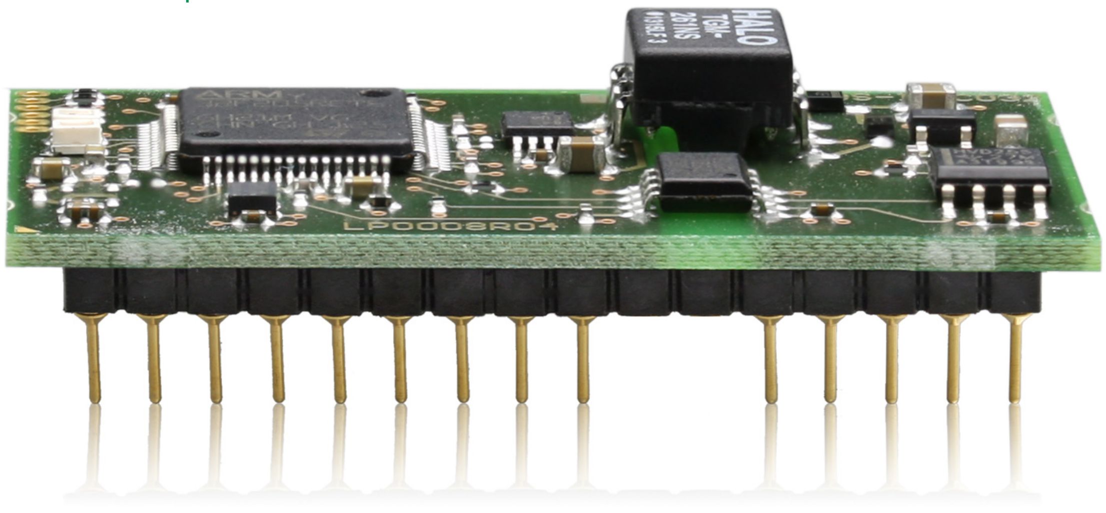


User Manual

KUNBUS-IC module for CANopen

100058_UM00EN

Table of Contents

1 General information 4
1.1 Disclaimer 4
1.2 Notes regarding this user manual 4
1.3 Validity 5
1.4 Limitation of Liability 5
1.5 Customer Service 5

2 Safety Guidelines 6
2.1 User 6
2.2 Symbols 6
2.3 General Safety Guidelines 7
2.4 Environmental Conditions 7

3 Overview 8
3.1 Introduction 8
3.2 Application Interface 8
3.3 Status LEDs 9

4 Components 11
4.1 Module Components 11
4.2 Storage Unit 12
4.3 Data Broker 13
4.4 Fieldbus Interface 21
4.5 CDI - Configuration and Debug Interface 21
4.6 SDI - Serial Data Interface 22
4.7 Synchronous serial interface 23
4.8 Scripter 34

5 Commissioning 35
5.1 Installation 35
5.2 Configuration 41
5.3 Firmware Update 41

6 Memory Register 42
6.1 Overview of the Memory Register 42
6.2 General Device Parameters 45
6.3 Register for the Mapping 67
6.4 Memory of the Communication Channels 69
6.5 Fieldbus specific Registers 71

7 Communication model 79
7.1 Object Directory 79

8 CDI 83
8.1 Setting up a Serial Connection 83

8.2 CDI Menus 85
9 Disposal 117
9.1 Dismantling and Disposal 117
10 Technical data 118
10.1 Technical data 118
11 Appendix 119
11.1 Configuration via Modpoll 119

1 General information

1.1 Disclaimer

KUNBUS GmbH
Heerweg 15 c
73770 Denkendorf
Deutschland

www.kunbus.de

1.2 Notes regarding this user manual

This user manual provides important technical information that can enable you, as a user, to efficient, safe and convenient integration of the IC-Module into your applications and systems. It is intended for trained, qualified personnel, whose sound knowledge in the field of electronic circuits and expertise of CANopen is assumed.

As an integral part of the module, the information provided here should be kept and made available to the user.

1.3 Validity

This document describes the application of the KUNBUS IC-Moduls with the product number:

- PR100009 (variant with 5 Volt), Release 02
- PR100058 (variant with 3.3 Volt), Release 00

1.4 Limitation of Liability

Warranty and liability claims will lapse if:

- the product has been used incorrectly,
- damage is due to non-observance of the operating manual,
- damage is caused by inadequately qualified personnel,
- damage is caused by technical modification to the product (e.g. soldering).

1.5 Customer Service

If you have any questions or suggestions concerning this product, please do not hesitate to contact us:

KUNBUS GmbH
Heerweg 15 C

+49 (0)711 3409 7077

support@kunbus.de
www.kunbus.de

2 Safety Guidelines

2.1 User

The Modul may only be assembled, installed and put into operation by trained, qualified personnel. Before assembly, it is absolutely essential that this documentation has been read carefully and understood. Expertise in the following fields is assumed:

- Electronic circuits,
- Basic knowledge of CANopen,
- work in electrostatic protected areas,
- Locally applicable rules and regulations for occupational safety.

2.2 Symbols

The symbols used have the following meaning:

⚠ DANGER	**Hazard** Observe this information without fail! There is a safety hazard that can lead to serious injuries and death.
⚠ CAUTION	**Caution** There is a safety hazard that can result in minor injuries and material damage.
NOTICE	**Note** Here you will find important information without a safety hazard.

2.3 General Safety Guidelines

⚠ DANGER	**Danger of electric shock**

If unsuitable power supply is used, this can cause an electric shock.

- ➢ This can cause death, serious injuries and material damage to your systems and modules.

➔ Only use a power supply that complies with the regulations for safety extra-low voltage (SELV) or protective extra-low voltage (PELV).

⚠ CAUTION	**Fault due to mechanical load**

A continuous mechanical load of over 5 G or shock loads of over 15 G can cause faults on your modules.

➔ Comply with these load limits and avoid any unnecessary loads.

⚠ CAUTION	**Damage due to subsequent processing**

Avoid subsequent processing of the IC-Modul.

- ➢ Soldering can cause components to become detached and thus damage or destroy the module.
- ➢ Please note that the warranty shall become invalid if the products are changed technically.

