

Application



EN Integration of EKS with PROFINET interface in BECKHOFF TwinCAT 3

Contents

1.	Abou	It this document	3
	1.1.	Version	3
	1.2.	Scope	3
	1.3.	Target group	3
	1.4.	Supplementary documents	3
	1.5.	Notice	3
2.	Com	ponents/modules used	4
	2.1.	EUCHNER	4
	2.2.	Others	4
	2.3.	Software	4
3.	Func	tional description	4
4.	Over	view of the communication data	5
	4.1.	Input	5
	4.2.	Output	5
5.	Insta	lling the GSD file	6
6.	Setti	ng the control system parameters	7
7.	Conf	iguring and setting the parameters of the EKS with PROFINET interfac	e 8
	7.1.	Configuring the PROFINET network	8
	7.2.	Configuring the EKS with PROFINET interface	9
	7.3.	Setting the EKS parameters	11
	7.4.	Assigning PROFINET device names to the EKS	12
8.	Using	g the BECKHOFF library	
	8.1.	Installation of the library	14
	8.2.	Calling the library and description of the block interface	16
		8.2.1. Calling the library	16
		8.2.3. Complete EKS data type call	
9.	Linki	ng the input and output areas in EKS	19
10.	Read	ling and writing Electronic-Kev data	
	10.1.	Transferring program to the PLC	
	10.2.	Reading content of the memory in the Electronic-Key using the block interface	21
	10.3.	Writing content of the memory in the Electronic-Key using the block interface	21
11.	Impo	rtant note – please observe carefully!	22

1. About this document

1.1. Version

Version	Date	Change/addition	Chapter
01-01/19	10.01.2019	Prepared	All

1.2. Scope

The purpose of this document is the integration and configuration of the EKS with PROFINET interface (from device version as per table) in BECKHOFF TwinCAT 3.

Order no.	Designation	Device version
106305	EKS-A-IIX-G01-ST02/03	V3.0.0
106306	EKS-A-IIXA-G01-ST02/03/04	V3.0.0
122352	EKS-A-AIX-G18	V1.X.X
122353	EKS-A-AIXA-G18	V1.X.X

1.3. Target group

Design engineers and installation planners for safety systems on machines, as well as setup and servicing staff possessing special expertise in handling safety components as well as expertise in the installation, setup, programming and diagnostics of programmable logic controllers (PLC) and bus systems.

1.4. Supplementary documents

The overall documentation for this application consists of the following documents:

Document title (document number)	Contents	
Manual (2516210)	Electronic-Key-System Manual EKS and EKS FSA with PROFINET IO interface	www
Possibly enclosed data sheets	Item-specific information about deviations or additions	

1.5. Notice

This application is based on the manual for the EKS with PROFINET interface. Please refer to the manual for the technical details and other information. In the rest of this document the EKS with PROFINET interface is referred to as the "EKS" for short.

2. Components/modules used

TIP!

2.1. EUCHNER

Description	Order number / item
EKS with PROFINET interface	106305 / EKS-A-IIX-G01-ST02/03

More information and downloads about the aforementioned EUCHNER products can be found at <u>www.euchner.com</u>. Simply enter the order number in the search box.

2.2. Others

 (\mathbf{i})

Description	Order number / item
CX9020-0110-M930	СХ9020-0110-М930

2.3. Software

Description	Version
Microsoft Visual Studio 2013 Shell (Integrated)	Version 12.0.21005.1 REL
Microsoft .NET Framework	Version 4.7.03056
TcMeasurement	1.0
TcProjectCompare	1.0.0.9
TcTargetBrowserPackage Extension	1.0
TcXaeDebuggerLiveWatch	1.0
TcXaeHelper	1.0
TcXaeModules	1.0
TwinCAT XAE Base	3.1.0.0
TwinCAT XAE EventLogger	1.0
TwinCAT XAE PLC	3.1.0.0

3. Functional description

EKS PROFINET devices are read/write systems with electronics for the inductive bidirectional interface to the transponder and the interface electronics.

The system is connected via the integrated PROFINET interface, which is designed as an RJ45 socket. A separate switch may be required for the PROFINET connection. The EKS does not have an integrated switch.

