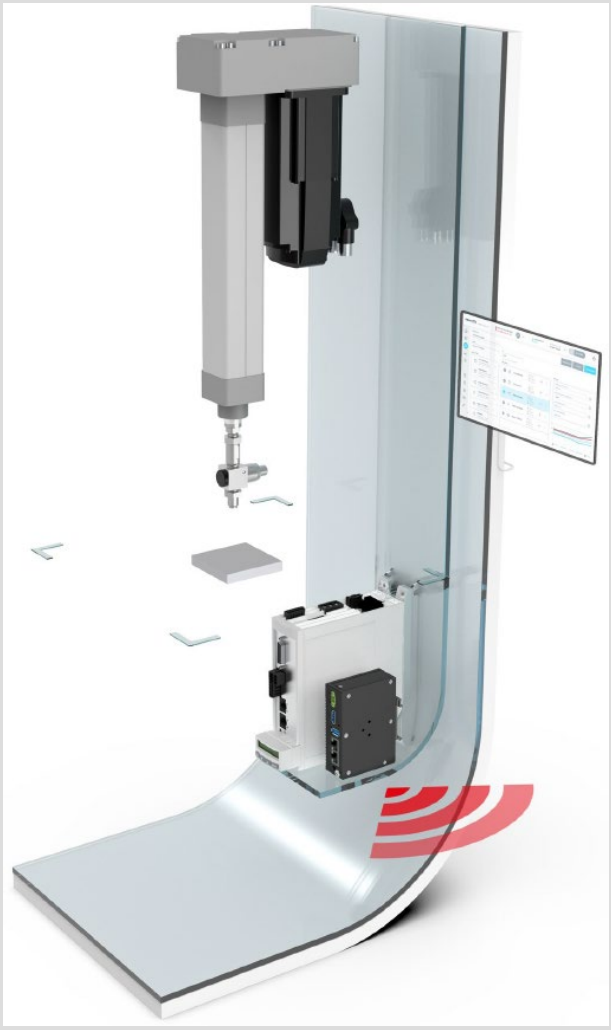
rexroth
A Bosch Company

Smart Function Kit SFK - Fieldbus

Instructions
R320103209 / 2021-07

English



This data has been provided solely for the purpose of product description. Any references to possible uses are provided merely as a convenience and shall be understood as sample applications or suggestions. Catalog data may not be construed as guaranteed characteristics. The information given does not release the user from the obligation of own judgment and verification. It should be noted that our products are subject to a natural process of aging and wear.

The title page contains an illustration of a sample configuration. The product as delivered can differ from the illustration.

The original instructions have been prepared in German.

Contents

1 About this documentation

1.1 Required and supplementary documentation

- Only commission the product once you have obtained the system documentation that is marked with a book symbol 🕮 and understood and complied with its contents.

Table 1: Required and supplementary documentation

	Title	Document number	Document type
🕮	Smart Function Kit SFK - Instructions	R320103194	Instructions
🕮	Smart Function Kit SFK - Software	R320103208	Instructions
	Smart Function Kit SFK - Fieldbus	R320103209	Instructions

2 Introduction

Via the fieldbus interface, the *Smart Function Kit* can be used as a subsystem in an automation plant.
The following fieldbus protocols are available for the *Smart Function Kit*

- SercosIII
- EtherCAT (SoE)
- Profinet
- Ethernet/IP

The fieldbus protocol can be selected via the *Selection list* in the *System Configuration* area. The exact procedure for this is explained in the following chapter.

NOTE
To ensure error-free operation as a subsystem of a master control, data exchange must be carried out according to the instructions described here.

3 Hardware

3.1 Topology

The following figures show the physical integration of the *Smart Function Kit* (consisting of IndraDrive HCS01 and PR21) into a higher-level automation system. In doing so, care must be taken to ensure correct cabling and use of the appropriate device ports to ensure error-free communication between the devices involved.
In the figures, a Rexroth XM21 control system is used as fieldbus master. This control version already has an integrated "SercosIII-Master" fieldbus connection. For the deployment as a Profinet master, the XFE01.1-FB-03 expansion module is used.
It is possible to use third-party hardware for the fieldbus master. In this case please refer to the additional documentation on the homepage.

Figure 1 shows the structure of a Profinet fieldbus.

