts a byte raw value into a REAL variable.
F_Byte_To_TurnSwitch [▶ 33]	This function converts a raw byte value to a Boolean array.

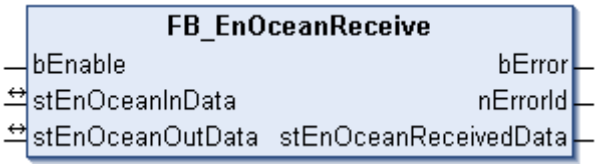
4.1.1 KL6021-0023

Function blocks	Description
FB_EnOceanReceive [▶ 14]	Communication with a KL6021-0023

Read

Function blocks	Description
FB_EnOceanPTM100 [▶ 15]	Receives the signals of a PTM100 module
FB_EnOceanPTM200 [▶ 16]	Receives the signals of a PTM200 module
FB_EnOceanSTM100 [▶ 18]	Receives the signals of a STM100 module (outdated)
FB_EnOceanSTM100Generic [▶ 19]	Receives the signals of a STM100 module
FB_EnOceanSTM250 [▶ 21]	Receives the signals of a STM250 module

4.1.1.1 FB_EnOceanReceive



The function block *FB_EnOceanReceive()* is a receive block, which makes the telegrams sent by the EnOcean modules available in the structure *stEnOceanReceivedData*. This structure can then be analyzed with further blocks. The documentation for this blocks also includes sample programs, which illustrate the operating principle.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality.

VAR_OUTPUT

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

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

stEnOceanReceivedData: This structure contains the received data (see ST_EnOceanReceivedData [▶ 36]).

VAR_IN_OUT

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

stEnOceanInData: Is linked with the input addresses of the KL6021-0023 in the System Manager (see ST_EnOceanInData [▶ 36]).

stEnOceanOutData: Is linked with the output addresses of the KL6021-0023 in the System Manager (see ST_EnOceanOutData [▶ 37]).

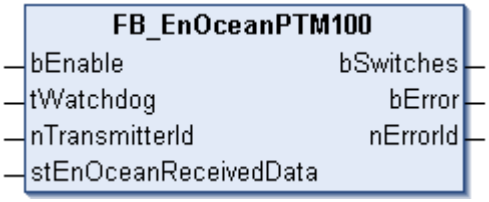
Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.2 Read

Function blocks	Description
FB_EnOceanPTM100 [▶ 15]	Receives the signals of a PTM100 module
FB_EnOceanPTM200 [▶ 16]	Receives the signals of a PTM200 module
FB_EnOceanSTM100 [▶ 18]	Receives the signals of a STM100 module (outdated)
FB_EnOceanSTM100Generic [▶ 19]	Receives the signals of a STM100 module
FB_EnOceanSTM250 [▶ 21]	Receives the signals of a STM250 module

4.1.1.2.1 FB_EnOceanPTM100



The function block *FB_EnOceanPTM100()* provides a user-friendly evaluation of the state of an EnOcean PTM100 module. The function block FB_EnOceanReceive() [▶ 14] is required for this purpose.

In contrast to the PTM200 and PTM250 modules, only one button at a time can be pressed in the PTM100 module. In addition, the PTM100 module supports eight buttons, not four.

A new instance of this function block must be created for each button module used.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality, and all outputs are set to 0 or FALSE.

tWatchdog: Monitoring time. Within this time, new information must reach this block via the input *stEnOceanReceivedData* described below. If this time is set to t#0s, the watchdog function is inactive.

nTransmitterId: ID of the EnOcean module, to which the block should respond.

stEnOceanReceivedData: Information and required connection to the EnOcean receive block FB_EnOceanReceive() [▶ 14]. This information is stored in a structure (see ST_EnOceanReceivedData [▶ 36]).

VAR_OUTPUT

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

bSwitches: This field of 8 Boolean values describes the states of the 8 buttons on the button module.

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

The following sample program illustrates the operating principle of this block:

```
PROGRAM MAIN
VAR
    fbEnOceanReceive  : FB_EnOceanReceive;
    fbEnOceanPTM100_1 : FB_EnOceanPTM100;
```

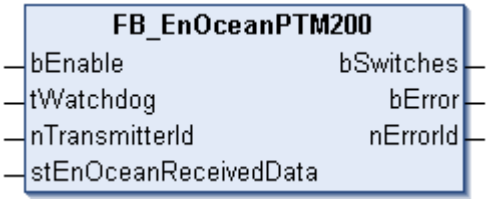
```
    fbEnOceanPTM100_2 : FB_EnOceanPTM100;
    bSwitches1   : ARRAY    [0..7]    OF    BOOL;
    bSwitches2_1 : BOOL;
    bSwitches2_2 : BOOL;
    bSwitches2_3 : BOOL;
    bSwitches2_4 : BOOL;
    bSwitches2_5 : BOOL;
    bSwitches2_6 : BOOL;
    bSwitches2_7 : BOOL;
    bSwitches2_8 : BOOL;
END_VAR

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 this example program two transmitter modules (PTM100) are queried: a transmitter module with the transmitter ID 16#C4 and another module with the transmitter ID 16#C5. A function block FB_EnOceanPTM100 is created for both transmitter modules. Both function blocks obtain their information from the upstream receive block FB_EnOceanReceive and are only active (input bEnable) if the receive block is active and not in error. The buttons of the first transmitter module are assigned to a Boolean array bSwitches1 for further analysis, while the buttons of the second transmitter module are assigned to individual Boolean variables bSwitches2_1 to bSwitches2_8; both options are conceivable.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.2.2 FB_EnOceanPTM200



The function block *FB_EnOceanPTM200*() provides a user-friendly evaluation of the state of an EnOcean PTM200 or PTM250 module. The function block FB_EnOceanReceive() [▶ 14] is required for this purpose.

In contrast to the PTM100 module, in the PTM200/250 module two buttons can be pressed simultaneously. In addition, the PTM200/250 module supports four, not eight buttons.

A new instance of this function block must be created for each button module used.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality, and all outputs are set to 0 or FALSE.

tWatchdog: Monitoring time. Within this time, new information must reach this block via the input *stEnOceanReceivedData* described below. If this time is set to t#0s, the watchdog function is inactive.

nTransmitterId: ID of the EnOcean module, to which the block should respond.

stEnOceanReceivedData: Information and required connection to the EnOcean receive block FB_EnOceanReceive() [▶ 14]. This information is stored in a structure (see ST_EnOceanReceivedData [▶ 36]).

VAR_OUTPUT

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

bSwitches: This field of 4 Boolean values describes the states of the 4 buttons on the button module.

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

The following sample program illustrates the operating principle of this block:

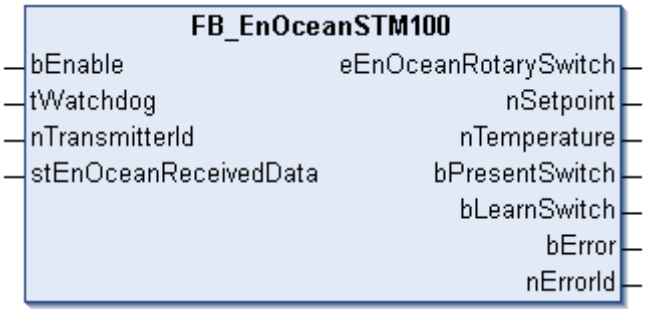
```
PROGRAM MAIN
VAR
    fbEnOceanReceive  : FB_EnOceanReceive;
    fbEnOceanPTM100_1 : FB_EnOceanPTM200;
    fbEnOceanPTM100_2 : FB_EnOceanPTM200;
    bSwitches1   : ARRAY [0..3] OF BOOL;
    bSwitches2_1 : BOOL;
    bSwitches2_2 : BOOL;
    bSwitches2_3 : BOOL;
    bSwitches2_4 : BOOL;
END_VAR