The current state of the Electronic-Key adapter is displayed using a 3-color LED.

The Electronic-Key is placed on the Electronic-Key adapter for operation. The power supply for the transponder and the data are transferred between the Electronic-Key adapter and the Electronic-Key without using any contacts.

4. Overview of the communication data

4.1. Input

PROFINET	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit O
Byte 0 (Status byte)	Job in progress	Job finished	-	-	-	-	Electronic-Key detected	Device ready for operation
PROFINET		Description				Function		
Byte 1								
· .	1			May 124 butas usar	data from the Elect	rania Kay plua 2 but	as reserve lf fourer	but a of data wara
		Receive data		selected during confi	guration, these are	filled with 0 hex.	es reserve. Il lewer	Dytes of uata were
•				÷	- ·			
Byte 127								

4.2. Output

PROFINET	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit O
Byte 0 (command byte)	-	-	-	-	-	-	-	Write Electron- ic-Key
PROFINET		Description				Function		
Byte 1		Start address		Defines first byte command byte. S	in the memory in t tart address of use	the Electronic-Key t r data: byte no. 0, 4	hat is written on se 4, 8 112.	etting bit no. 0 in the
Byte 2		Number of bytes		Defines the number 0 in the command	er of bytes in the m I byte. Number of b	emory in the Electro oytes of user data: 4	onic-Key that are wri 1, 8, 12 … 116 byte	tten on setting bit no. es.
Byte 3		Not used						
Byte 4								
		Transmit data		If bit no. 0 in the co starting from the s	ommand byte is set start address define	to 1, the content of t ed.	these bytes is writter	n to the Electronic-Key
Byte 119								
Byte 120								
:		Not used						
Byte 127								

í	TIP!
-	You will find further information in the manual.

NOTE!
On the Electronic-Key read/write with 116 bytes freely programmable, the memory is organized in 4-byte blocks. This means the start address for writing must be given in the range byte number 0 to byte number 112, always in 4-byte steps (byte number 0, 4, 8 112). Also a multiple of 4-byte sized blocks must always be written (4, 8, 12 116 bytes).
During reading it is possible to access the memory byte-by-byte without the above-mentioned restric- tion for writing.

EN

5. Installing the GSD file

You will require the corresponding GSD file in the GSDML format to integrate the EKS into TwinCAT 3, depending on the design of the EKS:

Design	Related GSD file
compact (Order No. 106305/106306)	GSDML-V2.31-Euchner-EKS_3.x.x_109539-YYYYMMDD.xml
modular (Order No. 122352/122353)	GSDML-V2.31-EUCHNER-EKS_PN_modular_126145-YYYYMMDD.xml

You will find the GSD files in the download area at <u>www.euchner.com</u>. Always use the latest GSD file.

Please proceed as follows to integrate the GSD file in TwinCAT 3:

Open the path as shown in Figure 1 and then add the unpacked GSDML and bitmap file.

← → × ↑ 📜 > This P	C > Local Disk (C:) > TwinCAT > 3.1 > Config > Io > Profinet 1.	
	Name	Date modified
Y Quick access	GSDML-0135-0201-EKS_PN.bmp	06/11/2018 11:33
la OneDrive	GSDML-V2.31-Euchner-EKS_3.x.x_109539-20180628.xml	06/11/2018 11:33

Figure 1: Adding GSD file

6. Setting the control system parameters

Specify the cycle time for the PlcTask. Use the value 2 for this purpose.



Figure 2: Setting the control system parameters

7. Configuring and setting the parameters of the EKS with PROFINET interface 7.1. Configuring the PROFINET network

Add the PROFINET network as follows:

1. In Solution Explorer click I/O, right-click Devices and choose Scan.



Figure 3: Adding PROFINET network



NOTE!

To undertake scans the TwinCAT must be in the Config Mode.