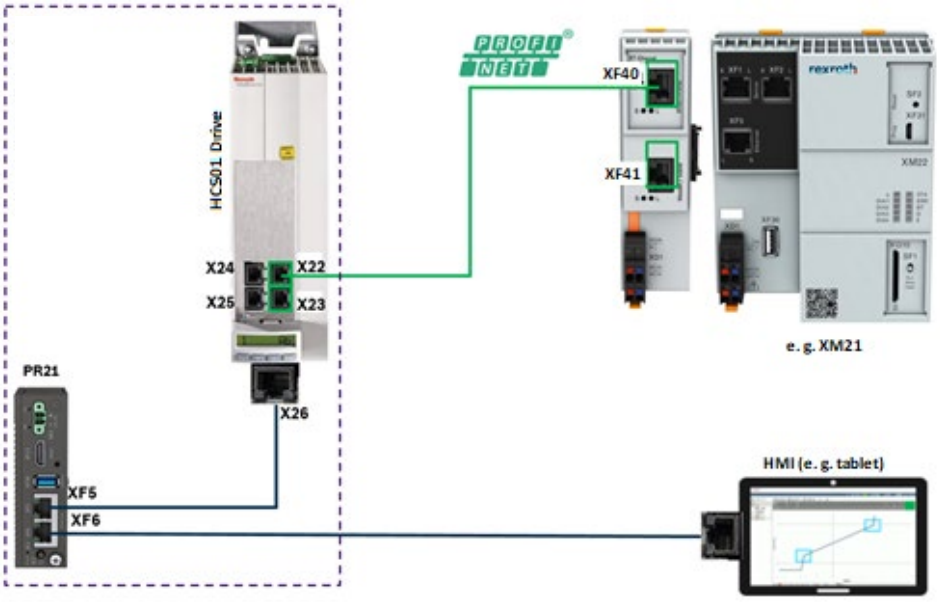


Figure 1: Physical structure of a Profinet fieldbus

Figure 2 shows the structure and cabling of a SercosIII fieldbus.
This fieldbus can be designed either as a line or ring topology. To increase the fault tolerance against EMC interference and cable defects, the SercosIII master expects a ring topology (cable 1 + 2). This ensures redundancy of the data.

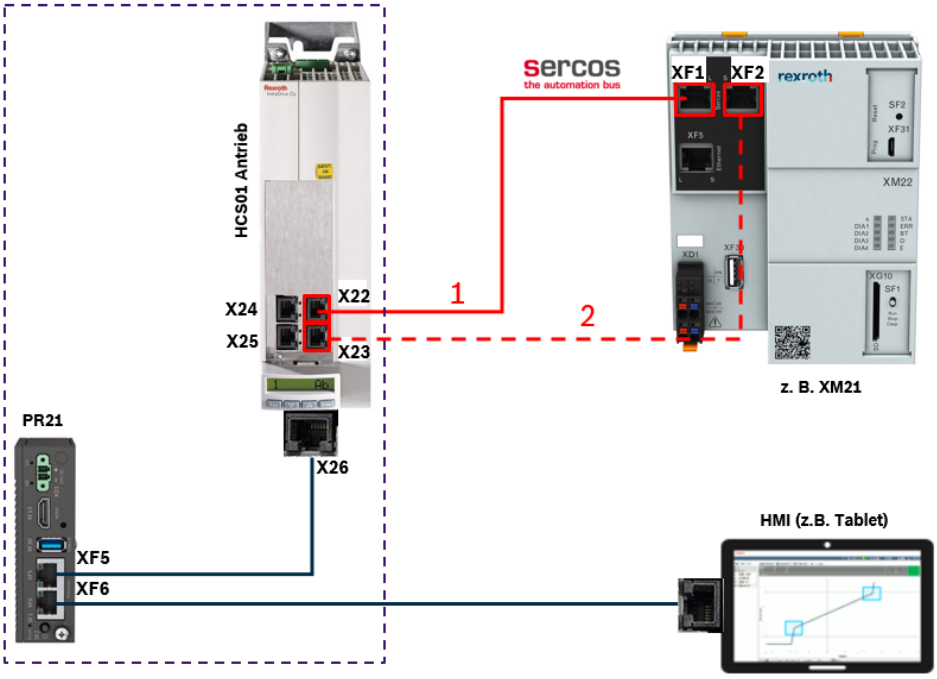


Figure 2: Physical structure of a SercosIII fieldbus

3.2 Settings

3.2.1 General requirements

For communication with the *Smart Function Kit* web interface, a faultless network connection between HMI (e.g. tablet) and PR21/XF6 is required.
The following requirements apply to the HMI (e.g. notebook, tablet) for connection to the *Smart Function Kit*:

- ✓ Web browser with HTML5 and Java Script support
  e.g. Firefox **version 52** or higher, Chrome **version 74** or higher
- ✓ Screen diagonal 10 inches (format 16:9) or higher
- ✓ Screen resolution 120px 720px or higher

3.2.2 IP addresses configuration

The correct configuration of the IP addresses and subnet masks for the individual ports of the devices is a prerequisite for error-free communication.
The standard settings are listed below:

- ✓ IndraDrive HCS01 – X22/X23 (MT):
  - IP address: 172.31.254.1
  - Subnet mask: 255.255.255.0

- ✓ IndraDrive HCS01 – X26 (ET):
  - IP address: 192.168.1.1
  - Subnet mask: 255.255.255.0

- ✓ Industrial PC PR21 – XF5 (cannot be changed):
  - IP address: 192.168.1.100
  - Subnet mask: 255.255.255.0

- ✓ Industrial PC PR21 – XF6 (can be changed via the web interface):
  - IP address: 192.168.0.1
  - Subnet mask: 255.255.255.0

NOTE
The IP addresses can be changed or adapted individually for some ports. The corresponding configuration procedure is explained below.

It is possible to adapt or change the IP addresses for ports X22/X23 (MT) on the IndraDrive HCS01.
Do not change the IP address of port X26.
Figure 3 shows the procedure using the example of ports X22/23 (MT) via the control unit of the drive.