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 this example program two transmitter modules (PTM200/PTM250) are queried; one with the transmitter ID 16#C6, another one with the transmitter ID 16#C7. A function block *FB_EnOceanPTM200* is created for both transmitter modules. Both function blocks obtain their information from the upstream receive block FB_EnOceanReceive [▶ 14] and are only active (input *bEnable*) if the receive block is active and not in error. The buttons of the first transmitter module are assigned to a Boolean array *bSwitches1* for further analysis, while the buttons of the second transmitter module are assigned to individual Boolean variables *bSwitches2_1* to *bSwitches2_8*; both options are conceivable.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.2.3 FB_EnOceanSTM100



Outdated

For new projects the block FB_EnOceanSTM100Generic() [▶ 19] should be used!

The function block *FB_EnOceanSTM100()* provides a user-friendly evaluation of the data of an EnOcean STM100 module. The function block FB_EnOceanReceive() [▶ 14] is required for this purpose.

A new instance of this function block must be created for each button module used.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality, and all outputs are set to 0 or FALSE.

tWatchdog: Monitoring time. Within this time, new information must reach this block via the input *stEnOceanReceivedData* described below. If this time is set to t#0s, the watchdog function is inactive.

nTransmitterId: ID of the EnOcean module, to which the block should respond.

stEnOceanReceivedData: Information and required connection to the EnOcean receive block FB_EnOceanReceive() [▶ 14]. This information is stored in a structure (see ST_EnOceanReceivedData [▶ 36]).

VAR_OUTPUT

```
eEnOceanRotarySwitch : E_EnOceanRotarySwitch;
nSetpoint            : INT;
nTemperature         : INT;
bPresentSwitch       : BOOL;
bLearnSwitch         : BOOL;
bError               : BOOL := FALSE;
nErrorId             : UDINT := 0;
```

eEnOceanRotarySwitch: The value at this output describes the position of the rotary switch at the room control unit (see E_EnOceanRotarySwitch [▶ 35]).

nSetpoint: This output variables indicates the value set at the device. The value range is -100 to +100.

nTemperature: This output provides the measured temperature in 1/10 °C, with a measuring range of 0 °C to 40 °C. If the watchdog is triggered, the block suspects a broken wire type error, and the value is set to 850 °C.

bPresentSwitch: If the presence button at the room control unit is activated, this output becomes *TRUE*.

bLearnSwitch: If the teach-in button at the room control unit is activated, this output becomes *TRUE*.

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

The following sample program illustrates the operating principle of this block:

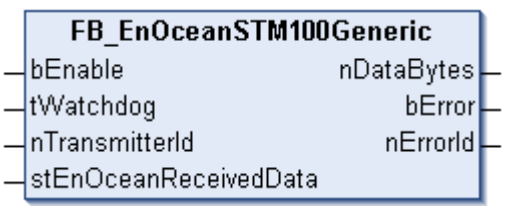
```
PROGRAM MAIN
VAR
    fbEnOceanReceive  : FB_EnOceanReceive;
    fbEnOceanSTM100_1 : FB_EnOceanSTM100;
    fbEnOceanSTM100_2 : FB_EnOceanSTM100;
    nTemperature : ARRAY [1..2] OF INT;
    nSetpoint : ARRAY [1..2] OF INT;
    nStateRotarySwitch : ARRAY [1..2] OF E_EnOceanRotarySwitch;
    bPresentSwitch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

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

fbEnOceanSTM100_1(
    bEnable := NOT fbEnOceanReceive.bError AND fbEnOceanReceve.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 this example program two room control units are queried; one with the transmitter ID 16#000000C4 and another one with the transmitter ID 16#000000C5. A function block *FB_EnOceanSTM100* is created for both modules. Both function blocks obtain their information from the upstream receive block FB_EnOceanReceive [▶ 14] and are only active (input *bEnable*) if the receive block is active and not in error. The first device monitored with the watchdog function. New values have to be transferred to the controller within 1 hour; the second device is programmed without watchdog monitoring. The output values at the function blocks are assigned flags for further evaluation.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.2.4 FB_EnOceanSTM100Generic



The function block *FB_EnOceanSTM100Generic*() provides a user-friendly evaluation of the data of an EnOcean STM100 module. The function block FB_EnOceanReceive() [▶ 14] is required for this purpose.

A new instance of this function block must be created for each button module used.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality, and all outputs are set to 0 or FALSE.

tWatchdog: Monitoring time. Within this time, new information must reach this block via the input *stEnOceanReceivedData* described below. If this time is set to t#0s, the watchdog function is inactive.

nTransmitterId: ID of the EnOcean module, to which the block should respond.

stEnOceanReceivedData: Information and required connection to the EnOcean receive block FB_EnOceanReceive() [▶ 14]. This information is stored in a structure (see ST_EnOceanReceivedData [▶ 36]).

VAR_OUTPUT

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

nDataBytes: 4-byte array with the user data sent by the STM100 module. The purpose of the individual bytes is manufacturer-specific.

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

The following sample program illustrates the operating principle of this block:

```
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];
```

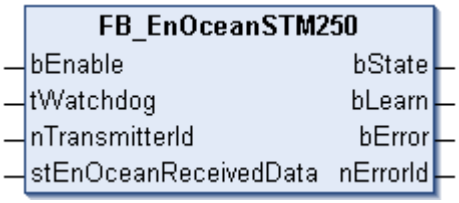
In this example program two EnOcean transmitter modules are queried; one with the transmitter ID 16#000000C4, another one with the transmitter ID 16#000000C5. A function block *FB_EnOceanSTM100Generic* is created for both transmitters. Both function blocks obtain their information

from the upstream receive block FB_EnOceanReceive [▶ 14] and are only active (input bEnable) if the receive block is active and not in error. The first device monitored with the watchdog function. New values have to be transferred to the controller within 1 hour; the second device is programmed without watchdog monitoring. The output values at the function blocks are assigned variables for further evaluation. Before the values can be used further, they have to be scaled to physical values. Details of the conversion can be found in the data sheet for the sensor.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.2.5 FB_EnOceanSTM250



The function block *FB_EnOceanSTM250()* provides a user-friendly evaluation of the data of an EnOcean STM250 module. The function block FB_EnOceanReceive() [▶ 14] is required for this purpose.

A new instance of this function block must be created for each STM100 module used.

VAR_INPUT

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

bEnable: A positive signal at this input activates the block. A negative signal at the input disables the block functionality, and all outputs are set to 0 or FALSE.

tWatchdog: Monitoring time. Within this time, new information must reach this block via the input *stEnOceanReceivedData* described below. If this time is set to t#0s, the watchdog function is inactive.

nTransmitterId: ID of the EnOcean module, to which the block should respond.

stEnOceanReceivedData: Information and required connection to the EnOcean receive block FB_EnOceanReceive() [▶ 14]. This information is stored in a structure (see ST_EnOceanReceivedData [▶ 36]).

VAR_OUTPUT

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

bState: Upon activation of the reed contact in the STM250 module, this output becomes *TRUE* (contact closed).

bLearn: This output becomes *FALSE* if the teach-in button at the STM250 module is activated.

bError: this output goes TRUE as soon as an error occurs. This error is described via the variable *nErrorId*.

nErrorId: Describes the error type (see error codes [▶ 22]).

The following sample program illustrates the operating principle of this block:

```
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 this example program an STM250 module with the transmitter ID 16#000008CA is queried. To this end the function block *FB_EnOceanSTM250* is created. This function blocks obtains its information from the upstream receive block FB_EnOceanReceive [▶ 14] and is only active (input *bEnable*) if the receive block is active and not in error. The output values at the function blocks are assigned variables for further evaluation.

Requirements

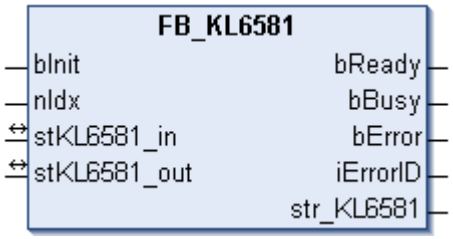
Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.1.3 Error codes

Value (hex)	Description
0x0000	No error.
0x0001	Checksum error.
0x0002	Watchdog monitoring.
0x0003	Buffer overflow (in the KL6023)
0x0004	No data received yet from sensor receive.

4.1.2 KL6581

4.1.2.1 FB_KL6581



This function block takes care of communication with the KL6581 EnOcean bus terminal. The KL6581 is configured and the data exchange with the EnOcean network is started via this block.

Restrictions

- Only one call per instance
- Call must be made once per PLC cycle
- Instance must be called in the same PLC task as the send and receive blocks assigned to it
- Maximum 64 instances per PLC project allowed

VAR_INPUT

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

bInit: Activates the block that configures the KL6301 and then activates the data exchange.

nIdx: The idx number must be unique for each KL6581, if more than one Bus Terminal per PLC program is used (valid values: 1...64).

VAR_OUTPUT

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

bReady: The block is ready for sending and receiving data.

bBusy: The block is active.

bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

iErrorID: Describes the error type (see E_KL6581_Err [▶ 38]).

str_KL6581: Is linked to the send and receive blocks (see STR_KL6581 [▶ 41]).

VAR_IN_OUT

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

stKL6581_in: Is linked to the input addresses of the KL6581 in the System Manager (see KL6581_Input [▶ 39]).

stKL6581_out: Is linked to the output addresses of the KL6581 in the System Manager (see KL6581_Output [▶ 39]).

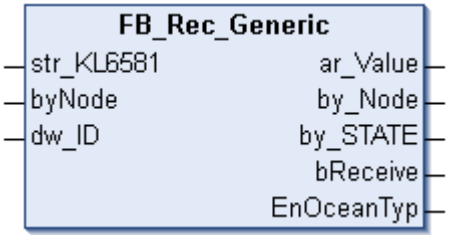
Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.2 Read

Function blocks	Description
FB_Rec_Generic [▶ 24]	Receives all types of EnOcean telegrams
FB_Rec_1BS [▶ 24]	Receives data with ORG telegram 6.Typical EnOcean device: Window contact
FB_Rec_RPS_Switch [▶ 25]	Receives data with ORG telegram 5.Typical EnOcean device: Button
FB_Rec_RPS_Window_Handle [▶ 26]	Receives data with ORG telegram 5.Typical EnOcean device: Window handle

4.1.2.2.1 FB_Rec_Generic



This function block receives all data that were received via EnOcean. This block can be used for all kinds of EnOcean telegrams.

The user must interpret the data himself. The manufacturer's documentation for the sending EnOcean device is necessary for this.

VAR_INPUT

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

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

dw_ID: EnOcean ID to be received.

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 data.

by_Node: Node number of the KL6583 that has received the EnOcean telegram.

by_STATE: EnOcean STATUS field.

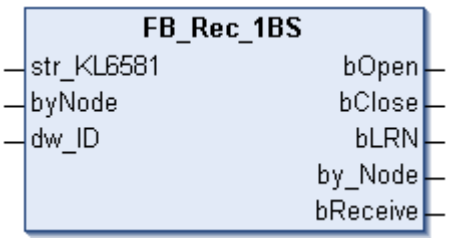
bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

EnOceanTyp: EnOcean ORG field (see E_EnOcean_Org [▶ 38]).

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.2.2 FB_Rec_1BS



This function block receives data that were received via EnOcean. This block is used for integration of window contacts, for example (ORG field 6).

VAR_INPUT

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

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

dw_ID: EnOcean ID to be received.

VAR_OUTPUT

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

bOpen: Contact open.

bClose: Contact closed.

bLRN: LRN button pressed.

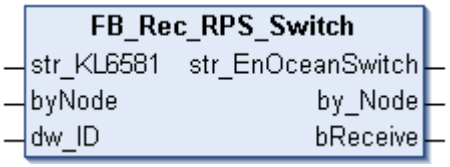
by_Node: Node number of the KL6583 that has received the EnOcean telegram.

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.2.3 FB_Rec_RPS_Switch



This function block receives data from a switch that were received via EnOcean. The block outputs the data in a data structure (ORG field 5).

VAR_INPUT

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

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

dw_ID: EnOcean ID to be received.

VAR_OUTPUT

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

str_EnOceanSwitch: Switch data (see STR_EnOceanSwitch [▶ 41]).

by_Node: Node number of the KL6583 that has received the EnOcean telegram.

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.2.4 FB_Rec_RPS_Window_Handle

This function block receives data from a window handle that were received via EnOcean. The block outputs the data in a data structure (ORG field 5).

VAR_INPUT

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

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

dw_ID: EnOcean ID to be received.