- 2. Select the PROFINET controller and accept using OK.
- 4 new I/O devices found

Device 1 (Profinet Controller CCAT (RT))	ОК
Device 2 (EtherCAT) Device 3 (EtherCAT Automation Protocol) [FEC1] Device 5 (NOV-DP-RAM)	Cancel
	Select All
	Unselect All

Figure 4: Selection of PROFINET controller

×

3. You are then prompted as to whether a search is to be made for additional boxes (devices). Please answer this prompt with No because it cannot be ensured the correct EKS will be configured.



Figure 5: Rejecting search for devices

7.2. Configuring the EKS with PROFINET interface

4. Right-click the PROFINET controller and then choose Add New Item...



Figure 6: Adding a device

EN

5. Select the corresponding GSDML file.

Insert Box		
Type:	 Beckhoff Automation GmbH Euchner GmbH + Co. KG ↓ 0 Euchner EKS (GSDML-V2.31-Euchner-EKS_3××_109539-20180628.xm Miscellaneous PROFIdrive MC (DPV2 / PNIO) 	Ok Cancel Multiple:
Name:	Box 1	

Figure 7: Selection of the GSDML file

6. Select the corresponding EKS.

Module DAP	3	ОК
<u>Т</u> уре:	EKS-A-IIXA-G01-ST02/03/04 EKS-A-IIX-G01-ST02/03	

Figure 8: Selection of the EKS

7.3. Setting the EKS parameters

The following PROFINET parameters must be set:

- > Device name (factory setting from GSD file): [eks-pn].
- IP address: fixed

Solution Explorer 🔹 👎 🗙	EKS_Library_Integration 😕 🗙 Library Manager MAIN Output
○ ○ ☆ 'o · @ <i>¥</i> -	General Device Diagnosis Features ADS Shared Device
Search Solution Explorer (Ctrl+;)	Adapter Properties Stationname
SYSTEM MOTION PLC SAFETY	VendorldDeviceIdHW VersionSW Version0x01350x02020.00T 0.00
GHETT GHE C++ ▲ ☑ I/O ▲ □ Devices	IP configuration IP address 192 . 168 . 0 . 53
 Device 1 (Profinet Controller CCAT (RT)) Image Inputs Outputs 	Subnet 255 255 0 Gateway 192 168 0 52 Refresh GSDML
 Outputs State of the second seco	Instance Properties
Mappings	ID FrameId 0x8000
	MaxLengthIn MaxLengthOut ActLengthIn ActLengthOut 1440 Byte 1440 Byte 133 Byte 133 Byte

Figure 9: PROFINET parameters

IO cycle real time settings

These values are already set to the recommended default values.



Figure 10: PROFINET real time settings



7.4. Assigning PROFINET device names to the EKS

There are two ways of assigning a device name to the EKS. Either using the EKS web browser (see manual) or using TwinCAT. In the following we show name assignment using TwinCAT.

1. To assign the name to the EKS using TwinCAT, please right-click the PROFINET controller and then choose Scan.



Figure 11: Searching for devices online

2. Select the EKS from the list. Enter the device name in Stationname and accept using Set Stationname.

	00:TA:5C:05:D	132.100.1.1	200.200.200. U	0.0.0.0	
					Add Devices
<				>	
a transmitte					
eks-ph					Set Stationname
P configuratio	n	1 1			Set Stationname
P configuration	n 192 . 168 .	1 , 1	DHCP en	able	Set Stationname Set IP configuration Start Signal

Figure 12: Assigning device name



8. Using the BECKHOFF library

The library is intended to assist you during programming. The library contains pre-prepared data that you can then use. Open the page with the EKS applications in the download area at <u>www.euchner.com</u> and download the library for the EKS.

8.1. Installation of the library

- 1. Click the *PLC* tab and open the *Library Repository*.
- 2. Install the library and select the path where you have saved the library.

🕽 Library Rep	pository		
Location:	System	~	Edit Locations
	(C:\TwinCAT\3.1\Components\Plc\Managed Libraries)		
Installed libra	ries:		Install
Company:	(All companies)	~	Uninstal
₽ ● App	lication	^	Grintotal
Built Built Con Con Con Data	dingAutomation imunication troller aAccess ern h surement ion	~	Find
<		>	Details
Group b	y category		Dependencies
Library Pr	ofiles		Close

Figure 13: Installing library



3. Next you must add the library to the project. In Solution Explorer, right-click References and then choose Add library.