➔ Speak to your contact person at the KUNBUS GmbH about customised solutions.

2.4 Environmental Conditions

Operate the IC-Modul only in an environment that complies with the operating conditions in order to prevent any damage.

Suitable Environmental Conditions:

Operating temperature	0 °C to +60 °C
Humidity	0% not 95%, non-condensing

3 Overview

3.1 Introduction

With the KUNBUS IC-Modul you can make a sensor or actuator fieldbus-capable. To do this, simply insert the module into your application and connect it to the fieldbus.

The IC-Modul thereby saves you time-consuming in-house developments.

3.2 Application Interface

The main board is connected to the device controller via a 32-pin connector strip. Thus, you have the option to plug the module directly into your DIL socket.

NOTICE

If the module is plugged in and unplugged frequently, mechanical stresses may damage the module.

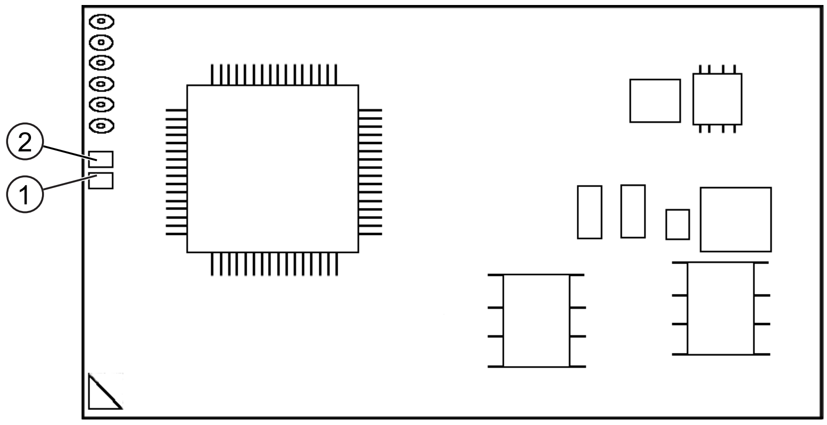
Use a zero insertion force socket to prevent damage.

You can find detailed information on this topic in section"Installation [▸ 35]".

3.3 Status LEDs

On-Board LEDs

When you configure the first output shift register of the SSC channel as fieldbus status display, the following LEDs are available as indicators:



No.	LED	Bit	State	Meaning	Comment
1	Module Error (Red)	1	Off	No error has occurred	
			Flashes	Configuration error	At least one system component is not running due to a configuration error.
			On	Internal system error	A fatal internal system error has occurred. The error is entered in the error stack.
2	Module Operation (Green)	0	Off	Module not running	
			Flashes	Start-up	At least one system component has not yet finished its initialisation. This also includes the automatic bitrate detection.
			On	Normal operation	All system components are running perfectly

You have the option to integrate additional LEDs into your application. These LEDs can be activated using the shift register.

✓ To do this, configure the first output shift register

- In register 0x0025 or
- in CDI menu `"2.3 SSC Communication"`

The signals have the following meaning:

LED	Bit	State	Meaning	Comment
Module Operation (Green)	0	Off	Module not running	
		Flashes	Start-up	At least one system component has not yet finished its initialisation. This also includes the automatic baudrate detection.
		On	Normal operation	All system components are running perfectly
Module Error (Red)	1	Off	No error has occurred	
		Flashes	Configuration error	At least one system component is not running due to a configuration error.
		On	Internal system error	A serious, internal system error has occurred. The error is entered in the error stack.
		Flashes	Start-up	At least one system component has not yet finished its initialisation. This also includes the automatic baudrate detection.
		On	Normal operation	All system components are running perfectly
Script RUN (green)	2	Flashes	Script deactivated or not loaded	
		On	Script is running	
Script ERR (Red)	3	Off	No Error	
		On	Error in Script	
CANopen Run (Green)	4	Off	Module not running	
		On	CANopen State: Operational	
		1 Flash	CANopen State: Stopped	
		Flashes	CANopen State: Preoperational	
		Flickers	CANopen Autobaud Detection or LSS	CAN Automatic Baudrate Detection active or LSS Node setting active
CANopen Error (Red)	5	Off	No error pending	
		On	Bus Off	CAN Controller is in Bus-Off State
		1 Flash	Warning Limit reached	CAN Controller has reached a Warning Limit

4 Components

4.1 Module Components

The module is divided into independent components to ensure a high level of flexibility in the application.

The following pages describe the individual components:

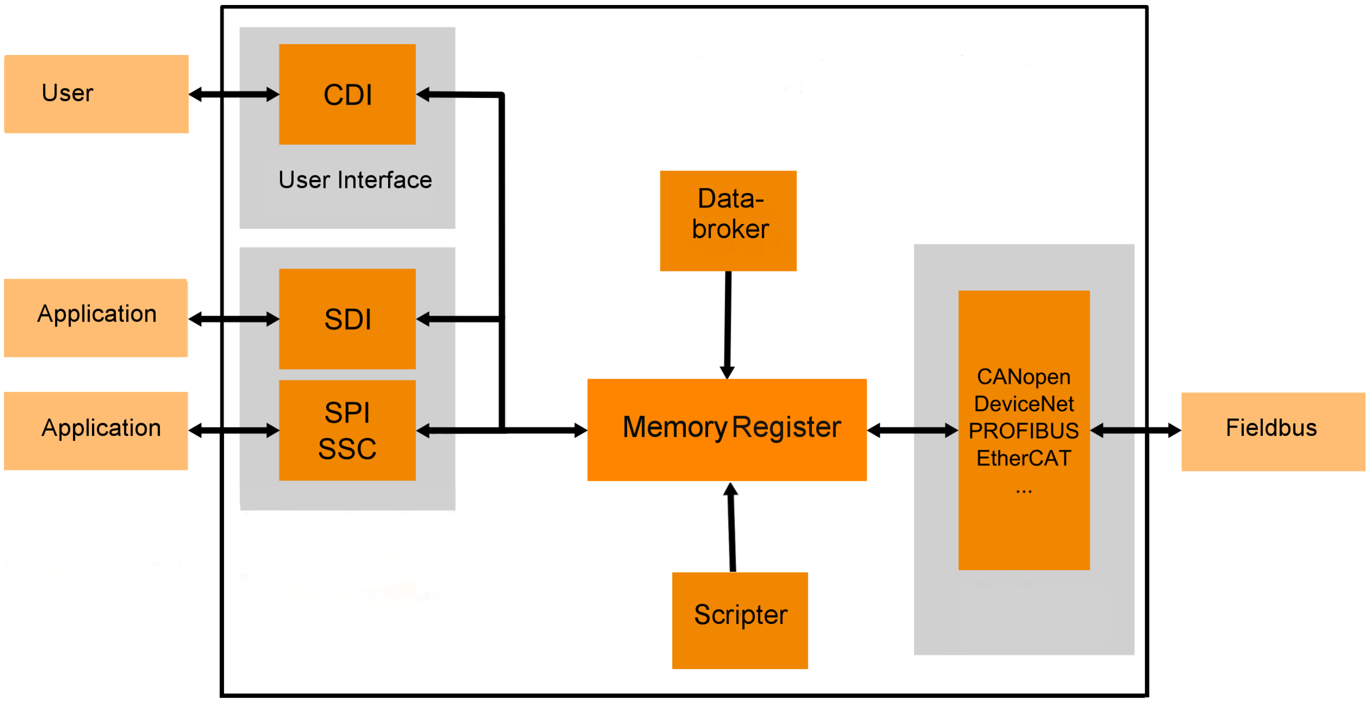


Illustration 1: Components

4.2 Storage Unit

The storage unit is the central component for all functions of the IC-Module. It is subdivided into individual Memory Register with a width of 16 bits each. In these Memory registers the following information is stored:

– Input and output data
– Configuration settings
– Module Status
– Error states

The functionality of the addressing was incorporated from Modbus. The register assignment depends on the application and is not specified by the Modbus specification. A Memory Register according to this specification has a register number between 1 (0x0001) and a maximum of 65536 (0x10000), of which the module only uses a small part, however.

With 8-bit values, 1 byte remains unused. 32-bit values are stored in 2 registers.

The 16-bit values are stored internally in the memory in Little Endian order. This must be taken into account when you access data via the fieldbus interface, SDI or SSC.

NOTICE! In the description of the individual memory registers and CDI, the memory registers are also referred to as Modbus registers.

In section Overview of the Memory Register [▸ 42] we have compiled a detailed overview of registers for you.

4.3 Data Broker

The Data Broker decouples individual components from each other and distributes the data streams between the interfaces. The targeted forwarding of the data ensures a high level of functionality between the data sources and data sinks of the module.

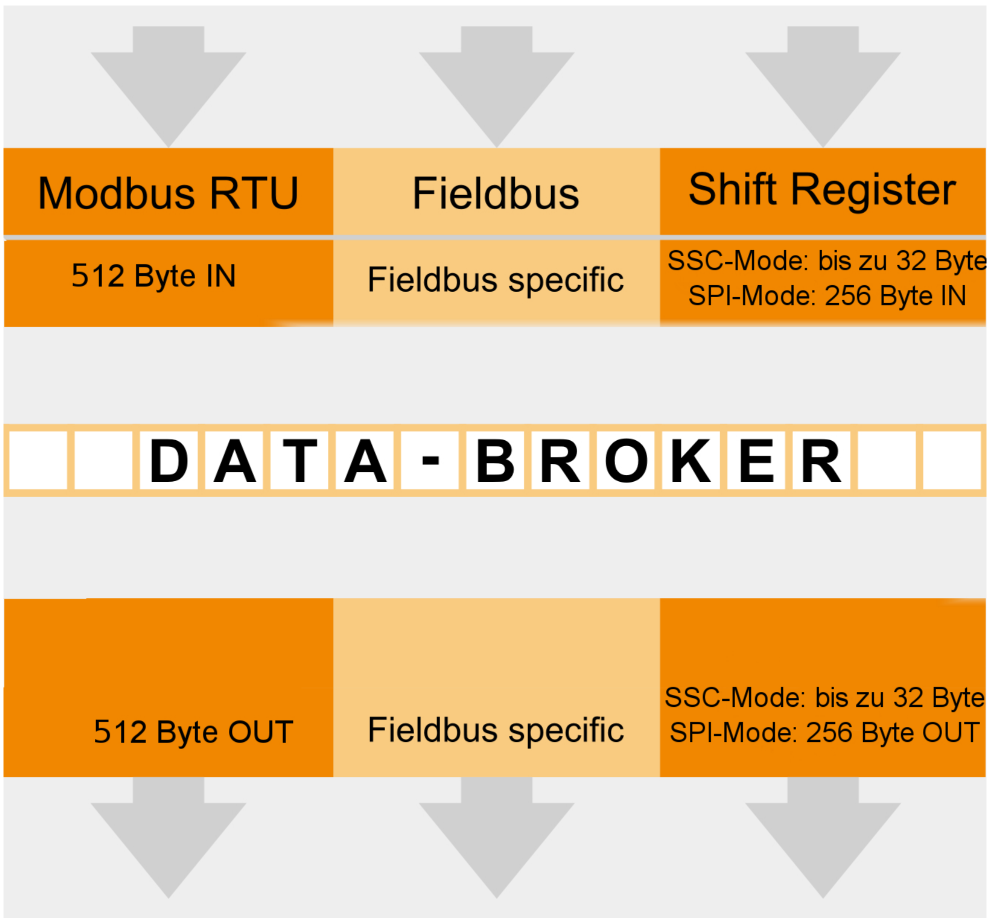


Illustration 2: Internal mapping by the Data Broker

Mapping

You have the option to define the allocation (mapping) yourself. This allows you to define which input register the Data Broker should accept data from and which output register it should transfer data to.

You can define up to 8 register areas with freely definable lengths in the respective output register area for each of the interfaces. Any register area of the same length is assigned to the output register areas from one of the input register areas of all interfaces. Here, the 8 target areas are always on consecutive output register positions, starting with the lowest register address for the respective interface.