VAR_OUTPUT

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

ar_Data: window handle data (see set_EnOceanWindow [▶ 40]).

by_Node: Node number of the KL6583 that has received the EnOcean telegram.

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

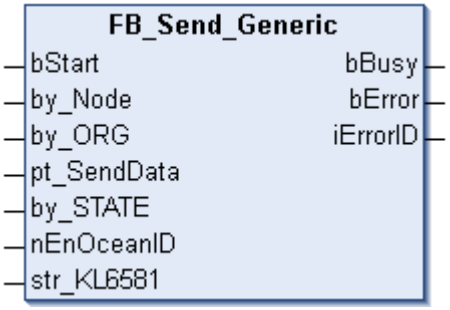
Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.3 Send

Function blocks	Description
FB_Send_Generic [▶ 27]	Sends arbitrary EnOcean telegrams
FB_Send_4BS [▶ 28]	Sends EnOcean telegrams in the 4BS format
FB_Send_RPS_Switch [▶ 28]	Sends EnOcean telegrams in the format of a button
FB_Send_RPS_SwitchAuto [▶ 29]	Sends EnOcean telegrams in the format of a button

4.1.2.3.1 FB_Send_Generic



This function block sends data via EnOcean. The type and the data contents are arbitrary. All kinds of EnOcean data telegrams can be sent with this block.

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: A rising edge sends the data.

by_Node: Address of the KL6583 module to which the telegram is to be sent (valid values: 1...8).

by_ORG: ORG field of the EnOcean telegram (see E_EnOcean_Org [▶ 38]).

pt_SendData: Pointer to the data to be sent. ADR is used to determine the pointer address. The pointer must point to 4-byte variable.

by_STATE: EnOcean STATE. Can be changed by the TCM module.

nEnOceanID: Virtual EnOcean ID. A value of 0…127 is added to the real EnOcean ID (valid values: 0...127).

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bBusy: The block is active. No new data can be sent at this stage.

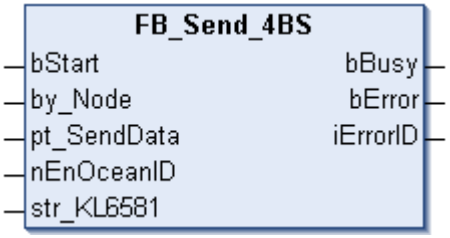
bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

iErrorID: Describes the error type (see E_KL6581_Err [▶ 38]).

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.3.2 FB_Send_4BS



This function block sends data via EnOcean. The ORG field is set permanently to 7.

VAR_INPUT

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

bStart: A rising edge sends the data.

by_Node: Address of the KL6583 module to which the telegram is to be sent (valid values: 1...8).

pt_SendData: Pointer to the data to be sent. ADR is used to determine the pointer address. The pointer must point to 4-byte variable.

nEnOceanID: Virtual EnOcean ID. A value of 0…127 is added to the real EnOcean ID (valid values: 0...127).

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bBusy: The block is active. No new data can be sent at this stage.

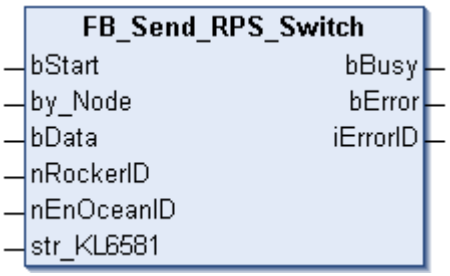
bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

iErrorID: Describes the error type (see E_KL6581_Err [▶ 38]).

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.3.3 FB_Send_RPS_Switch



This block sends EnOcean telegrams in the format of a button. The value of *bData* is sent with a positive edge of *bStart*. In order to simulate a keystroke, the block usually has to be started twice, once with *bData* = TRUE, once with *bData* = FALSE. For simpler handling the block FB_Send_RPS_SwitchAuto() [▶ 29] can be used.

VAR_INPUT

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

bStart: A rising edge sends the data.

by_Node: Address of the KL6583 module to which the telegram is to be sent (valid values: 1...8).

bData: Value to be transferred.

nRockerID: Button number, valid values 0 to 3.

nEnOceanID: Virtual EnOcean ID. A value of 0…127 is added to the real EnOcean ID (valid values: 0...127).

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bBusy: The block is active. No new data can be sent at this stage.

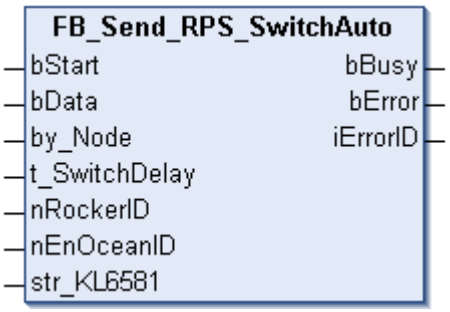
bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

iErrorID: Describes the error type (see E_KL6581_Err [▶ 38]).

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.3.4 FB_Send_RPS_SwitchAuto



This block sends EnOcean telegrams in the format of a button. The value of *bData* is sent with a positive edge of *bStart*. The signal "Release button" is sent once the time *t_SwitchDelay* has elapsed.

VAR_INPUT

```
bStart        : BOOL;
bData         : BOOL;
by_Node       : BYTE;
t_SwitchDelay : TIME := T#100ms;
```

```
nRockerID     : INT;
nEnOceanID    : BYTE;
str_KL6581    : STR_KL6581;
```

bStart: A rising edge sends the data.

bData: Value to be transmitted.

by_Node: Address of the KL6583 module to which the telegram is to be sent (valid values: 1...8).

t_SwitchDelay: How long the button has to be pressed.

nRockerID: Button number, valid values 0 to 3.

nEnOceanID: Virtual EnOcean ID. A value of 0…127 is added to the real EnOcean ID (valid values: 0...127).

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bBusy: The block is active. No new data can be sent at this stage.

bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

iErrorID: Describes the error type (see E_KL6581_Err [▶ 38]).

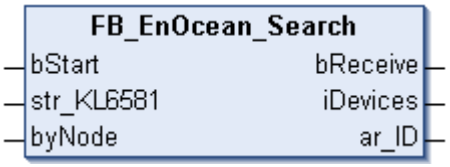
Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.4 Other

Function blocks	Description
FB_EnOcean_Search [▶ 30]	Function block recognizes all EnOcean devices within its range and displays them.
FB_Rec_Teach_In [▶ 31]	This function block indicates if the LRN bit in an EnOcean telegram is set, independent of its EnOcean ID.
FB_Rec_Teach_In_Ex [▶ 32]	This function block indicates pressing of the Learn button at an EnOcean device.

4.1.2.4.1 FB_EnOcean_Search



This function block displays all EnOcean IDs that it has received and enters them in a reception array, (*ar_ID)*. Up to 256 EnOcean devices can be recognized. Alternatively the block can also be created separately for each KL6583. This allows you to recognize whether an EnOcean device is received by several KL6583s.

