EKS Profibus on Siemens S7-300 – reading in EKS Electronic-Keys

Contents

Components/modules used

EUCHNER

Description	Order no./item designation
EKS Profibus	084800 / EKS-A-IDX-G01-ST09/03
EKS Electronic-Key	077859 / EKS-A-K1RDWT32-EU 084735 / EKS-A-K1BKWT32-EU 091045 / EKS-A-K1BUWT32-EU 094839 / EKS-A-K1GNWT32-EU 094840 / EKS-A-K1YEWT32-EU

Tip: More information and downloads about the aforementioned EUCHNER products can be found at www.EUCHNER.de. Simply enter the order number in the search box.

Others

Description	Item
S7-300, CPU 315F-2 PN/DP	6ES7315-2FJ14-0AB0

Functional description

General

The EKS is connected to a Siemens S7-300 PLC via the Profibus. All data corresponding to the data structure below should be read out.

Example of an Electronic-Key structure

The data on the Electronic-Key are structured as follows:

Byte no.	Description	Type	Length	Explanation
103 – 104	KEYCRC	CRC	2 bytes	Checksum over a certain part of the Electronic-Key as copy protection. Refer to the EKM manual for details about the CRC.
105 – 112	Expiry date	Date	8 bytes	Electronic-Key expiry date.
113 – 114	Authorization level	Word	2 bytes	Authorization level for access to the machine.
115	Department	Byte	1 byte	Number describing a limited quantity of machines or installations.
116 – 123	KeyID	KeyID	8 bytes	The KeyID is a number that is permanently pre-programmed on the Electronic-Key by EUCHNER. This number is different for each Electronic-Key. This number can be used to identify workers.

The structure corresponds to application example AP000169-2…

Setting the EKS Electronic-Key adapter

Profibus address

The device must be set to address 75. Address 75 is 1001011 in binary notation. DIP switches 1 to 7 are set accordingly (DIP switch 1 is the least significant bit).

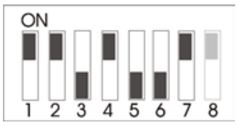

Figure 1

Write-protection setting

The device is configured only for reading. Correspondingly, DIP switch 8 is set to ON.

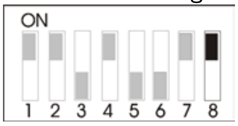

Figure 2

Configuration in the control system

Hardware

Simatic Manager version 5.5+SP1 is used for configuration. To perform parameter assignment for EKS on the Profibus, drag the object “EKS-A-IDX-G01-ST09/03” to the Profibus, followed by the “Read/Write: 32 Byte” module to the first slot. The address range can remain set at 256 to 287. The size of 32 bytes is selected because a total of 21 bytes of user data, including the KeyID, are to be retrieved. An adequately large input range must be reserved in the control system for this purpose.
When a new Electronic-Key is inserted, the data are always read automatically from byte 0. As the user data are located at the end of the Electronic-Key instead of at the start in this example, the actual user data are loaded in a read routine in the sequential program. The KeyID is retrieved at the same time.