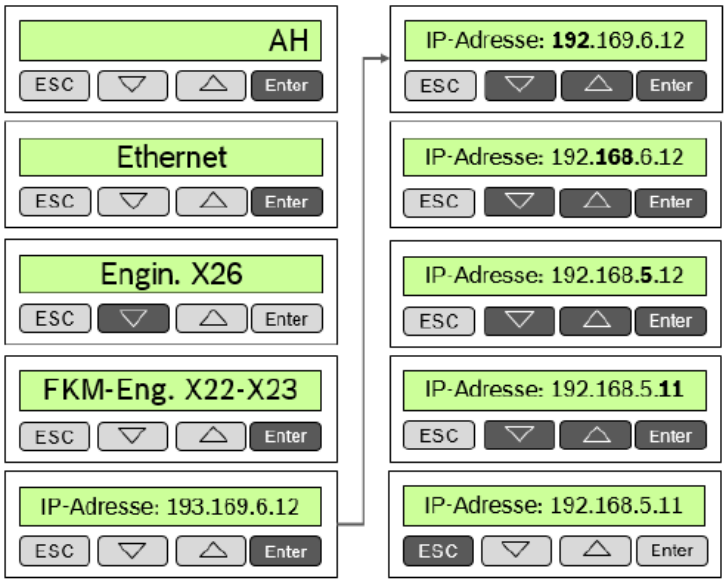


Figure 3: Adaptation of the IP address via the operating panel of the drive

On the PR21 industrial PC, only the IP address for port XF6 can be changed, which is used to establish the connection between the HMI (e.g. tablet) and the web interface. For adaptation or change of the IP address/subnet mask, the ***Configuration*** can be opened after registration. In the ***System*** tab under ***Network***, the IP address and subnet mask for port XF6 can be changed.

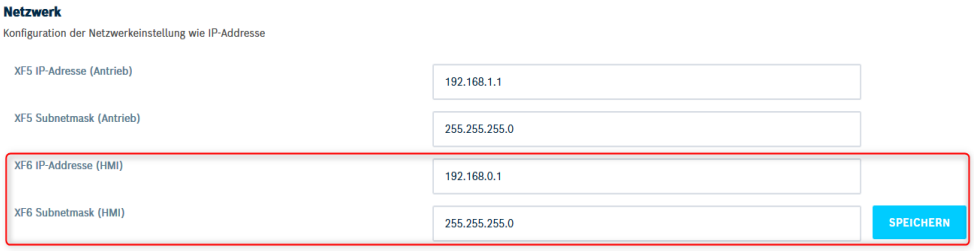


Figure 4: Network settings PR21 - Adjust XF6

3.2.3 Activation Fieldbus

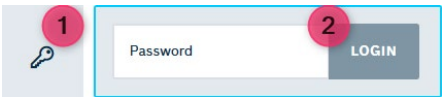
In order for the *Smart Function Kit* to communicate or control via a fieldbus, the corresponding fieldbus protocol must first be selected in the web interface. To do this, the following steps are required:

- ✓ Open web browser in the HMI, clear cache in case of connection problems, and delete history, cookies and other data.
  Note: Web browsers usually have a private mode in which no data such as cookies are stored. If this is activated, there is no need to empty the cache or delete the history.

- ✓ Enter the IP address in the address bar.
  Standard IP address: 192.168.0.1

- ✓ Web interface of the *Smart Function Kit* is opened

- ✓ Login with password: **admin (if not changed)**



- ✓ Scroll down under ***Configuration → System***

- ✓ Select the corresponding type under ***Fieldbus configuration***. If needed, change the Device-name from “SPK” to another name.

- ✓ Save.

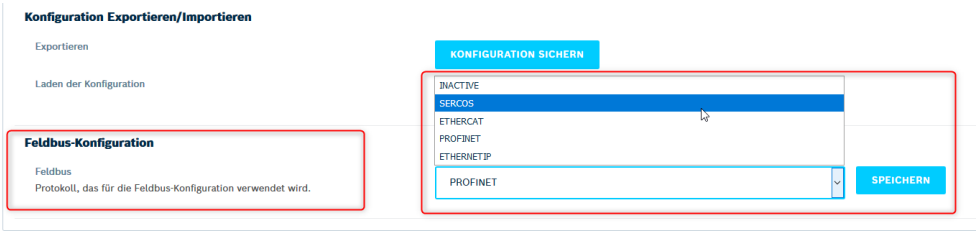


Figure 5: Selection of the fieldbus protocol in the web interface

The assignment of a device-name is possible in Profinet and Ethernet/IP. By default, the name “SPK” is defined.
The device name has to meet the following conventions:

- ✓ No more than 240 characters (letters, numerals, hyphen or dot)
- ✓ The length of a character string between two dots may not exceed 63 characters
- ✓ No special characters except for hyphens (umlauts, brackets, underscore, slash, blank, etc.)
- ✓ Neither the first nor the last character of the device name may be '-'
- ✓ The device name may not have format n.n.n.n (n = 0...999)
- ✓ The device name may not begin with string 'port-xyz-' (x,y,z = 0...9)

4 Configuration Fieldbus

To establish the communication, the master control must be configured correctly. Depending on the fieldbus protocol, the configuration can look different.
As examples, the configurations for the SercosIII and Profinet fieldbus based on an XM21 control from Bosch Rexroth are described below. *IndraWorks Engineering* is used as the development environment.

4.1 Create variable list

For communication between the *Smart Function Kit* and the higher-level control, specific data is transmitted via the fieldbus in the form of "words". These are recorded by the fieldbus master as inputs or outputs.
The following figure shows an overview of the fieldbus data.