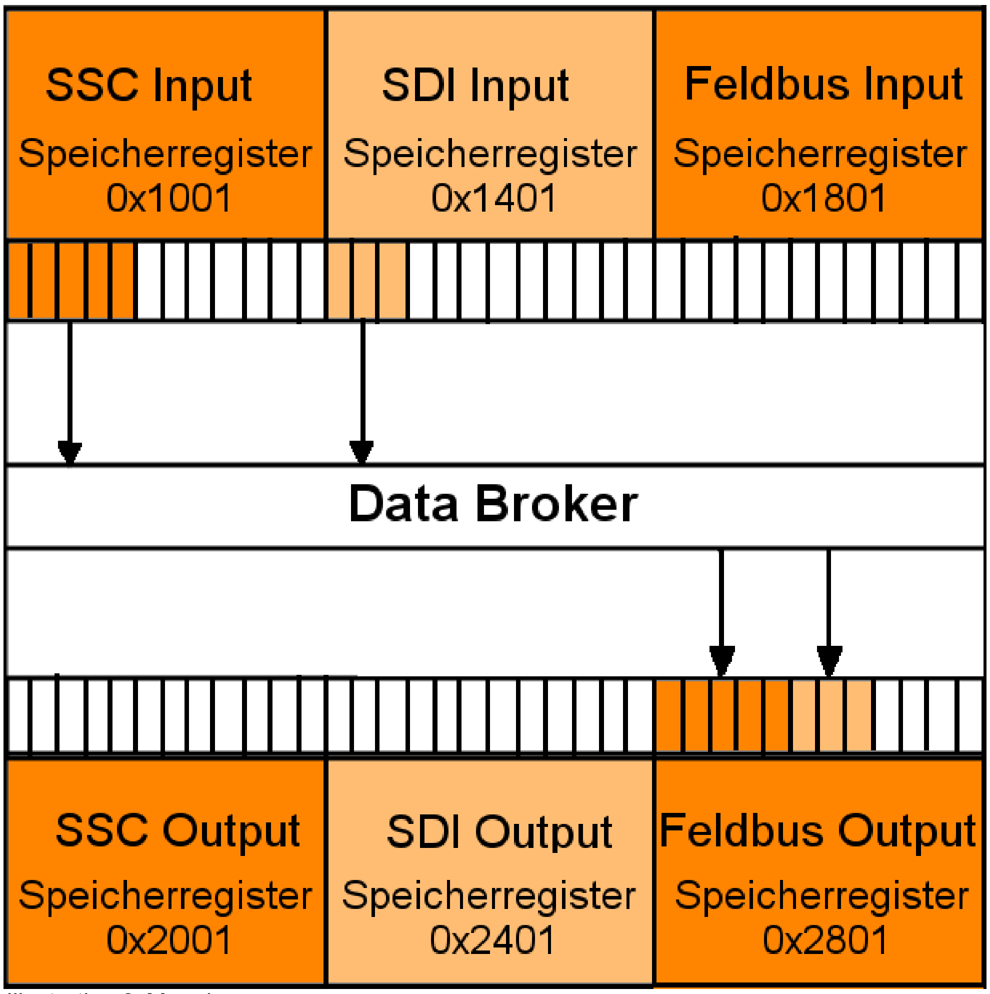


Illustration 3: Mapping

NOTICE

Viewpoint

Please note that the description of the input and output values is written from the perspective of the module and not from the perspective of the overall system or controller.

➔ Output: Values that the module sends to the fieldbus or application.

➔ Input: Values that the module receives from the fieldbus or application.

You can configure standard values that the Data Broker writes in the relevant output register instead of an input register in the event of a failure of a data provider. That has the advantage that the data processing cannot abort uncontrollably.

The module uses the Little Endian byte order for the internal processing. You can also configure the Data Broker so that it exchanges the high and low byte when copying if necessary. To do this, add the value 0x8000 or 32768 for the required mapping area (see details below).

Extended Mapping

Some applications work with data that is viewed bit by bit. To make the mapping for such applications even more flexible, an "Extended Mapping" is provided. This mapping basically works in the same way as the mapping described above:

Individual areas of the output Memory Register are assigned from areas of the input Memory registers. In Extended Mapping you define such assignments for up to 16 areas. When doing so, enter a number of consecutive bits for each of these areas.

The limit of a register must not be exceeded: The area may be up to 1024 bits long. Unlike the simple mapping described above, however, the 16 target areas do not necessarily have to be at consecutive addresses. You are totally free to define the position of the first bit of the target area by entering an output register address and the corresponding bit position (0 to 15). The source area is also defined by entering the input register and a start bit position.