- Figure 14: Adding library to the project
- 4. Select the library prepared by EUCHNER.

dd Library	
Enter a string for a fulltext search in all libraries	
Library + Application + BuildingAutomation + Communication + Controller + DataAccess + DataAcc	Company
Advanced	OK Cancel

Figure 15: Selection of the library

ΕN

8.2. Calling the library and description of the block interface

8.2.1. Calling the library

1. To be able to use the library, the blocks from the library must be called in the main program (MAIN). For this purpose, open the block and select the programming section. You can select the blocks with the aid of the *Input Assistant* that can be opened using the right mouse button or the F2 key.

	• ×			_			PROGR	AM MAIN
^	Scope	Name A	ddress	Data type	e Ini	tialization	Comment	Attributes
1						1	A V	
	55	Add to scop Browse Call	e Tree					
		Find All Refe	erences	Shift	+F12			
	2	Input Assista	ant	F2	R			
		Auto Declar	e					
	h:	Run To Curse	or	Ctrl+	F10			
		Display Mod	le		•			
	ж	Cut		Ctrl+	х			
	ď	Сору		Ctrl+	С			
	பி	Paste		Ctrl+	v			
	×	Delete		Del				
	×.	Select All		Ctrl+	A			

Figure 16: Opening the *Input Assistant*

2. You will find the library on the *Categories* tab. Select *Function Blocks*. You will find the EUCHNER library in the window on the right. You can select two *function blocks* from the library. *EKS_Read* is required to read the Electronic-Key data and *EKS_Write* to write data to the Electronic-Key.

ext search Categories					
Variables Module Calls	Name EKS_Library_Inter	gration	Type Application	Origin	
Instance Calls Function Blocks	Ibrary_EUCHNER DOUS	_EKS_AP000240_	01_19 Library	Library_EUCHNE	
Keywords	IF EKS_Rea	1	FUNCTION_BLOCK	Library_EUCHNE	
Conversion Operators	EKS_Writ	e	FUNCTION_BLOCK	Library_EUCHNE	
	# {} Tc2_Standard		Library	Tc2_Standard, 3	
	# {} Tc2_System		Library	Tc2_System, 3	
			Library	Tc3_Module, 3.3	
Structured view			⊡ Insert <u>w</u> ith arg	uments	Insert with namespace pre
Structured view ocumentation: FUNCTION_BLOCK EKS_Rea	ad		⊡ Insert <u>w</u> ith arg	uments	Insert with namespace pre
] Structured view Sygumentation: -UNCTION_BLOCK EKS_Rea Device_ready_for_operation	ad n BOOL	VAR_OUTPUT	☑ Insert with arg	uments he device signals tl	Insert with namespace pre
] structured view ogumentation: =UNCTION_BLOCK EKS_Ree Device_ready_for_operation Electronic_Key_detected	ad BOOL BOOL	VAR_OUTPUT VAR_OUTPUT	∑ Insert with arg After successful configuration the The detection of a valid Electro	uments he device signals th nic-Key is signaled	Insert with namespace pre nat it is ready via bit no. 0. using bit no. 1.
] structured view sgumentation: FUNCTION_BLOCK EKS_Ree Device_ready_for_operation Electronic_Key_detected Job_finished	ad BOOL BOOL BOOL	VAR_OUTPUT VAR_OUTPUT VAR_OUTPUT	✓ Insert with arg After successful configuration th The detection of a valid Electro Bit no. 6 indicates that a write i	uments he device signals th nic-Key is signaled process has been s	Insert with gamespace pre hat it is ready via bit no. 0. using bit no. 1. uccessfully finished.
] structured view scumentation: UNCTION_BLOCK EKS_Rea Device_ready_for_operation Electronic_Key_detected Job_finished Job_in_progress	ad BOOL BOOL BOOL BOOL	VAR_OUTPUT VAR_OUTPUT VAR_OUTPUT VAR_OUTPUT	∑ Insert with arg After successful configuration th The detection of a valid Electro Bit no. 6 indicates that a write j Bit no. 7 indicates that a write j	uments he device signals th nic-Key is signaled process has been s process is currently	Insert with gamespace pro

Figure 17: Selecting the blocks (function blocks)

3. The blocks must be instanced. In our example we assign the name *EKS_Read_01* for the data type *EKS_Read*. Then you can repeat this process (step 1 to 3) for the data type *EKS_Write*.