ET-Master (Input 20 Words)
responseHandle (Word)
responseStatus (Word)
responseValue1 (DWord)
responseValue2 (DWord)
responseValue3 (DWord)
notificationID (Word)
notificationValue (DWord)
notificationStatus (Word)
notificationPosition (DWord)
notificationVelocity (DWord)
notificationForce (DWord)
digital IOMapping (Word)
vitual IO input (Word)

ET-Master (Output 9 Words)
requestHandle (Word)
requestAction (Word)
requestValue1 (DWord)
requestValue2 (DWord)
requestValue3 (DWord)
vitual IO output (Word)

To enable the fieldbus master to access the corresponding fieldbus data via functions, it is useful to assign these data to global variables.
A practical implementation consists of creating one global array of the "WORD" data type each for the input and output data.
The corresponding assignment of the input and output data of the fieldbus to the corresponding array elements (mapping) is explained below for the respective fieldbus protocols.

The following listing shows a printout from the global variable list *SFK_GlobalVariables* of the example projects provided on the homepage.

```
VAR_GLOBAL
  ///Cyclic mapping variables
  InputDataContainer: ARRAY[0..19] OF WORD;
  OutputDataContainer: ARRAY[0..7] OF WORD;
  virtualOut: WORD;
END_VAR
```

Listing 1: Excerpt from global variables list "SFK_GlobalVariables"

4.1.1 Fieldbus configuration

The first step is to add the *Smart Function Kit* drive to the fieldbus master as a Sercos device:

✓ Start IndraWorks and add new project (empty project)

✓ Add fieldbus master (here XM21)
Right mouse button → Add → IndraMotion MLC →IndraControl XM2

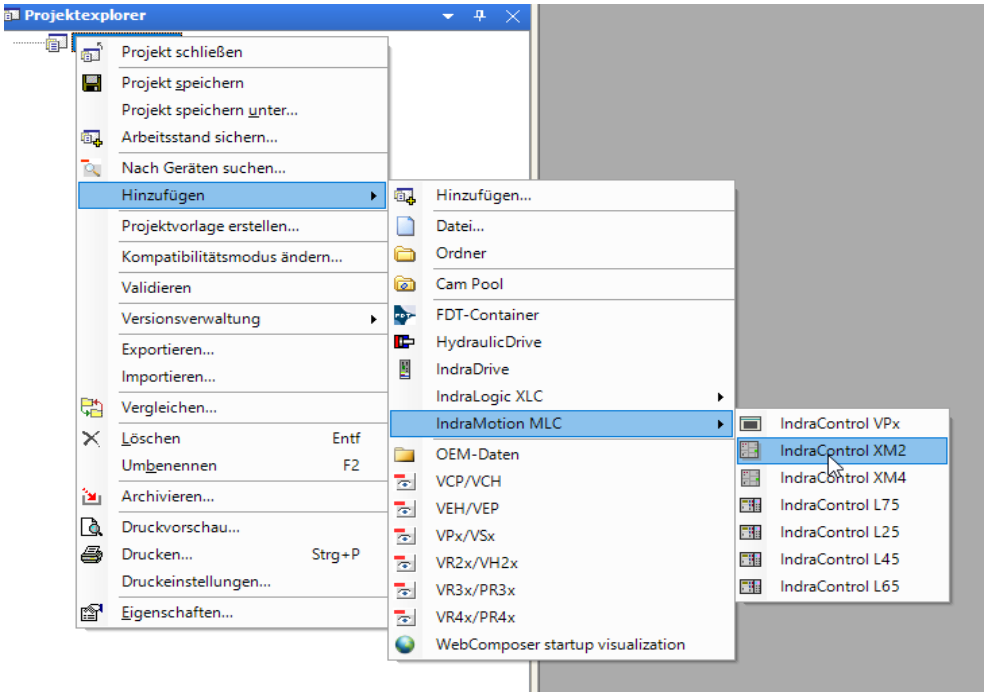


Figure 7: Add field bus master to the project

✓ Adding the Sercos device (IndraDrive HCS01)

a) Select Library → Periphery → Device "HCS0x"
or

b) Drag and drop under the Sercos node in the project.

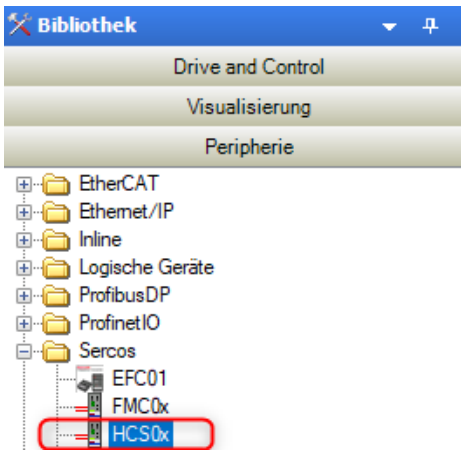


Figure 8: Adding IndraDrive HCS0x via library

In the next step, the fieldbus addresses in IndraWorks must be compared with those of the device and adapted if necessary. (Under HCS0x → General)