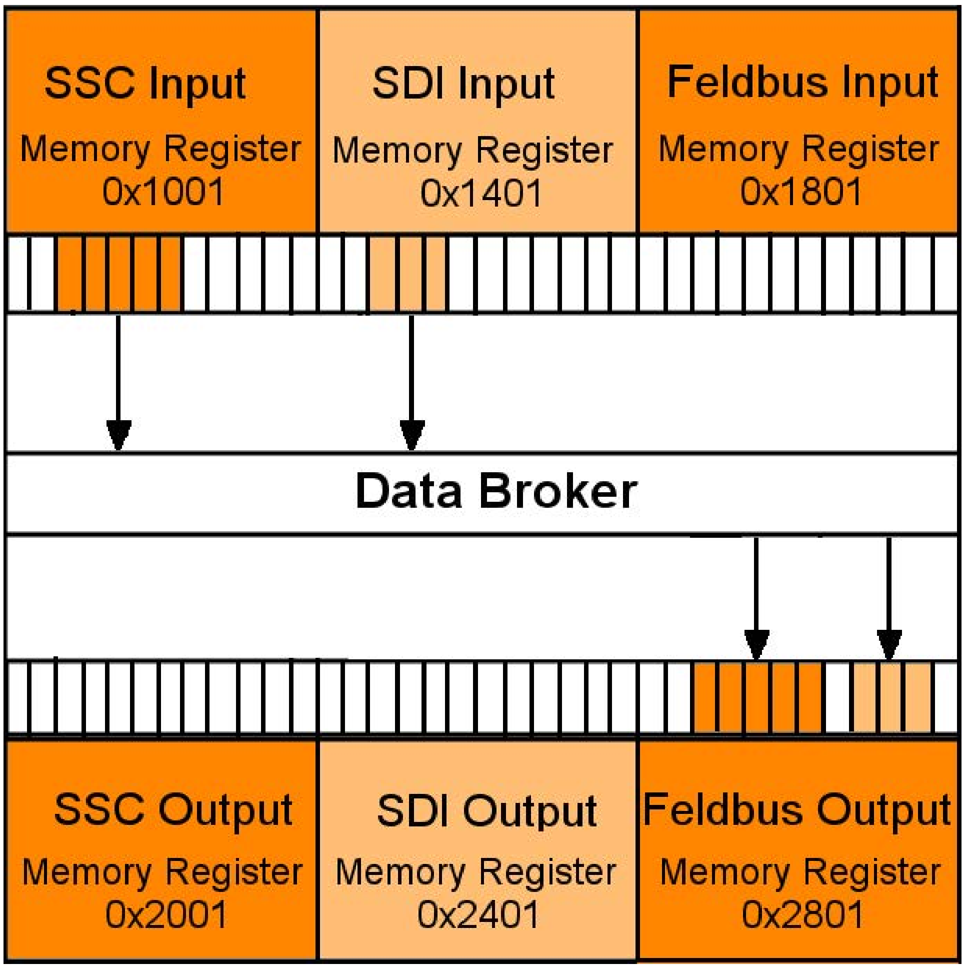


Illustration 4: Extended Mapping

All mapping areas are processed sequentially. The Data Broker first copies all simple mapping areas cyclically. After that, it executes the Extended Mapping. In the course of this, it is quite possible to intentionally overwrite a target area by several sources of data.

NOTICE

It is possible that bits are overwritten unintentionally by various input sources.

Make sure that the target areas do not overlap unintentionally.

Validity period of the process data

Data sources that write data to the input area of the central memory are called producers because they produce process data. The Data Broker collects this data and copies it into the output area of the central memory. From there, the data is sent to its target, the so-called consumer, via the corresponding interfaces.

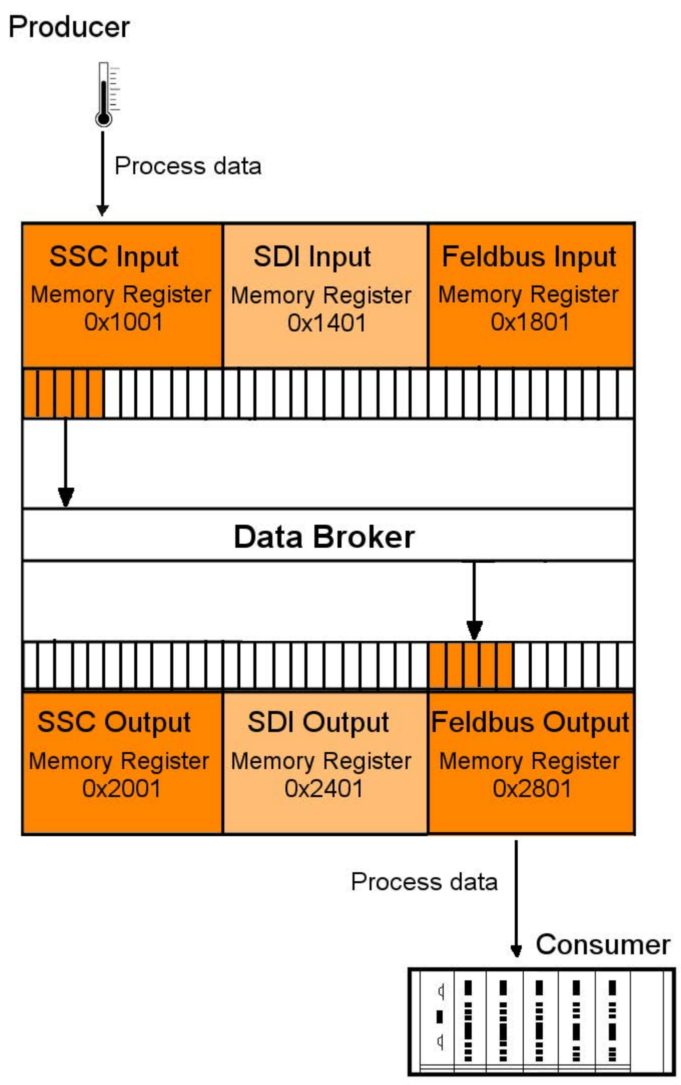


Illustration 5: Distribution to producers/consumers

Process data is normally exchanged cyclically between producers and consumers. If a producer fails (e.g. a connector is removed or a cable is broken), the consumer must be able to deal with this situation appropriately. For this reason, you can define in advance which values the producer will receive in exchange for the failed process data.

The IC-Module allow a separate validity period to be defined for each producer (SDI, SSC, CANopen). When a producer supplies new process data, a stopwatch is started. If the producer does not supply any new process data before the predefined validity period expires, then the old data is invalid after this time.

Each consumer predefines which data he is to receive from the Data Broker in such a case:

– All bytes at 0

– All bytes at 1

– retain the last valid data

The set validity periods from IC-Modul are saved permanently in the Memory registers. They are also available after a restart. Likewise, the rule defined for a consumer, as to how to proceed if the validity period is exceeded. The respective time values of the validity period must be adapted, of course, to the cycle time of the interface concerned.

– For CANopen this cycle time is determined by parameters of the master.

– For SSC interface the time is determined from the shift register chain length, the cycle frequency, and for short or fast register chains, it is determined by the cycle time of the ICmodule.

– During SPI slave mode and SDI transfer, the master determines the cycle time of the respective interface.

Example of a Mapping

The following example explains, step-by-step, how to map the first three SDI input registers and the first five SSC input registers to the field output register.