VAR_INPUT

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

bStart: If TRUE the block is activated, if FALSE it is deactivated.

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

VAR_OUTPUT

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

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

iDevices: Number of EnOcean devices found.

ar_ID: EnOcean IDs that were found.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.4.2 FB_Rec_Teach_In

This function block indicates when a learn button is pressed on an EnOcean device. If the flag *bLearnType* is set, further information about the EnOcean device can be read. This function must be provided by the EnOcean device. So far, however, it is only supported by very few EnOcean devices.

VAR_INPUT

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

bStart: If TRUE the block is activated, if FALSE it is deactivated.

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bLearnType: If the bit is set you will find further data in the *str_Teach_In* structure.

by_Node: Number of EnOcean devices found.

dw_ID: EnOcean ID for which the teach-in button was pressed.

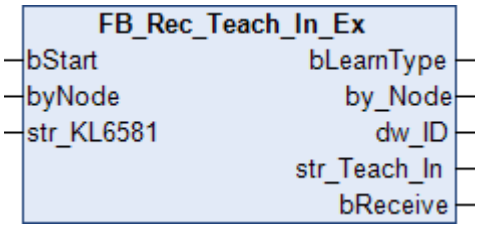
str_Teach_In: Data structure, profile, type and manufacturer ID (see STR_Teach_In [▶ 41]).

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.4.3 FB_Rec_Teach_In_Ex



This function block indicates when a learn button is pressed on an EnOcean device. If the flag *bLearnType* is set, further information about the EnOcean device can be read. This function must be provided by the EnOcean device. So far, however, it is only supported by very few EnOcean devices.

In addition to the FB_Rec_Teach_In() [▶ 31] function block, the system checks for an EEP telegram.

VAR_INPUT

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

bStart: If TRUE the block is activated, if FALSE it is deactivated.

byNode: Filter - if the value is zero the EnOcean telegrams from all KL6583s are received. If a value of 1 to 8 is entered, only the data from the corresponding KL6583 are received.

str_KL6581: Is linked with the data structure of block FB_KL6581() [▶ 22] (see STR_KL6581 [▶ 41]).

VAR_OUTPUT

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

bLearnType: If the bit is set you will find further data in the *str_Teach_In* structure.

by_Node: Number of EnOcean devices found.

dw_ID: EnOcean ID for which the teach-in button was pressed.

str_Teach_In: Data structure, function, type and manufacturer ID (see STR_Teach [▶ 42]).

bReceive: On receiving an EnOcean telegram this value is set to FALSE for one cycle.

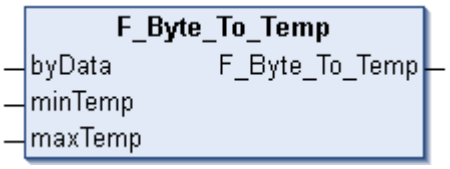
Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.32	Tc2_EnOcean from v3.4.6.0

4.1.2.5 Function

Function blocks	Description
F_Byte_To_Temp [▶ 33]	This function converts a byte raw value into a REAL variable.
F_Byte_To_TurnSwitch [▶ 33]	This function converts a raw byte value to a Boolean array.

4.1.2.5.1 F_Byte_to_Temp : REAL



This function converts a byte raw value into a REAL variable.

In EnOcean, temperature data are transmitted in a certain format, which is one byte in size. These data are usually scaled to a certain temperature value.

For example, a value is transmitted from a range of values from 0 to 40°C. The minimum and maximum data value and the raw value are transferred to the function. The output of the function then outputs the temperature as REAL variable.

VAR_INPUT

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

byData: Raw data.

minTemp: Minimum temperature.

maxTemp: Maximum temperature.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.5.2 F_Byte_to_TurnSwitch

This function converts a raw byte value to a Boolean array in the form of a data structure (see STREnOceanTurnSwitch [▶ 42]).

VAR_INPUT

```
byData : BYTE;
```

byData: Raw data.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.1.2.6 Error codes

Value (hex)	Value (dec)	Value (enum)	Description
0x0000	0	NO_ERROR	No error at the block.
0x000A	10	KL6581_WrongTerminal	Incorrect terminal connected.
0x0010	16	KL6581_WatchdogError	Timeout during initialization of function block FB_KL6581() [▶ 22].
0x0011	17	KL6581_NoComWithKL6581	This message usually means that there is no connection to the terminal. Terminal linked to the variables in the System Manager? Terminal plugged in incorrectly? Everything corrected, everything translated and re-read into the System Manager?
0x0012	18	KL6581_idx_number_not_OK	The input variable nIdx of block FB_KL6581() is greater than 64.
0x0013	19	KL6581_Switch_to_Stopp	The terminal has exited the data exchange with the EnOcean transmitter and receiver KL6583-0000, no EnOcean data was sent or received.
0x0014	20	KL6581_not_ready	Internal message for the function blocks connected to the FB_KL6581().
0x0015	21	KL6581_No_KL6853_Found	No KL6583 is connected to the EnOcean master terminal KL6581, or the communication does not exist.
0x0016	22	KL6581_TransmissionError	Data could not sent; check the address of the KL6583, or KL6583 not ready for operation.

4.2 DUTs

KL6021-0023/Hardware Types

Data types	Description
ST_EnOceanInData [▶ 36]	Process image of the inputs
ST_EnOceanOutData [▶ 37]	Process image of the outputs

KL6021-0023

Data types	Description
E_EnOceanRotarySwitch [▶ 35]	State of the rotary-switch on the transmitting-module
E_EnOceanSensorType [▶ 35]	Sensor type
ST_EnOceanReceivedData [▶ 36]	Internal structure

KL6581/Hardware Types

Data types	Description
KL6581_INPUT [▶ 39]	Process image of the inputs
KL6581_OUTPUT [▶ 39]	Process image of the outputs

KL6581

Data types	Description
AR_EnOceanWindow [▶ 40]	State of the window
E_ENOCEAN_Org [▶ 38]	EnOcean telegram type
E_KL6581_Err [▶ 38]	Error messages
STR_EnOceanSwitch [▶ 41]	State of the buttons
STR_KL6581 [▶ 41]	Internal structure
STR_Teach [▶ 42]	Data structure manufacturer ID, type and function
STR_Teach_In [▶ 42]	Data structure manufacturer ID, type and profile
STREnOceanTurnSwitch [▶ 42]	Position of the rotary switch at the room control unit

4.2.1 KL6021-0023

Hardware types

Data types	Description
ST_EnOceanInData [▶ 36]	Process image of the KL6021-0023 inputs
ST_EnOceanOutData [▶ 37]	Process image of the KL6021-0023 outputs

Data types	Description
E_EnOceanRotarySwitch [▶ 35]	Position of the rotary switch at the room control unit
E_EnOceanSensorType [▶ 35]	Sensor type
ST_EnOceanReceivedData [▶ 36]	Internal structure

4.2.1.1 Enums

4.2.1.1.1 E_EnOceanSensorType

Sensor type.

```
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.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.1.1.2 E_EnOceanRotarySwitch

E_EnOceanRotarySwitch describes the position of the rotary switch at the room control unit.