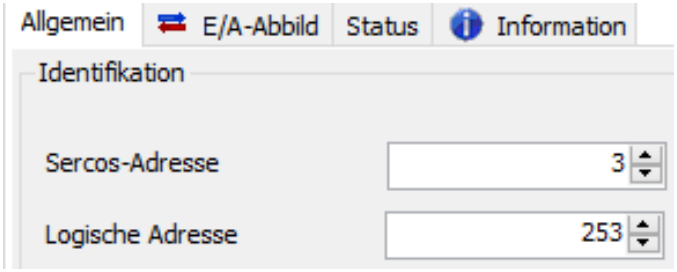
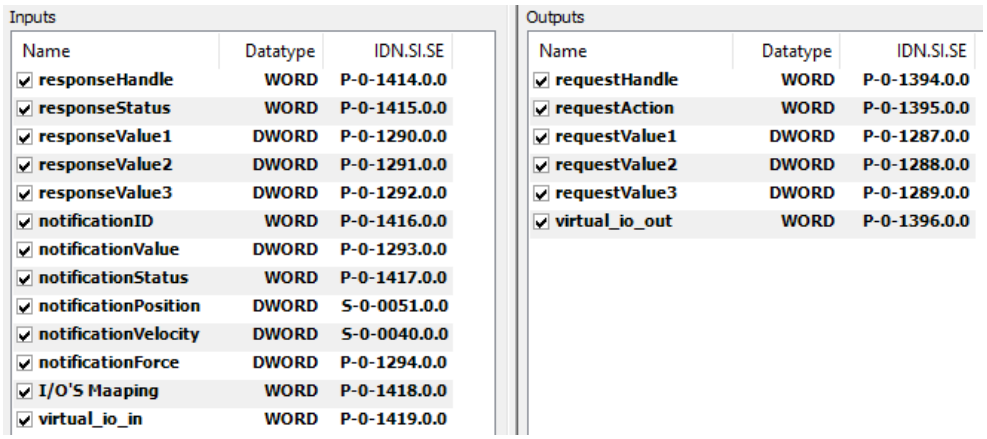


Figure 9: Adapting the field bus addresses

After that, the cyclic inputs and outputs must be adapted in such a way that they conform to the following figure.



Inputs

Name	Datatype	IDN.SI.SE
responseHandle	WORD	P-0-1414.0.0
responseStatus	WORD	P-0-1415.0.0
responseValue1	DWORD	P-0-1290.0.0
responseValue2	DWORD	P-0-1291.0.0
responseValue3	DWORD	P-0-1292.0.0
notificationID	WORD	P-0-1416.0.0
notificationValue	DWORD	P-0-1293.0.0
notificationStatus	WORD	P-0-1417.0.0
notificationPosition	DWORD	S-0-0051.0.0
notificationVelocity	DWORD	S-0-0040.0.0
notificationForce	DWORD	P-0-1294.0.0
I/O'S Maaping	WORD	P-0-1418.0.0
virtual_io_in	WORD	P-0-1419.0.0

Outputs

Name	Datatype	IDN.SI.SE
requestHandle	WORD	P-0-1394.0.0
requestAction	WORD	P-0-1395.0.0
requestValue1	DWORD	P-0-1287.0.0
requestValue2	DWORD	P-0-1288.0.0
requestValue3	DWORD	P-0-1289.0.0
virtual_io_out	WORD	P-0-1396.0.0

Figure 10: Adaptation of the cyclic input and output data of the field bus

For this purpose, an input form is opened via the "Add" button, where the name of the input or output, the data type and the corresponding parameter can be entered. Figure 12 shows the form.

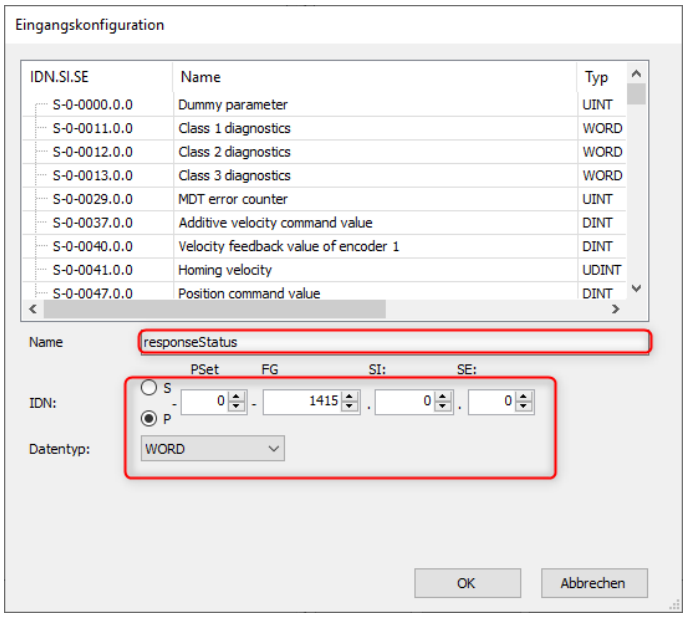


Figure 11: Input form for the cyclic input and output data

Finally, the inputs and outputs of the fieldbus must be assigned to the internal variables via the "I/O image" so that the fieldbus master can access the fieldbus data via functions, as mentioned above.