If you would like to participate in this example, you will need a functional CDI connection. Section "Setting up a Serial Connection [▶ 83]" explains how this works.

Input options in the CDI menu:

You can enter hexadecimal (with prefixed 0x) or decimal numbers in the CDI menu.

[Esc]	Go back one level
[Enter]	Confirm input/selection
[b]	Value is displayed in binary code
[h]	Value is displayed in hexadecimal code
[d]	Value is displayed in decimal code

◦ Open the main menu of the CDI as described in the Appendix "Setting up a serial connection using PuTTY".

Main Menu

The main menu is your access point for operating the module using the CDI. After a reset, the module transmits this main menu to the terminal.

```
--------------------------------------------------
KUNBUS-IC - Main Menu
--------------------------------------------------
1 - Module Information
2 - Interface Configuration
3 - Monitor Communication
4 - Module Status
--------------------------------------------------
>
```

Configuration menu

◦ In the main menu enter [2]+[Return].

⇨ You will be taken to the configuration menu "2-Interface Configuration"

In this menu you have the option to set the mapping for the data broker and the operational parameters for the different interfaces.

– Select "Fieldbus Output Mapping" to define the data source for the fieldbus output register.

```
--------------------------------------------------
KUNBUS-IC - Interface Configuration
--------------------------------------------------
1 - SDI Communication
2 - CDI Communication
3 - SSC Communication
4 - SDI Output mapping
5 - SSC Output mapping
6 - Fieldbus Output mapping
7 - Fieldbus Specific
8 - Set Arbitrary Register
9 - Reset Module
10 - Extended Databroker
11 - Script Interpreter
12 - Reset to Factory Settings
--------------------------------------------------
>
```

◦ Enter [1] + [Return].

◦ Specify the first 3 registers of the SDI input register as data source (start address 0x1401).

◦ Confirm your entry with [Return]

⇨ After confirming, you will return automatically to the "Fieldbus Output Mapping" menu

You can find an overview of the start addresses in the section "Overview of the Memory Register [▸ 42]".

```
--------------------------------------------------
KUNBUS-IC - Edit one map entry
--------------------------------------------------
Source Register: 0x1401
Number of Registers: 3
```

◦ Create another mapping at the next free position

◦ Select the first 5 registers of the SSC input register as data source (start address 0x1001)

```
--------------------------------------------------
KUNBUS-IC - Edit one map entry
--------------------------------------------------
Source Register: 0x1001
Number of Registers: 5
```

In the menu for fieldbus outputmapping, you can see the finished mapping.

```
--------------------------------------------------
KUNBUS-IC - Fieldbus Outputmapping
--------------------------------------------------
Src Register Number
1 - 1521 (0x1401) | 3
2 - 4097 (0x1001) | 5
3 - 1 (0x0001) | 0
4 - 1 (0x0001) | 0

5 - Default Data: all zero
6 - Valid Time: disabled
--------------------------------------------------
>
```

The new mapping becomes active after a restart of the module. To perform a restart, you have the following options:

1. Switch the module off and on again.
2. [Esc] takes you to the CDI menu [2] `"Interface Configuration"`. Here, enter [14] + [Return].

In CDI menu [2] `"Interface Configuration"` under menu item `"Set Arbitrary Register"` you now have the option to write the values in the SDI-In data area. The registers 0x1401 - 0x1500 are available to you for this purpose.

In menu [3] `"Monitor Communication"`, under menu item `" Arbitrary Register"` you can view the fieldbus output register from address 0x2801.

NOTICE

Fault due to fine settings

Some settings lead to malfunctioning of the module.

If you already want to test some settings now, read section CDI MenusCDI Menus [▹ 85].

Also see about this

- Register for the Mapping [▹ 67]
- Setting up a Serial Connection [▹ 83]

4.4 Fieldbus Interface

The fieldbus interface connects the Modul to CANopen. This also enables access to the fieldbus data areas FBS IN/Out.

Also see about this

🗎 Overview of the Memory Register [▸ 42]

4.5 CDI - Configuration and Debug Interface

At the application interface, serial cables are available (RS232 interface with 3.3 V logic levels). You can connect these cables to a terminal or PC with terminal simulation (e.g. PuTTY) using an interface IC on the main board (see the application sample circuit diagram, Appendix 2). You can read and change parameters using structured menus. The CDI is also used for downloading scripts and firmware updates.

The CDI is suitable for configuration during the development and for diagnostic purposes. To configure several modules automatically, we recommend performing the settings with "Modpoll". "Modpoll" is freely-available software. You can find an introduction and example of this in the Appendix Configuration via Modpoll [▸ 119].

The serial interface is located at the application interface. The respective cables are provided there with 3.3 V logic levels. To connect these cables, you have the following options:

– Connect the cables directly with the UART inputs of the microprocessor on the main board
– Convert the cables to standardised levels using level converters or interface ICs. Afterwards, place the converted levels onto connectors for connecting a PC or terminal.

We deliver the Modul to you with the following default settings to enable access via the CDI :

• 115200 bit/s

• 8 data bits

• 1 stop bit

• Even parity (Even)

In section CDI Menus, we have compiled a detailed description of the menus for you.

4.6 SDI - Serial Data Interface

The serial data interface allows the application to access the individual Memory Register via the Modbus-RTU protocol. This allows you to configure the IC-Modul automatically and to write productive data in the input registers or to read it from the output registers.

The serial data interface is located at the application interface. The respective cables are provided there with 3.3 V logic levels. To connect these cables, you have the following options:

– Connect the cables directly with the UART inputs of the microprocessor on the main board

– Convert the cables to standardised levels using level converters or interface ICs. Afterwards, place the converted levels onto connectors for connecting a PC or terminal.

We deliver the Modul to you with the following default settings to enable access via the SDI :

• automatic baudrate detection

• 8 data bits

• 1 stop bit

• Even parity (Even)

Automatic bitrate detection means that the module tests the following bitrates until it has received a correct Modbus-RTU telegram:

• 2400 bit/s

• 4800 bit/s

• 9600 bit/s

• 19200 bit/s

• 38400 bit/s

• 57600 bit/s

• 115200 bit/s

NOTICE! During automatic bitrate detection the module does not send a reply to the master until the correct bitrate has been detected. This procedure can require up to 40 polls of the master.