```
TYPE E_EnOceanRotarySwitch :
(
  eEnOceanRotarySwitchStep0 := 0,
  eEnOceanRotarySwitchStep1 := 1,
  eEnOceanRotarySwitchStep2 := 2,
  eEnOceanRotarySwitchStep3 := 3,
  eEnOceanRotarySwitchAuto  := 4
)
END_TYPE
```

eEnOceanRotarySwitchStep0: Switch in position “0”.

eEnOceanRotarySwitchStep1: Switch in position “1”.

eEnOceanRotarySwitchStep2: Switch in position “2”.

eEnOceanRotarySwitchStep3: Switch in position “3”.

eEnOceanRotarySwitchAuto: Switch in posture “Auto”.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.1.2 Structures

4.2.1.2.1 ST_EnOceanReceivedData

Internal structure.

This structure is used to link the block FB_EnOceanReceive() [▶ 14] with the receive blocks.

```
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: Data received.

nLength: Length.

eEnOceanSensorType: Sensor type (see E_EnOceanSensorType [▶ 35]).

nData: Data bytes.

nStatus : Status.

nTransmitterId: Transmitter ID.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.1.2.2 ST_EnOceanInData

Process image of the KL6021-0023 inputs.

Linked to the terminals in the System Manager.

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

nStatus: Status byte.

nData: 11 bytes for the input data.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.1.2.3 ST_EnOceanOutData

Process image of the KL6021-0023 outputs.

Linked to the terminals in the System Manager.

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

nCtrl: Control byte.

nData: 11 bytes for the output data.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2 KL6581

Hardware types

Data types	Description
KL6581_Input [▶ 39]	Process image of the KL6581 inputs
KL6581_Output [▶ 39]	Process image of the KL6581 outputs

Data types	Description
AR_EnOceanWindow [▶ 40]	State of the window
E_ENOCEAN_Org [▶ 38]	EnOcean telegram type
E_KL6581_Err [▶ 38]	Error messages
STR_EnOceanSwitch [▶ 41]	State of the buttons
STR_KL6581 [▶ 41]	Internal structure
STR_Teach [▶ 42]	Data structure manufacturer ID, type and function
STR_Teach_In [▶ 42]	Data structure manufacturer ID, type and profile
STREnOceanTurnSwitch [▶ 42]	Position of the rotary switch at the room control unit

4.2.2.1 Enums

4.2.2.1.1 E_ENOCEAN_ORG

EnOcean telegram type.

```
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 telegram.

STM_1BYTE_TELEGRAM: 1-byte telegram.

STM_4BYTE_TELEGRAM: 4-byte telegram.

CTM_TELEGRAM: CTM telegram.

MODEM_TELEGRAM: Modem telegram.

MODEM_ACK_TELEGRAM: Modem telegram with acknowledgement.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.1.2 E_KL6581_Err

Error messages.

```
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: No error at the block.

KL6581_WrongTerminal: Incorrect terminal connected.

KL6581_WatchdogError: Timeout during initialization of block "FB_KL6581".

KL6581_NoComWithKL6581: This message usually means that there is no connection to the terminal. Terminal linked to the variables in the System Manager? Terminal plugged in incorrectly? Everything corrected, everything translated and re-read into the System Manager?

KL6581_idx_number_not_OK: The input variable *nIdx* of block FB_KL6581() [▶ 22] is greater than 64.

KL6581_Switch_to_Stopp: The terminal has exited the data exchange with the KL6583. No EnOcean data was sent or received.

KL6581_not_ready: Internal message for the function blocks connected to the FB_KL6581().

KL6581_No_KL6853_Found: No KL6583 is connected to the KL6581, or the communication does not exist.

KL6581_TransmissionError: Data could not sent; check the address of the KL6583, or KL6583 not ready for operation.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2 Structures

4.2.2.2.1 KL6581_INPUT

Process image of the KL6581 inputs.

Linked to the terminal in the System Manager.

```
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: Data byte.

ORG: Data byte.

DB0: Data byte.

DB1: Data byte.

DB2: Data byte.

DB3: Data byte.

ID0: Data byte.

ID1: Data byte.

ID2: Data byte.

ID3: Data byte.

STATUS: Data byte.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.2 KL6581_Output

Process image of the KL6581 outputs.

Linked to the terminal in the System Manager.

```
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: Data byte.

ORG: Data byte.

DB0: Data byte.

DB1: Data byte.

DB2: Data byte.

DB3: Data byte.

ID0: Data byte.

ID1: Data byte.

ID2: Data byte.

ID3: Data byte.

STATUS: Data byte.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.3 ar_EnOceanWindow

This structure indicates the state of the window.

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

bUp: The window is tilted.

bOpen: The window is open.

bClose: The window is closed.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.4 str_EnOceanSwitch

State of the buttons.

```
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: Button 1 on.

bT1_OFF: Button 1 off.

bT2_ON: Button 2 on.

bT2_OFF: Button 2 off.

bT3_ON: Button 3 on.

bT3_OFF: Button 3 off.

bT4_ON: Button 4 on.

bT4_OFF: Button 4 off.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.5 str_KL6581

Internal structure.

This structure is used to link the block FB_KL6581() [▶ 22] with the send/receive blocks.