Scope:	<u>N</u> ame:	Type:
/AR	V EKS_Read_01	EKS_Read >
<u>D</u> bject:	Initialization:	Address:
MAIN [EKS_Library_Integration]	×	
Jags:	Co <u>m</u> ment:	
		1
PERSISTENT		
<u>PERSISTENT</u>		

Figure 18: Declaring the data type

4. The program must now be built. In this way the input and output variables for the project are generated; these variables must be linked later to the read and write submodules. Click the *BUILD* tab and select *Build Solution* or use the shortcut: Ctrl+Shift+B



Figure 19: Automatically generated input and output variables

ΞN

8.2.2. Description of the block interfaces

Parameter	Data type	Description
Device_ready_for_operation	BOOL	After completion of the configuration of the device, ready for operation is signaled.
Electronic_Key_detected	BOOL	Indication that an Electronic-Key is detected.
Job_finished	BOOL	Provides feedback on the successful completion of a write process.
Job_in_progress	BOOL	Indicates that a write process is in progress.
Receive_data	ARRAY [0123] OF BYTE	Electronic-Key data

Table 1: Read block interface

Parameter	Data type	Description				
Write_Electronic_Key	BOOL	Set this bit to issue the write command.				
Start_address	BYTE	Defines the first byte to be written in the memory in the Electronic-Key.				
Number_of_bytes	BYTE	Defines the number of bytes of data to be written in the memory in the Electronic-Key.				
Transmit_data	ARRAY [0115] OF BYTE	The content of this byte is written to the Electronic-Key.				

Table 2: Write block interface

8.2.3. Complete EKS data type call

2	\$ 3	X	PROGRAM MAIN						
	1 2	Scope VAR VAR	Name EKS_Read_01 EKS_Write_01	Address	Data type EKS_Read EKS_Write	Initialization	Comment	Attributes	
¢ 1	1	EKS_Rea	d_01(A V	_	
1	1 2 3	EKS_Rea Dev	d_01(ice_ready_for_	operation	=> ,		▲ ▼	_	
41	1 2 3 4	EKS_Rea Dev Ele Job	d_01(ice_ready_for_ ctronic_Key_de finished=> ,	operation	=> ,		×	_	
•	1 2 3 4 5	EKS_Rea Dev Ele Job Job	d_01(ice_ready_for_ ctronic_Key_de _finished=> , _ in_progress=>	operation tected=>	=> ,		A V.		
•	1 2 3 4 5 6	EKS_Rea Dev Ele Job Job Rec	d_01(ice_ready_for_ ctronic_Key_de _finished=>, _in_progress=> eive_data=>);	operation tected=>	=> ,		▲ ▼		
4	1 2 3 4 5 6 7	EKS_Rea Dev Ele Job Rec	d_01(ice_ready_for_ ctronic_Key_de _finished=> , _in_progress=> eive_data=>);	operation tected=>			<u>↓ ▼</u>		
	1 2 3 4 5 6 7 8	EKS_Rea Dev Ele Job Rec EKS_Wri	d_01(ice_ready_for_ ctronic_Key_de finished=> , in_progress=> eive_data=>); te_01(operation tected=>	=> ,		A V.		
4	1 2 3 4 5 6 7 8 9	EKS_Rea Dev Ele Job Rec EKS_Wri Wri	d_01(ice_ready_for_ ctronic_Key_de _finished=> , _in_progress=> eive_data=>); te_01(te_Electronic_	operation tected=> ., Key:= ,	=> ,		A V.		
•	1 2 3 4 5 6 7 8 9 10	EKS_Rea Dev Ele Job Rec EKS_Wri Sta	d_01(ice_ready_for_ ctronic_Key_de _finished=> , _in_progress=> eive_data=>); te_01(te_Electronic_ rt_address:= ,	operation tected=> ., Key:= ,	=> ,		A V.		