TIP: Set a fixed bitrate if the automatic bitrate detection lasts too long for you.

You can make the settings optionally using theCDI or in thememory register 0x0005 [▸ 48].

4.7 Synchronous serial interface

A synchronous serial interface is available to you on the application interface. The synchronous serial interface can be used in 2 operating modes. You can select the operating mode in the CDI menu or in the memory registers:

– CDI Menu 2.3 [▹ 89]
– Memory Register 0x0017 [▹ 54]

Output data from the Data Broker is written to the SSC output register area and input data is read from the SSC input register area in both operating modes. The SPI Slave operating mode also allows an SPI Master write and read access to all other Memory Register that are enabled for this. This section describes how this functions in detail.

Operating mode as SPI Slave

In SPI Slave mode, the transmission of the process data between an SPI Master and the SSC Input or Output registers takes place in data blocks, which, in addition to the actual process data, also contain metadata (e.g. for indicating the register addresses for source and target areas). Such data blocks are transmitted with a hardware handshake. The actual data transmission lines MOSI, MISO and Clock are used with 3.3 V logic in the usual manner, as described below in the document S12SPIV4 "SPI Block Guide" von Motorola / Freescale®. Here, you can freely select the normally alterable parameters CPOL (Clock polarity) and CPHA (Clock Phase) in IC-Modul and define these permanently via the CDI Menu [▹ 92] or memory register [▹ 55]. The bit sequence (MSB first or MSB last) is fixed for IC modules, the module always starts the transmission with the MSB (bit of highest value) of a byte. All bytes belonging to a block are transmitted in a continuous sequence.

The clock signal required is input externally from the Master.

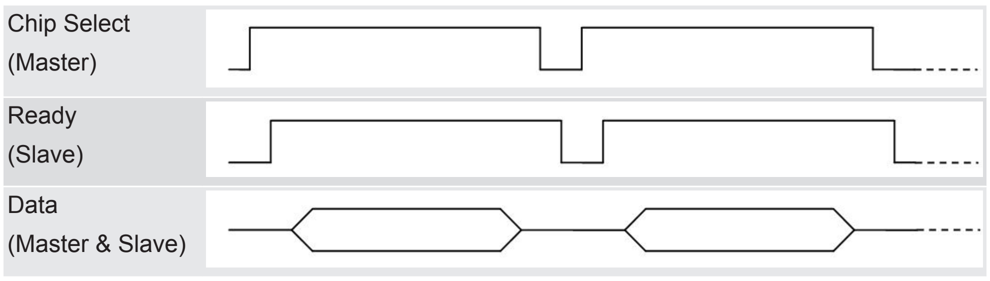
The IC-Modul can process maximum clock frequencies of 20 MHz.

Handshaking

The handshaking lines ensure that a Master first sends the subsequent transmission block after the module has processed the block that was received previously.

The module indicates by the "low" level on the SPI ready line that a transmission cycle has been completed, the status of the last transmission is waiting to be retrieved and the Master can trigger the next cycle. The Master starts this cycle by setting the SSC Chip

Select line to "high" to indicate to the module that data is ready for transmission and the following data block is meant for the module (theoretically, a master can address several modules). Once the module is now ready for this data transmission, it sets the SPI ready line to "high" and the Master can start transmission of the block immediately. A maximum delay between setting the CS signal and releasing by the ready signal of the module is 10 ms. All bytes of a data block are now transmitted directly in succession at the rate preset by the Master. After the last bit of the data block has been transmitted, the Master indicates the end of the transmission by resetting the SPI Chip Select line to "low". The module responds to this by resetting the SPI Ready line to "low". This happens at the earliest, however (maximum 10 ms after resetting CS), when the data has been processed insofar as the status was determined and is ready in the SPI output buffer so that the next transmission can start. This must first be requested, however, by the Master (as described above) by setting the SPI Chip Select line to "high".



Protocol

KUNBUS has defined a separate protocol for the data exchange via the synchronous serial interface. This protocol allows you to perform various read and write access operations. Here, the Master first always sends a transmission block with at least 5 bytes. The first 3 to 5 bytes of this transmission block consist of meta data (target address, etc.). Depending on the access type, another transmission block of variable data length follows the first block. Write and read access to the memory register of the module is performed. Only memory registers that have been enabled can be written or read, of course. The following areas cannot be written:

Input data areas:

– Fieldbus
– SDI

Output data areas:

– Fieldbus
– SSC

– SDI

When writing to the SSC input data area, the time monitoring is reset for this area (see Valid Time, Section "Data Broker [13]").

The various access types are explained below.

Writing 1 byte

This access type is used if 1 byte is written to a memory register of the module by the Master.

The Master first sends a transmission block with a fixed length of 5 bytes, which have the following content:

Transmission block with fixed length				
Command code (1 byte)	**Address area (2 bytes)**	**Data area (1 byte)**	**Mask area (1 byte)**	**Description**
0x01	0x0000-0xFFFF	0x00-0xFF	0x00-0xFF	WRITE_LOW_BYTE
0x02	0x0000-0xFFFF	0x00-0xFF	0x00-0xFF	WRITE_HIGH_BYTE
0x00	0xXXXX	0xXX	0xXX	NO_OPERATION*

Theoretically, you could use all memory register addresses between 0 and 0xFFFF. In practice, however, the write access is limited to registers that are enabled for this purpose. The byte can be written to the high or low byte position of the 16-bit wide register by selecting the associated command code. The mask byte only makes it possible to write single bits to the target register. Thereby, only bits that are set to "1" in the mask are transferred from the data byte (i.e. these bits are set to the value as found in the data byte). All other bits are left unchanged in the register.

During transmission of this first block, the module sends the status of the previous data transmission. The module first returns the status for the previous access when sending the next transmission block. If, however, no further write or read operation should follow the write access, then the Master must send another transmission block with the command code 0 ("NO_OPERATION") for retrieving the status, in which the module returns the status for the last write access operation.