```
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    : PVOID;
  iErrorId  : E_KL6581_Err;
  by_STATE  : BYTE;
  bError    : BOOL;
  idx       : USINT;
END_STRUCT
END_TYPE
```

by_Status: Status.

by_Node: Node number of the KL6583 that has received the EnOcean telegram.

by_ORG: EnOcean telegram type.

ar_DB: Data bytes.

_Dummy: Placeholder, no other purpose.

dw_ID: Transmitter ID.

ptData: Pointer.

iErrorId: Describes the error type (see E_KL6581_Err [▶ 38]).

by_STATE: State.

bError: this output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

idx: Index.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.6 STR_Teach

Data structure manufacturer ID, type and function.

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

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.32	Tc2_EnOcean from v3.4.6.0

4.2.2.2.7 str_Teach_In

Data structure manufacturer ID, type and profile.

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

nManufacturerID: Manufacturer ID.

nTYPE: Type.

nProfile: Profile.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.2.2.2.8 STREnOceanTurnSwitch

STREnOceanTurnSwitch describes the position of the rotary switch at the room control unit.

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

bStageAuto: Switch in posture “Auto”.

bStage_0: Switch in position “0”.

bStage_1: Switch in position “1”.

bStage_2: Switch in position “2”.

bStage_3: Switch in position “3”.

Requirements

Development environment	required TC3 PLC library
TwinCAT v3.1.4020.14	Tc2_EnOcean from v3.3.5.0

4.3 Integration into TwinCAT

4.3.1 KL6581 with CX5120

This example describes how a simple PLC program for EnOcean can be written in TwinCAT and how it is linked with the hardware. The task is to receive four probe signals of an EnOcean wireless switch module.

Example: https://infosys.beckhoff.com/content/1033/tcplclib_tc2_enocean/Resources/6200373771/.zip

Hardware

Setting up the components

The following hardware is required:

- 1x CX5120 Embedded PC
- 1x KL6581 EnOcean master terminal
- 1x KL6583-0000 EnOcean transmitter and receiver
- 1x KL9010 end terminal

Set up the hardware and the EnOcean components as described in the associated documentation.

This example assumes that the ID of the wireless switch module is known.

Software

Creation of the PLC program

Create a new "TwinCAT XAE project" and a "Standard PLC project".
Add the library Tc2_EnOcean in the PLC project under "References".