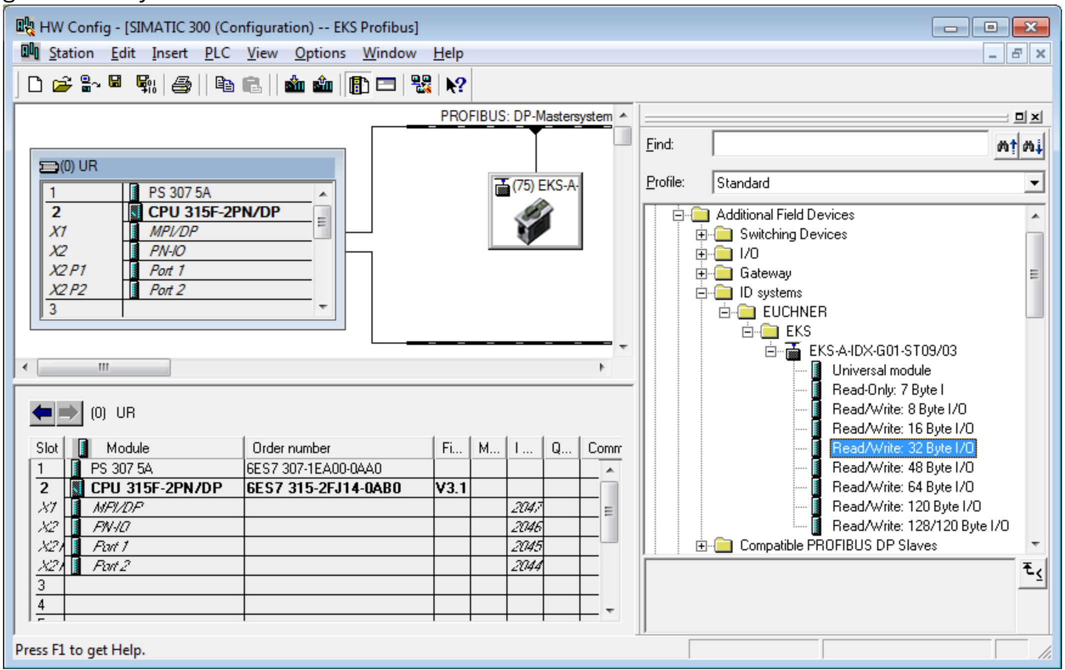

Figure 3

Set address 75 in the properties for the Profinet interface of the DP slave, matching the settings on the DIP switches. You set the subnet parameters according to your bus system.

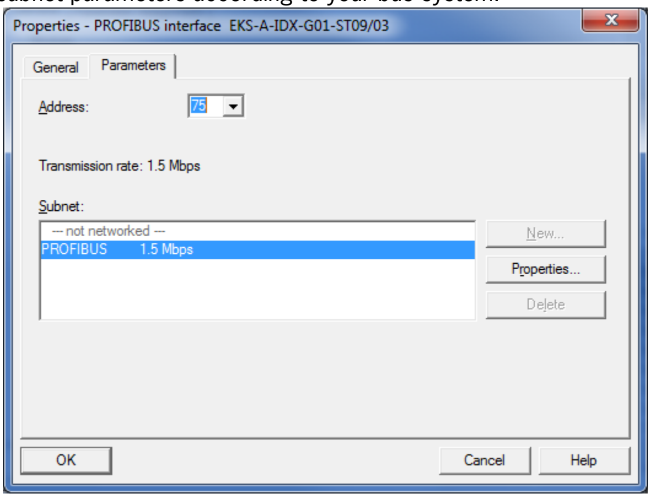

Figure 3

EUCHNER
More than safety.

Programming in the control system

Global data blocks

Data blocks are created for saving the transmitted and received data for the EKS.
The data are created in a structured manner in data block DB1 for reading, with all data items longer than one byte being created as individual bytes to circumvent the even-numbered alignment in the control system. The data block must be the same length as the input range of the EKS, otherwise the system function for reading will not work.

DB1, ReadBufferEKS

Address	Name	Type	Initial val	Comment
0.0		STRUCT		
+0.0	ReadEKSStatus	BYTE	B#16#0	Statusbyte from EKS
+1.0	ReadKeyCount	BYTE	B#16#0	Counter for keys
+2.0	ReadStartAddress	BYTE	B#16#0	First byte
+3.0	ReadNumberBytes	BYTE	B#16#0	Number of bytes read
+4.0	ReadCRC_00	BYTE	B#16#0	CRC Byte 0
+5.0	ReadCRC_01	BYTE	B#16#0	CRC Byte 1
+6.0	ReadDate_00	BYTE	B#16#0	Date Byte 0
+7.0	ReadDate_01	BYTE	B#16#0	Date Byte 1
+8.0	ReadDate_02	BYTE	B#16#0	Date Byte 2
+9.0	ReadDate_03	BYTE	B#16#0	Date Byte 3
+10.0	ReadDate_04	BYTE	B#16#0	Date Byte 4
+11.0	ReadDate_05	BYTE	B#16#0	Date Byte 5
+12.0	ReadDate_06	BYTE	B#16#0	Date Byte 6
+13.0	ReadDate_07	BYTE	B#16#0	Date Byte 7
+14.0	ReadAuthorization_00	BYTE	B#16#0	Access Level Byte 0
+15.0	ReadAuthorization_01	BYTE	B#16#0	Access Level Byte 1
+16.0	ReadDepartment	BYTE	B#16#0	Department
+17.0	ReadKeyID_00	BYTE	B#16#0	KeyID Byte 0
+18.0	ReadKeyID_01	BYTE	B#16#0	KeyID Byte 1
+19.0	ReadKeyID_02	BYTE	B#16#0	KeyID Byte 2
+20.0	ReadKeyID_03	BYTE	B#16#0	KeyID Byte 3
+21.0	ReadKeyID_04	BYTE	B#16#0	KeyID Byte 4
+22.0	ReadKeyID_05	BYTE	B#16#0	KeyID Byte 5
+23.0	ReadKeyID_06	BYTE	B#16#0	KeyID Byte 6
+24.0	ReadKeyID_07	BYTE	B#16#0	KeyID Byte 7
+26.0	Buffer	ARRAY[0..5]		NC for filling up to 32 bytes
*1.0		BYTE		
=32.0		END_STRUCT		