Figure 20: Complete call in PROGRAM MAIN

9. Linking the input and output areas in EKS

The read (Inputs) and write (Outputs) modules must be linked using the variables generated in chapter 8.2.1.

1. In Solution Explorer, open the tree as shown below. Double-click the read (Inputs) module to open the properties.



Figure 21: EKS read (Inputs) module in the hardware configuration

2. In the properties for the read (Inputs) module, click the Linked to... button .

Solution Explorer 🔹 🖣 🗙	EKS_Library_Integr	ation 🤊 🗙 MAIN Output
○ ○ 습 'o - 司 / ≠ <u>-</u>	Variable Flags	
Search Solution Explorer (Ctrl+;)	Name:	Inputs
Solution 'EKS_Library_Integration' (1 project)	Туре:	ARRAY [0127] OF BYTE
SYSTEM	Group:	Inputs Size: 128.0
MOTION PLC	Address:	4 (0x4) User ID: 0
SAFETY	Linked to	
 Key C++ ▲ 2 I/O 	Comment	^
 Devices Device 1 (Profinet Controller CCAT (RT)) Image Inputs 		
Cutputs Gets-pn Getsuts	ADS Info:	Port 11, IGrp: 0x3040010, IOffs: 0x80000004, Len: 128
Guiptits Guiptits API Frrm 1 (DAP Module) Frrm 2 (Read: 128 bytes)	Full Name:	TIID^Device 1 (Profinet Controller CCAT (RT))^eks-pn^API^Term 2 (Read: 128 by
 Bubterm 1 (Read: 128 byte: Inputs 	5)	
👂 🔁 Inputs		
Uutputs	Error List	
Term 3 (Write: 128 bytes)	📕 🍸 🔻 🐼 0 Erro	rs 👔 0 Warnings 🚺 28 Messages 🛛 Clear

Figure 22: Properties of the read (Inputs) module

3. Select the input variable and click OK. To link the output variables, undertake steps 1-3 again for the write (*Outputs*) module that is to be found in *Solution Explorer* in *Term 3 (Write: 128 bytes)*.

Attach variable inputs (input)	
Search: X PLC EKS_Library_Integration KKS_Read_01 Read_data > IB 879720.0. ARRAY [0.127] OF BYTE [1280]	Show Variables

Figure 23: Adding the input variables

10. Reading and writing Electronic-Key data

10.1. Transferring program to the PLC

Transfer the program to the control system by clicking Activate configuration 1

10.2. Reading content of the memory in the Electronic-Key using the block interface

In the following an extract of the status bytes and part of the data in the memory in the Electronic-Key are shown using the block interface. Go online by clicking *Login*. To read the Electronic-Key data, it is only necessary to place the Electronic-Key in the Electronic-Key adapter. The Electronic-Key data are transferred cyclically to the PLC.

MAIN [Online] 😐 🗙 Output						
EKS_Library_Integration.EKS_Library_In	tegration.MAIN			11		
Expression	Туре	Value	Prepared value	Address	Comment	-
EKS_Read_01	EKS_Read					
Device_ready_for_operation	BOOL	TRUE			After successful configuration the d	
Electronic_Key_detected	BOOL	TRUE			The detection oflid Electronic-K	
Sob_finished	BOOL	FALSE			Bit no. 6 indicatesat a write proce	
Job_in_progress	BOOL	FALSE			Bit no. 7 indicatesat a write proce	
🖃 🍫 Receive_data	ARRAY [0123]				Only key data.	
Receive_data[0]	BYTE	16#00				
Receive_data[1]	BYTE	16#01				
Receive_data[2]	BYTE	16#02				
Receive_data[3]	BYTE	16#03				
Receive_data[4]	BYTE	16#04				
Receive_data[5]	BYTE	16#05				
Receive_data[6]	BYTE	16#06				
Receive_data[7]	BYTE	16#07				-

Figure 24: Reading Electronic-Key data example

10.3. Writing content of the memory in the Electronic-Key using the block interface

The same block interface has been prepared such that data can also be written to the Electronic-Key. For this purpose the start address and the number of bytes must be defined (cf. Chapter 4.2). In this example the first 8 bytes of the memory in the Electronic-Key are written. In the *Receive_data* array complete the data in the *Prepared value* column, set the *Write_Electronic_Key* bit to *TRUE* and transfer everything by clicking the *Force value* button. Then the *Write_Electro* Key bit must be reset to the value *FALSE*.