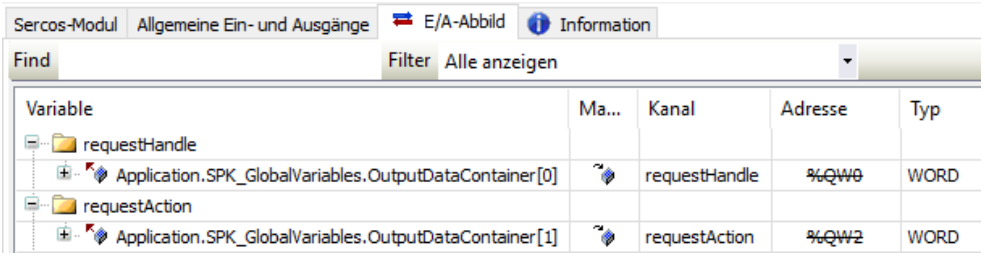


Figure 12: I/O mapping to internal program variables

NOTE
After configuration, a restart of the fieldbus may be required. **This restart is carried out with the following changes of the control modes: Operating mode (BB) → Download mode (P0) →Parameterization mode (P2) → Operating mode (BB)**

4.1.2 Profinet Fieldbus configuration

After a new project (empty project) has been created in IndraWorks and the fieldbus master (here Bosch Rexroth XM21) has been added, the device *Profinet-IO-Controller XM* must be added to the project tree under the tab ***RT_Ethernet...***.

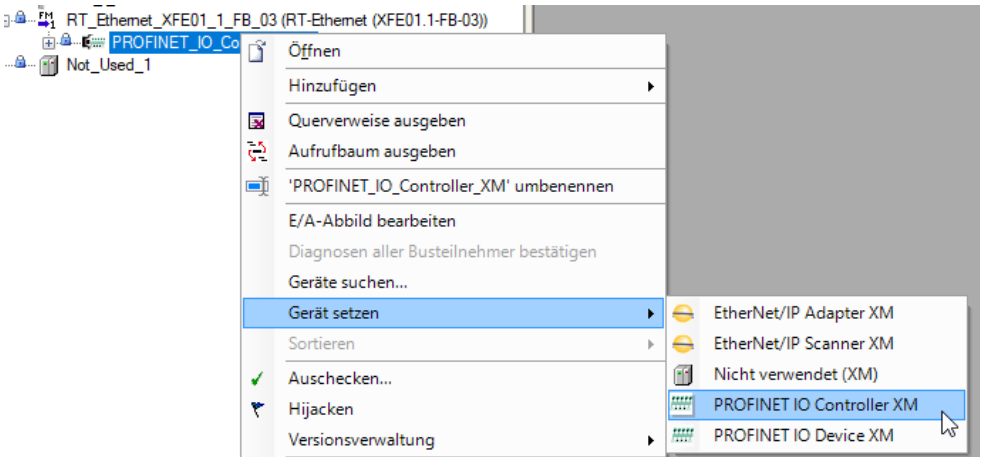


Figure 13: Insert Profinet IO controller

In the next step, the IndraDrive HCS01 is added to the Profinet IO controller as a fieldbus slave. Two methods are possible here:

a) Via ProfiNet-IO_Controller (right mouse button)

Add → Device → Drives →IndraDrive 02V02 GSDML V2.1

or

b) Via library (drag & drop)
Library → Periphery → ProfinetIO → Drives → IndraDrive 02V02.

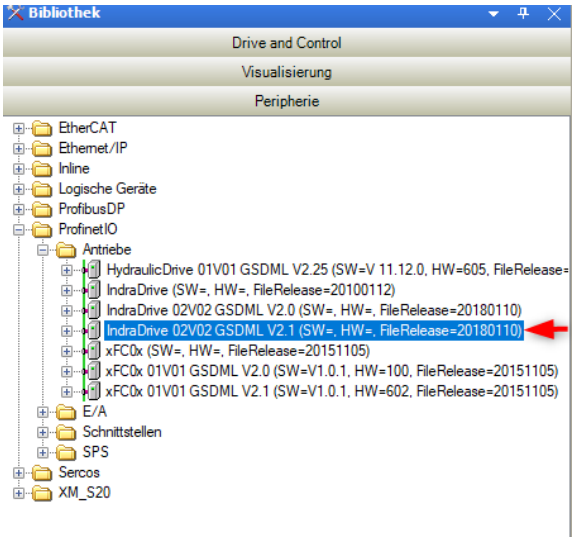


Figure 14: Adding the IndraDrive HCS01 via library

In the corresponding *General* tab, "**SPK**" must be entered under Station name or the user-defined value. No distinction is made here between lower and upper case.
In the project tree, the IndraDrive HCS01 (Profinet slave) is displayed under the *Profinet IO controller* as shown in figure 16.

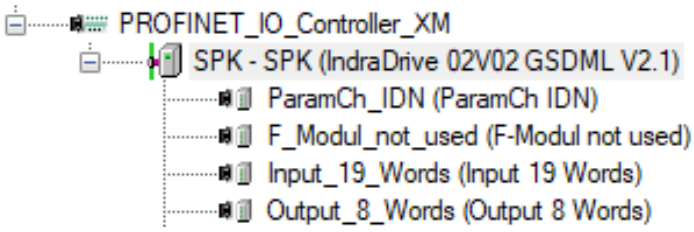


Figure 15: Profinet node structure

If Profinet is used, the IndraDrive obtains its IP address automatically from the fieldbus master (for this reason it is necessary to enter the correct station name).
The following standard IP addresses are assigned by the fieldbus master:

- ✓ PROFINET_IO_Cotroller_XM (Master): 192.168.2.1 / 255.255.255.0
- ✓ SPK (IndraDrive 02V02) (slave): 192.168.2.2 / 255.255.255.0

To enable access to the fieldbus data (inputs and outputs), the corresponding blocks of the fieldbus slave must be configured or added as follows:

a. ParamCH_IDN
b. Input_20_Words
c. Output_9_Words

Finally, the inputs and outputs of the fieldbus must be assigned to the internal variables via the *I/O image* so that the fieldbus master can access the fieldbus data via functions, as mentioned above.