Figure 4

As a command has to be sent to the EKS so that the data from byte No. 103 and the KeyID can be read, the data required for this purpose are located at the start of data block DB2. The same data range of the EKS Electronic-Key is always read. The data block is suitably pre-filled during initialization. These are the bytes WriteCommand (64(dec.)), WriteStartAddress (103(dec.)) and WriteNumberBytes (21(dec.)). The data block must be the same length as the output range of the EKS, otherwise the system function for writing will not work.

DB2, WriteBufferEKS

Address	Name	Type	Initial val	Comment
0.0		STRUCT		
+0.0	WriteCommand	BYTE	B#16#40	Transmit buffer
+1.0	WriteStartAddress	BYTE	B#16#67	Start address on kye (103)
+2.0	WriteNumberBytes	BYTE	B#16#15	number of bytes read from key
+4.0	Buffer	ARRAY[0..27]		Buffer for rest of telegram
*1.0		BYTE		
=32.0		END_STRUCT		

Figure 5

DB10, instance module for FB1
As the function module FB1 operates with static variables, a DB must be used as an instance module. In the example, DB10 is created for this purpose.

STL program for retrieving the Electronic-Key content

The reading program is programmed in FB1 in this example. The program reads only when an Electronic-Key is inserted and new data are ready. An Electronic-Key that has been read in once will not be read in again. The data from byte 103 (KeyCRC), including the KeyID, are read and are provided in data block DB1 from byte 4 for further processing. 21 bytes of user data in total are retrieved from the EKS Electronic-Key.
The status bytes of the EKS are saved in bytes 0 to 3 of DB1.

Description of the interface

Input data
None.

Output data
Error message, new Electronic-Key and status of the DP slave.

Input/output data
None.

Static data
The counter value of the EKS is created statically. This value is compared with the value read by the EKS. Data are retrieved only if both values differ.

Temporary data
None.

Name	Data type	Address	Start value	Comment
IN		0.0		
OUT		0.0		
Error	Bool	0.0	FALSE	Error message
NewKey	Bool	0.1	FALSE	New key read
DPStatus	Word	2.0	W#16#0	Status of DP Slave
IN_OUT		0.0		
STAT		0.0		
KeyCount	Byte	4.0	B#16#0	Counter of plugged key
CycleMarker	Int	6.0	0	Sequence counter of program
TEMP		0.0		

Figure 6

Changed registers
A1, A2, SW

Unchanged registers
AR1, AR2, DBR1, DBR2

System functions used
SFC14, DPRD_DAT – read standard DP slaves/PROFINET IO devices
SFC15, DPWR_DAT – write standard DP slaves/PROFINET IO devices

Global data
Data blocks DB1 and DB2 with a minimum size of 32 bytes each are assumed.
The content of data block DB1 is completely overwritten.
The first three bytes (command byte, EKS address and length of the user data) are overwritten in data block DB2.