Generate a global variable list with the name GVL_EnOcean and create the following variables:

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

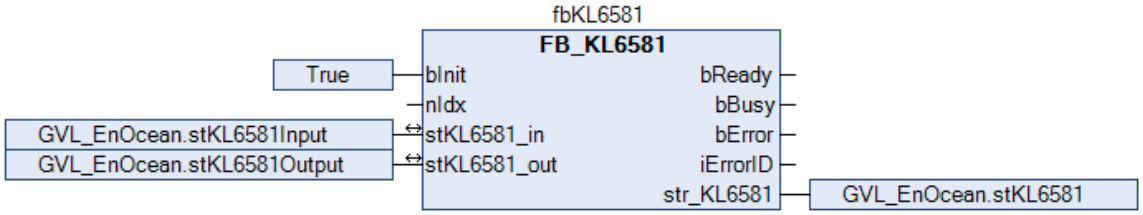
stKL6581Input: Input variable for the EnOcean terminal (see KL6581_INPUT [▶ 39]).

stKL6581Output: Output variable for the EnOcean terminal (see KL6581_Output [▶ 39]).

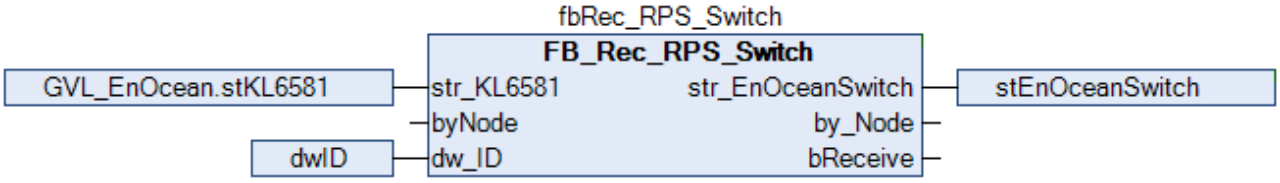
stKL6581: Required for communication with EnOcean (see str_KL6581 [▶ 41]).

All EnOcean function blocks must be called in the same task.

Create a MAIN program (CFC) in which the function blocks FB_KL6581 [▶ 22] and FB_Rec_RPS_Switch [▶ 25] are called. Ensure that the communication block is linked with the structures *stKL6581Input*, *stKL6581Output* and *stKL6581*.

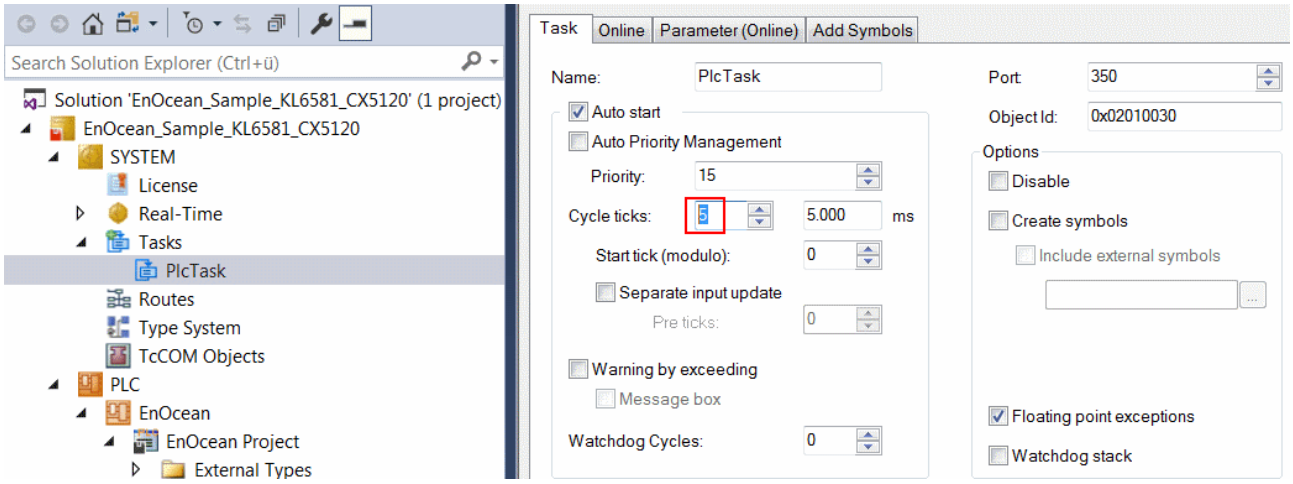


The input *dw_ID* of the receive block is linked with the local variable *dwId* (ID from wireless switch module) and *str_KL6581* with the global variable *stKL6581*.



Go to the task configuration and give the task a lower interval time.

Further conditions can be found in the description of the function block FB_KL6581 [▶ 22].



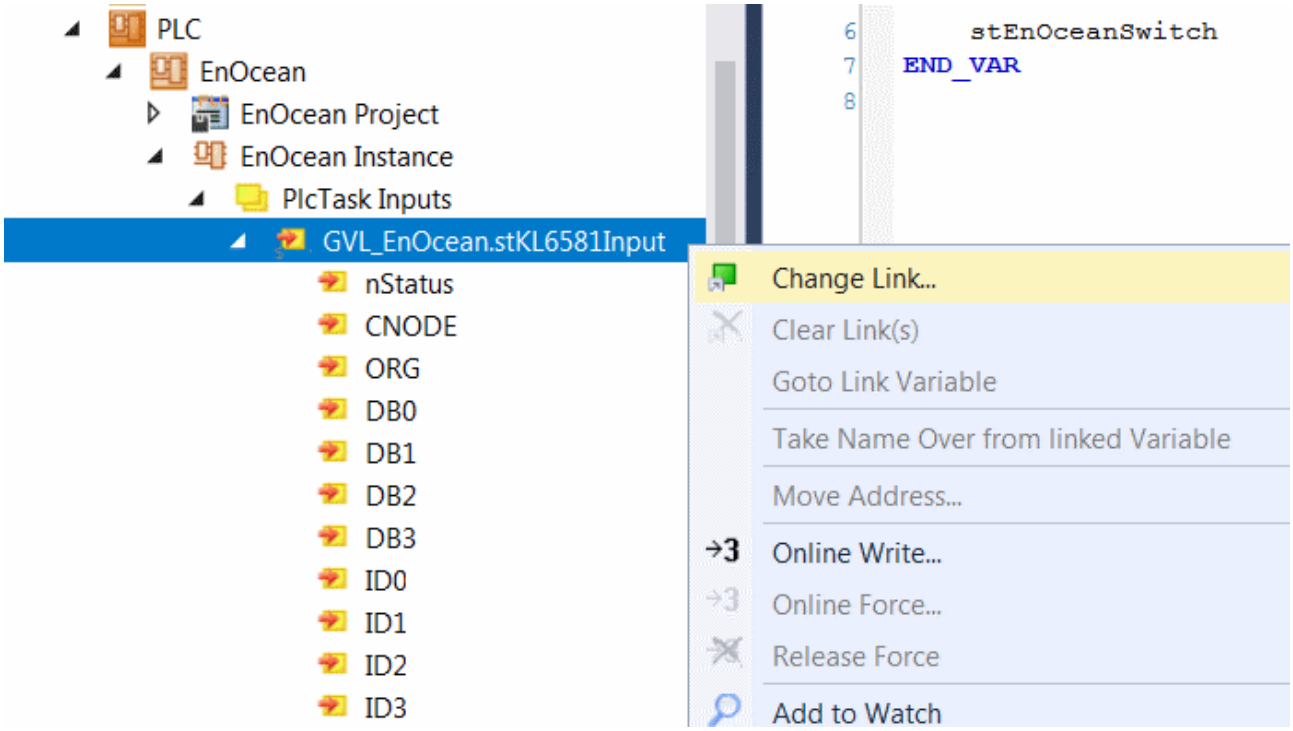
I/O configuration

Select the CX as target system and initiate a search for its hardware. In the project instance within the PLC section, you can see that the input and output variables are assigned to the task.

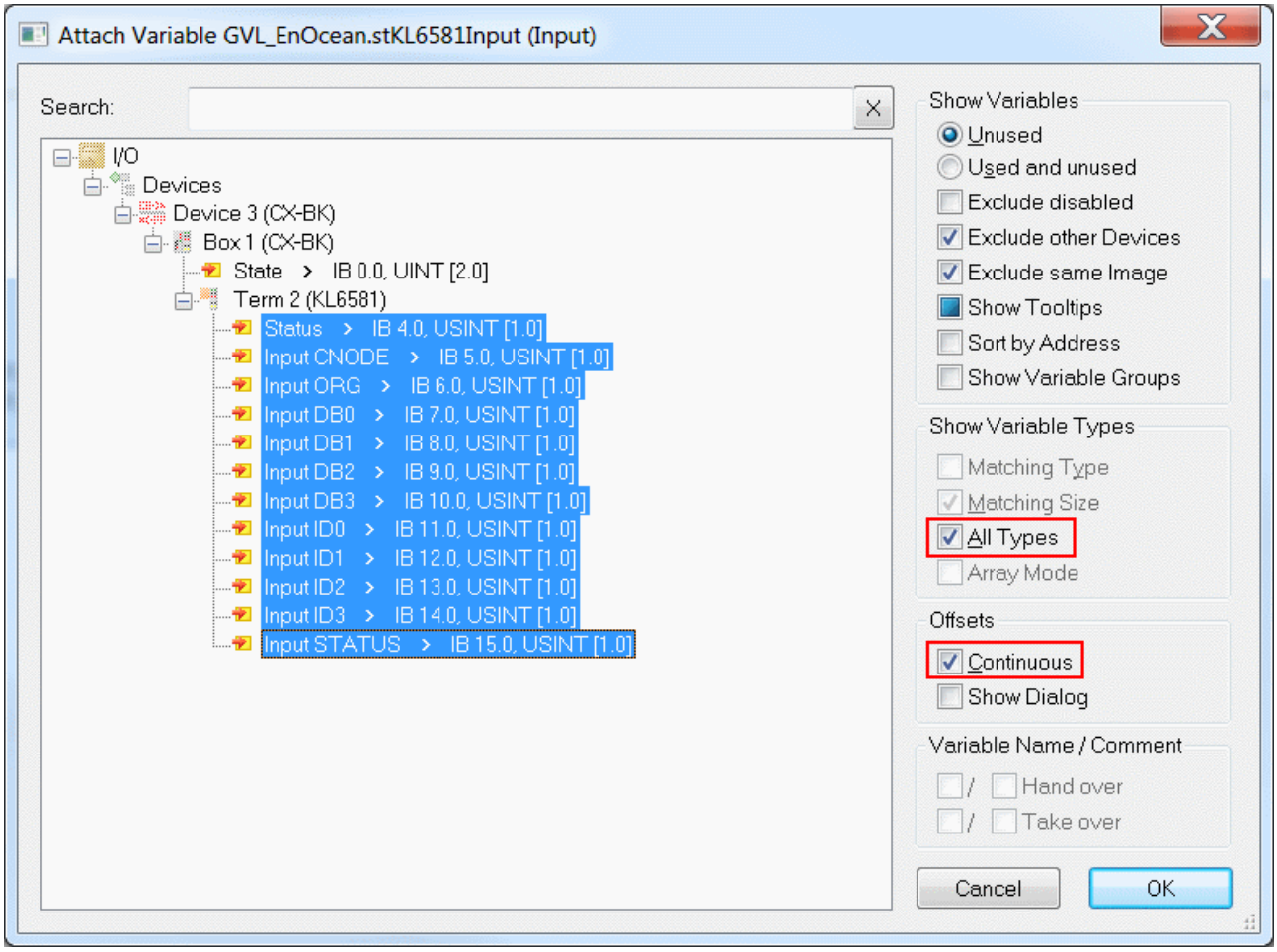
Now link the global variables with the inputs and outputs of the Bus Terminals.

The linking of the EnOcean variables is described in detail below.

Right-click the structure *stKL6581Input* and select “Change link”.



Under “I/O Configuration” select the terminal, select “All Types” and “Continuous”, then select “Status” to “InputStatus” with the left mouse button and the >SHIFT< key. Then press "OK".



You can now check the connection. To do this go onto the KL6581 and open it. All terminal data should now show an arrow. If that is the case, then proceed in exactly the same way with the outputs.

5 Appendix

5.1 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Download finder

Our download finder contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963-157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963-460
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963-0
e-mail:	info@beckhoff.com
web:	www.beckhoff.com

More Information:
www.beckhoff.com/te1000

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