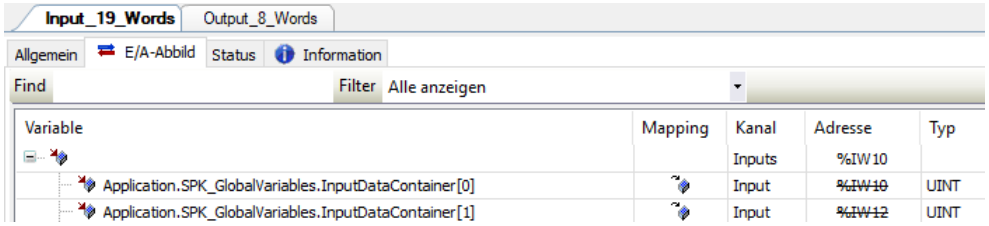


Figure 16: I/O mapping to internal variables

5 Communication

Control via the fieldbus is carried out via requests, responses and notifications. For each request, the response corresponding to the request is sent. Notifications are divided into cyclic and acyclic information.
The following figure schematically shows the functional principle of the communication.

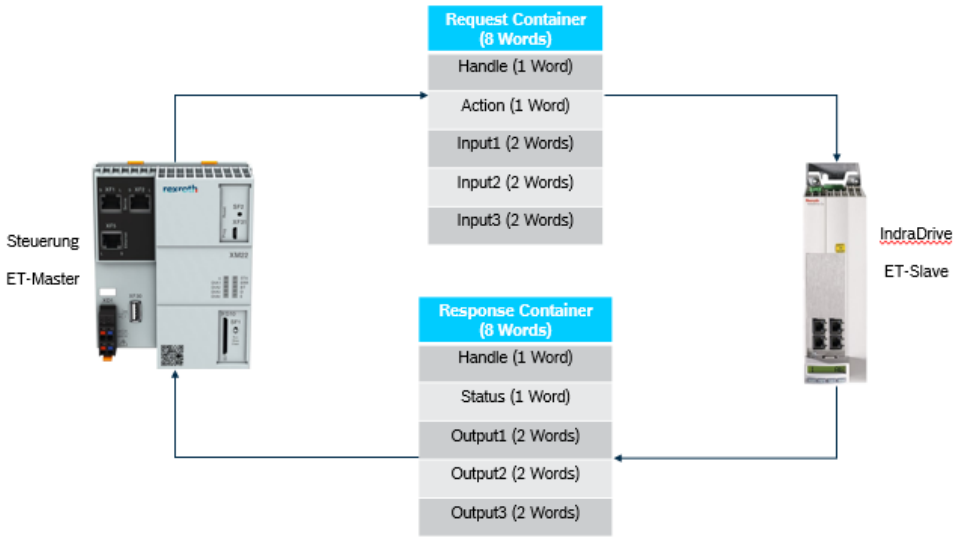


Figure 17: Schematic structure of field bus communication

5.1 Requests

The response to a request is given after the entire command/request has been processed.
The following table contains all available commands that can be sent via the fieldbus.

Requests:

ID	Command	Value 1	Value 2	Value 3
1	Start program	(Serial number digit 1-7)	(Serial number digit 8-14)	(Serial number digit 15-21)
2	Set program active	Fieldbus ID	x	x
3	Positioning	Target position	Travel speed	Acceleration
4	Jog	Distance	Travel speed	Acceleration
5	Delete error	x	x	x
6	Stop movement	x	x	x
7	Restart drive	x	x	x
8	Reserved	-	-	-
9	Tare	Offset		
10	Start homing	x	x	x
11	Set system variable	Type	Number	Value

12	Lock invoker	Invoker	Lock (1) / Unlock (0)	x
13	Set reference	x	x	x
14	Get system variable	Type	Variable number	x

Start program:
With this command, the currently active program is started. It is processed once completely and after completion the corresponding response is provided. The program sequence is considered to be finished when the quality assessment has been completed.
There is the possibility to link a custom serial number with up to 21 digits in total to the started program dataset, which will be stored in the SFK database. Therefore, the serial number has to be split up into up to three real numbers, depending on the total length. The following example shows the right transition behavior:
Serial number: '123451234567541'
→ Value 1 = 4567.541 | Value 2 = 2345.123 | Value 3 = 0.001

Set program active:
The command activates the program specified in value 1. The fieldbus ID corresponds to the fieldbus ID assigned during program creation. If the program is switched, the corresponding response is provided.

Positioning:
The command enables the movement to a defined target position. The desired travel speed profile can be realized via the parameters travel speed and acceleration. When the target position is reached, the corresponding response is provided.

Jog:
With the *Jog* command, a distance relative to the current position can be covered. The desired travel speed profile can be realized via the parameters travel speed and acceleration. When the target position is reached, the corresponding response is provided.

Delete error:
The command acknowledges all errors present on the Smart Press Kit. When the command has been processed by the system, the corresponding response is provided.

Stop movement:
The command terminates any currently pending command. If a movement is active, it is aborted and the drive is stopped. When the drive is stopped, the corresponding response is provided.

Restart drive:
The command restarts the drive controller.

Tare:
Executes the Tare command and sets the actual force value to the specified offset. When the command has been completed, the corresponding response is provided.

Start homing:
Executes the homing command. Check the notes in chapter ‘Homing’ in the software manual before executing first.