Symbol table

	Status	Symbol	Adresse	Datentyp	Kommentar
1		Calculate CRC	FB 2	FB 2	
2		COMPLETE RESTART	OB 100	OB 100	Complete Restart
3		Data FB1	DB 10	FB 1	
4		Data FB2	DB 11	FB 2	
5		DIS_AIRT	SFC 41	SFC 41	Delay the Higher Priority Interrupts and Asynchronous Errors
6		DPNRM_DG	SFC 13	SFC 13	Read Diagnostic Data of a DP-Slave
7		DPRD_DAT	SFC 14	SFC 14	Read Consistent Data of a Standard DP Slave
8		DPWR_DAT	SFC 15	SFC 15	Write Consistent Data to a Standard DP Slave
9		EKSactive	E 256.0	BOOL	EKS active
10		EKSIn	E 256.1	BOOL	Fla if key plugged
11		EKSInCount	EB 257	BYTE	Counter of EKS
12		EKSJobactive	E 256.7	BOOL	EKS job is currently active
13		EKSJobdone	E 256.6	BOOL	EKS job is finished
14		EKSMemIn	EB 256	BYTE	First byte of input buffer EKS
15		EKSMemOut	AB 256	BYTE	First byte of output buffer EKS
16		EN_AIRT	SFC 42	SFC 42	Enable Higher Priority Interrupts and Asynchronous Errors
17		F_CTRL_1	FB 273	FB 273	
18		F_CTRL_2	FB 274	FB 274	F_: Test Block an Programm Run Control
19		F_DIAG_N	FB 275	FB 275	F_: Diagnosticbuffer Message with non CPU-Stop
20		F_GLOBDB	DB 545	DB 545	F_: F_Global_Data Block
21		F_IO_CGP	FB 272	FB 272	F_: Driver Block In-Output with Channel Granular Passivation
22		Globaler Speicher	DB 3	DB 3	
23		I/O_FLT1	OB 82	OB 82	I/O Point Fault 1
24		Main Program	OB 1	OB 1	
25		PROG_ERR	OB 121	OB 121	Programming Error
26		RDSYSST	SFC 51	SFC 51	Read a System Status List or Partial List
27		Read EKS	FB 1	FB 1	
28		ReadBufferEKS	DB 1	DB 1	
29		STP	SFC 46	SFC 46	Change the CPU to STOP
30		VAT_1	VAT 1		
31		WriteBufferEKS	DB 2	DB 2	

Figure 8

STL program in FB1- ReadEKS

```
//Check of whether EKS is ready. If not: end of block.
      UN    "EKSactive";           // EKS ready
      BEB   ;
NETWORK
TITLE =Check whether Electronic-Key inserted

      UN    "EKSIn";              // Check whether an Electronic-Key is inserted
      SPB   RES;

NETWORK
TITLE =Electronic-Key inserted
//Check whether Electronic-Key is new.
      L     #KeyCount;            // Already read if same value is contained in the key counter
      L     "EKSInCount";         // Current counter value in EKS
      ==I   ;                     // Compare only if counter value in EKS is higher
      SPBN  ZS;                   // Retrieve data only with new Electronic-Key

      R     #Error;               // Feedback, no error
      R     #NewKey;              // Feedback, no Electronic-Key

      BE    ;
```

Figure 9a

```
NETWORK
TITLE =Start cycle sequence to read the Electronic-Key data
//Start of the actual reading routine from a newly inserted Electronic-Key.
//Cycle sequence.
ZS:   L     #CycleMarker;
      SPL   NEXT;

      SPA   z1;                   // 1st cycle: set Read command
      SPA   z2;                   // 2nd cycle: reset Read command
      SPA   z3;                   // 3rd cycle: read data

NEXT: BEA   ;
NETWORK
TITLE =1st cycle
//Transfer of parameters to the EKS so that the range from byte 103,
//including the KeyID, is sent.
//The transmit-data range is already pre-allocated in DB2 by initialization and
//does not have to be written again.
//The command must be set in each case.
z1:   U     "EKSJobactive";      //Wait until no job is active
      UN    "EKSJobDone";        //and job is done
      BEB   ;

      L     64;                  // Set Read command from Electronic-Key byte 103 with 21 bytes
      T     "WriteBufferEKS".WriteCommand;
      L     103;
      T     "WriteBufferEKS".WriteStartAddress;
      L     21;
      T     "WriteBufferEKS".WriteNumberBytes;

      L     1;
      T     #CycleMarker;       // Prepare cycle marker for the next cycle

      CALL "DPWR_DAT" (         // Call of SFC 15 DPWR_DAT
           LADDR := W#16#100,   // Address of the EKS memory range
           RECORD:= P#DB2.DBX0.0 BYTE 32, // Start address of the DB, length must be 32
           RET_VAL := MW 1);    // Feedback

// Check whether an error occurred
      L     MW     1;           // Write feedback from Profibus
      L     0;                  // Only return value 0 is OK
      ==I   ;
      SPBN  MERR;               // If a value <> 0 was returned: error

      BEA   ;
```