MAIN [Online] 🕆 🗙 Output						
EKS_Library_Integration.EKS_Library_	_Integration.MAIN					
Expression	Туре	Value	Prepared value	Address	Comment	*
EKS_Write_01	EKS_Write					
* Write_Electronic_Key	BOOL	TRUE			Bit to write the key.	
* Start_address	BYTE	16#00			Defines first byte the memory of	
Number_of_bytes	BYTE	16#08			Defines the number of bytes in the	
🖃 🍫 Transmit_data	ARRAY [0115]				The content of these bytes is writte	
✤ Transmit_data[0]	BYTE	16#00				
M Transmit_data[1]	BYTE	16#01				
* Transmit_data[2]	BYTE	16#02				
Transmit_data[3]	BYTE	16#03				
牧 Transmit_data[4]	BYTE	16#04				
✤ Transmit_data[5]	BYTE	16#05				
* Transmit_data[6]	BYTE	16#06				
🏷 Transmit_data[7]	BYTE	16#07				w



11. Important note - please observe carefully!

This document is intended for a design engineer who possesses the requisite knowledge in safety engineering and knows the applicable standards, e.g. through training for qualification as a safety engineer. Only with the appropriate qualification is it possible to integrate the example provided into a complete safety chain.

The example represents only part of a complete safety chain and does not fulfill any safety function on its own. In order to fulfill a safety function, the energy switch-off function for the danger zone and the software within the safety evaluation must also be considered, for example.

The applications provided are only examples for solving certain safety tasks for protecting safety doors. The examples cannot be comprehensive due to the application-dependent and individual protection goals within a machine/installation.

If questions concerning this example remain open, please contact us directly.

According to the Machinery Directive 2006/42/EC, the design engineer of a machine or installation has the obligation to perform a risk assessment and take measures to reduce the risk. While doing this, the engineer must comply with the applicable national and international safety standards. Standards generally represent the current state-of-the-art. Therefore, the design engineer should continuously inform himself about changes in the standards and adapt his considerations to them. Relevant standards include EN ISO 13849 and EN 62061. This application must be regarded only as assistance for the considerations about safety measures.

The design engineer of a machine/installation has the obligation to assess the safety technology him/herself. The examples must not be used for an assessment, because only a small excerpt of a complete safety function was considered in terms of safety engineering here.

In order to be able to use the safety switch applications correctly on safety doors, it is indispensable to observe the standards EN ISO 13849-1, EN ISO 14119 and all relevant C-standards for the respective machine type. Under no circumstances does this document replace the engineer's own risk assessment, and it cannot serve as the basis for a fault assessment.

In particular in relation to a fault exclusion, it must be noted that a fault can only be excluded by the machine's or installation's design engineer and this action requires justification. A general fault exclusion is not possible. More information about fault exclusion can be found in EN ISO 13849-2.

Changes to products or within assemblies from third-party suppliers used in this example can lead to the function no longer being ensured or the safety assessment having to be adapted. In any event, the information in the operating instructions on the part of EUCHNER, as well as on the part of third-party suppliers, must be used as the basis before this application is integrated into an overall safety function. If contradictions should arise between the operating instructions and this document, please contact us directly.

Use of brand names and company names

All brand names and company names stated are the property of the related manufacturer. They are used only for the clear identification of compatible peripheral devices and operating environments in relation to our products.

EN



Euchner GmbH + Co. KG Kohlhammerstraße 16 D-70771 Leinfelden-Echterdingen info@euchner.de www.euchner.com

Edition: AP000240-01-01/19 Title: Application EKS Integration of EKS with PROFINET interface in BECKHOFF TwinCAT 3

Copyright: © EUCHNER GmbH + Co. KG, 01/2019

Subject to technical modifications; no responsibility is accepted for the accuracy of this information.