Set system variable
With this command, the system variables specified in the table below can be modified. For these values, the system does not switch to parametrization mode. Check the chapter “Extended parameters” (see Instructions R320103208 „Smart Function Kit SFK – Software“) for further information.

Variable	Description	Type	Number	Value
VFR004	Set maximal positioning force	0	4	Force-Value [N] 0=inactive (default)
MFR002	Filter actual values	1	2	0= No | 1= Yes
MFR005	Cyclic variables documentation	1	5	0= No | 1= Yes

Encoding of variable types:
0: VFR 10: VF
1: MFR 11: MF

Example: Set system variable VF500
Arg. 1: 10
Arg. 2: 500
Arg. 3: value

Lock invoker:
This command locks or unlocks access to the SFK, filtered by the type of invoker. Commands from a locked invoker do not affect the system anymore. The lock settings do last until a reboot. The HMI will still show current values if locked. Types of invokers:

- Web HMI (**1**)
- Fieldbus (**2**)
- Physical In- & Outputs (**3**)

The first input parameter selects the invoker to be locked – see the numbers above. The second input parameter is used to decide between locking and unlocking of the specified invoker (first input parameter). Set the value like:
0: Unlock Invoker
1: Lock Invoker

Set reference:
"Set reference" command will set the absolute reference for the system at the current position. By default, the value is set to 0 mm, but can be changed globally for the system inside extended hardware configuration settings.

Get system variable:
This command can be used to read system variables. The first parameter specifies the variable type (s. table "Encoding of variable types"). The second parameter specifies the variable number.

Encoding of variable types:

0: VFR	10: VF
1: MFR	11: MF

Example: Get system variable VF500
Arg. 1: 10
Arg. 2: 500

The Smart Function Kit will return the value of VF500 within ResponseValue[1].

Also, the following values can be read for the last pressing cycle via the fieldbus interface:

VF600	Max. position
VF601	Max. force
VF602	Cycle time

These values are only valid after a pressing process and must be read before starting the next pressing. The duration required for the readout is non-deterministic, as this is done via the NRT channel.
An invalid return value shows as -1. This can occur, for example, due to a program abort in the case of the cycle time.

5.2 Status information

With the status information, the states of the *Smart Function Kit* can be queried. This enables targeted control of the *Smart Function Kit* via the fieldbus master. The following tables show the corresponding commands (ID) and assignments within the status word.

Status information:

ID	Description	Functional description
301	Active program number	The program number active in the drive is read out.
401	Current position	Cyclically transmitted position is read out
402	Current travel speed	Cyclically transmitted travel speed is read out
403	Current force	Cyclically transmitted force is read out

Status word:

Bit	Description	Functional description
0 - 2	Reserve	
3	Program active	1 – Pressing active, program is being processed Data is recorded (force - travel curve) 0 – No program active No data is recorded
4	Last pressing OK	1 – Last pressing provides result Good 0 – No result
5	Last pressing NOK	1 – Last pressing provides result Bad 0 – No result
6	Sensor tared	1 – Force sensor is tared (offset active) 0 – Force sensor is not tared (no offset active)
7	Press ready	1 – Press is ready and can be moved 0 – Press is not ready, check status
8	Errors	1 – Error active 0 – No error active
9	Warning	1 – Warning active 0 – No warning active
10	Reserve	
11	Request possible	1 – A request can be sent to the system. 0 – No request can be sent to the system
12	Response available	1 – Valid response available 0 – Response pending
13	Notification available	1 – Valid notification available 0 – Notification pending
14 - 15	Reserve	

5.3 Virtual I/O

Virtual I/Os can be used within the program sequence of the Smart Function Kit (slave) to interact between the running program and the master control. This enables a variety of use cases, such as dynamic stops during the running program sequence to wait for external evaluations or signals.

For each direction, 16 bits can be written and red any time via fieldbus. To access the functionalities, mapping of the required input and output bits has to be done in the Smart Function Kit settings menu.

5.4 System errors

A detailed list of all available error codes, their meaning and associated solution steps can be found in the software manual R320103208 in the chapter "Error Codes" / “System Errors.

6 Programming example

6.1 Example Positioning

To move to a specific target position with the *Smart Function Kit*, the *Positioning* command can be sent via the *Request* channel. When the target position is reached, the result is displayed in the *Response* container.

Request scheme for container request

Handle ID	Command ID	Value1		Value2		Value3	
151	3	50		150		150	
Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8

The Handle ID is a freely selectable integer number that is used to identify a request and the associated response. The ID can be used to check whether the current response matches the request. Two consecutive commands must necessarily have a different Handle ID, otherwise no unique assignment can be made.
When the target position is reached, the *Smart Function Kit* responds with the following response scheme:

Response Container

Handle ID	Status	Value1		Value2		Value3	
151	0	0		0		0	
Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8

Status = 0 → No error
Status <> 0 → Error. The status value corresponds to the error number of the occurred error.

6.2 Integration/Usage of example projects

It is also possible to use the supplied example implementations for the Sercos and ProfiNet fieldbus protocol.
The examples serve as an abstraction layer with the aim of making it easier for the user to familiarize himself with the request and response scheme. The function blocks based on the PLCOpen notation simplify the interface and enable a more intuitive use.

- ✓ Detailed description/integration and functionality in the respective documentation of the example projects
- ✓ The example projects can be downloaded directly in the user interface. Navigate to the help-section there and select the appropriate project.