Figure 9b

```
NETWORK
TITLE =2nd cycle
//Reset of the Read command
z2:   UN    "EKSJobactive";    // Wait until job is active and
      U     "EKSJobDone";      // job not ended, then reset of the Read command
      BEB   ;

      L     0;                 // Reset of the Read command
      T     "WriteBufferEKS".WriteCommand;

      L     2;
      T     #CycleMarker;      // Prepare cycle marker for next cycle

      CALL "DPWR_DAT" (        // Call of SFC 15 DPWR_DAT
           LADDR := W#16#100,  // Address of the EKS memory range
           RECORD := P#DB2.DBX0.0 BYTE 32,// Start address of the DB to be sent, length must be 32
           RET_VAL := MW 1);   // Feedback

//Check whether error occurred

      L     MW     1;          // Write feedback from Profibus
      L     0;                 // Only return value 0 is OK
      ==I   ;
      SPBN  MERR;              // If a value <> 0 was returned: error

      BEA   ;
```

Figure 9c

```
NETWORK
TITLE =3rd cycle
//Reading of the Electronic-Key data.
//
z3:   U     "EKSJobactive";    // Wait until no job is active
      UN    "EKSJobDone";      // and job is done
      BEB   ;

      CALL "DPRD_DAT" (        // Call of SFC 14 DPRD_DAT
           LADDR := W#16#100,  // Address of the EKS memory range
           RET_VAL := MW 1,    // Feedback
           RECORD := P#DB1.DBX0.0 BYTE 32);// Start address of the DB for reception, length must be 32

      L     0;
      T     #CycleMarker;

//Check whether error occurred

      L     MW     1;         // Read feedback from Profibus
      L     0;                // Only return value 0 is OK
      ==I   ;
      SPBN  MERR;             // If a value <> 0 was returned: error

//Electronic-Key read completely, the data are now in  DB1
      L     "EKSInCount";    // Read out current counter value from EKS
      T     #KeyCount;       // Note that reading was complete with this counter value
      S     #NewKey;         // Report back that a new Electronic-Key was read completely
      R     #Error;          // No error
      BEA   ;
```

Figure 9d

```
NETWORK
TITLE =Error processing

MERR: L     MW     1;
      T     #DPStatus;       // DP status as feedback in case of error
      S     #Error;          // Return value = 1, error occurred
      R     #NewKey;         // Feedback, no Electronic-Key
      L     0;
      T     #CycleMarker;    // Reset cycle sequence
      BE    ;

NETWORK
TITLE =Reset

RES:  R     #Error;          // No error
      R     #NewKey;         // Feedback, no Electronic-Key
      L     0;
      T     #CycleMarker;    // Reset cycle sequence
```

Figure 9e

FB1 call

```
//Retrieval of data from the EKS Electronic-Key
      CALL  "Read EKS" , "Data FB1"
       Error   :=M0.0                  // Return value for error
       NewKey  :=M0.1                  // Return value, whether new Electronic-Key
       DPStatus:=#Status               // Status of the DP slave

      U     M      0.0                 // Check whether error occurred
      SPB   MERR                       // If values = 1, jump to error routine
```

Figure 10

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 introduced example 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 hazard location and the software within the safety evaluation must also be considered, for example.

The introduced applications 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.

In accordance with Machinery Directive 2006/42/EC, the design engineer of a machine or installation is obligated to perform a risk assessment and take measures to reduce the risk. When doing this, the engineer must comply with the applicable national and international 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 is obligated to assess the safety technology itself. The examples must not be used for 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.

Particularly in case of fault exclusion, it must be noted that this can be performed only by the design engineer of a machine or installation and requires a reason. 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.

EUCHNER GmbH + Co. KG · Kohlhammerstraße 16 · 70771 Leinfelden-Echterdingen
Telephone: +49 711 75 97 -0 · Fax: +49 711 75 97 -303 · info@euchner.de · www.euchner.de