The status response from the module is structured as follows for all write access types:

Transmission block with fixed length			
Status code (1 byte)	**Error code (2 bytes)**	**Not used (2 bytes)**	**Description**
0x00	0xXXXX	0xXXXX	NO_PREVIOUS_OPERA-TION
0x01	0x0000	0xXXXX	WRITE_SUCCESS
0x02	ERROR_CODE[1]	0xXXXX	WRITE_FAILURE

[1] See Table "ErrorCode"

The first byte returns the status. If it is set to "0", the Master then indicates that it cannot return any current status information since there was no previous operation (this is usually the response to the very first block transmission). A "1" indicates the successful completion of the previous transmission. In the case of a 2, the module sends the error code of an error in the subsequent byte, which occurred during the previous block transmission. The possible error codes are listed at the end of this subsection.

Writing 2 bytes (Word)

This access type basically proceeds as when writing 1 byte. It differs in the following points:

– Instead of a mask byte, the second byte of the 16-bit wide user data is transmitted with the data block. Access to individual bits in the target register is not possible with this access type.
– The 16-bit wide register content to be written must be prepared by the Master in such a way that the higher-value byte is transmitted as the 4th byte and the lower-value byte is transmitted as the 5th byte ("Big-Endian" or "Motorola format").

Transmission block with fixed length

Command code (1 byte)	**Address area (2 bytes)**	**Data area (2 byte)**	**Description**
0x04	0x0000-0xFFFF	0x0000-0xFFFF	WRITE_WORD
0x00	0xXXXX	0xXXXX	NO_OPERATION*

The status response has the same structure and meaning as write access with 1 byte

Writing more than 2 bytes with one access (bulk-write)

This access type is suitable for larger volumes of data. The number of target registers to be written and start address are transmitted with the first transmission block. As with the previous access types, the first transmission block also has a fixed length of 5 bytes here. After this block with metadata, the user data follows in a separate transmission block with variable length. The maximum permitted number of target registers to be written depends on the target area: A maximum of 128 registers (each 16-bit = 1 word) are permitted for writing to the SSC input register area. A maximum of 16 registers per block is to be written for all other target areas.

All 16-bit wide register contents to be written must be prepared by the Master in such a way that the higher-value byte is transmitted as the first byte and the lower-value byte is transmitted as the second byte ("Big-Endian" or "Motorola format"). The register contents must be sent in ascending address order, i.e. the start address first.

Transmission block with fixed length			
Command code (1 byte)	**Address area (2 bytes)**	**Data length (2 bytes)**	**Description**
0x08	0x0000-0xFFFF	1-16/128	WRITE_BULK
0x00	0xXXXX	0xXXXX	NO_OPERATION*

NOTICE

The maximum data length for writing in the SSC input data area is 128 registers (256 bytes).

If this value is exceeded, errors in the data communication will result.

In the case of a transmission block of variable length, the module sends bytes with the value 0 to the master.

The status response has almost the same structure and meaning as write access with 1 byte. In the event of an error, a 16-bit wide register address is at position 4 and 5 for this transmission type, at which the first error occurred. The status is transmitted in the first transmission block that follows the data block with variable length.

Transmission block with fixed length			
Status code (1 byte)	**Error code (2 bytes)**	**Address area** (2 bytes)**	**Description**
0x00	0xXXXX	0xXXXX	NO_PREVIOUS_OPERATION
0x01	0x0000	0xXXXX	WRITE_SUCCESS
0x02	ERROR_CODE[1]	0x0000-0xFFFF	WRITE_FAILURE

** Address where an error occurs

Reading 2 bytes (Word)

This access type is used if just 1 register is to be read from a memory register of the module by the Master. The Master first sends a data block with a fixed length of 5 bytes, which have the following content:

Transmission block with fixed length			
Command code (1 byte)	**Address area (2 bytes)**	**Not used (2 bytes)**	**Description**
0x10	0x0000-0xFFFF	0xXXXX	READ_WORD
0x00	0xXXXX	0xXXXX	NO_OPERATION*

* This command allows the master to request the status of a read request without an additional read or write request having to be executed.

Theoretically, you could use all memory register addresses between 0 and 0xFFFF. In practice, however, the write access is limited to registers that are enabled for this purpose.

During transmission of the first data block, the module sends the status of the previous data transmission. The module first returns the data to be read when sending the next data block. If, however, no further write or read operation should follow the read access, then the Master must send another data block with the command code 0 ("NO_OPERATION") for retrieving the data to be read, in which the module returns the status for the last write access operation.

The response from the module is structured as follows for all read access operations:

Transmission block with fixed length			
Status code (1 byte)	**Error code (2 bytes)**	**Data area (2 byte)**	**Description**
0x00	0xXXXX	0xXXXX	NO_PREVIOUS_OPERATION
0x01	0x0000	0x0000-0xFFFF	READ_SUCCESS
0x02	ERROR_CODE[1]	0xXXXX	READ_FAILURE

[1] See Table "ErrorCode"

The first byte returns the status. If it is set to "0", the Master then indicates that it cannot return any current status information since there was no previous operation (this is usually the response to the very first block transmission). A "1" indicates the successful completion of the previous transmission. In the case of a 2, the module sends the error code of an error in the subsequent byte, which occurred during the previous block transmission.

If the status is "1", the 2 bytes after that at position 4 and 5 contain the content of the memory register to be read at the address that was transmitted at the last block with the read command. The 16-bit wide register content read is prepared by the module in such a way that the higher-value byte is transmitted as the 4th byte and the lower-value byte is transmitted as the 5th byte ("Big-Endian" or "Motorola format").

In the case of status "0" or "2", both data bytes at position 4 and 5 are invalid and must be discarded by the Master.

Reading more than 2 bytes (Bulk-Read)

In this access type, the number of source registers to be read as well as the start address are transmitted with the first transmission block that has a fixed length of 5 bytes. After this block with metadata, the transmission of the read data follows in a separate transmission block with variable length. Therefore, this access type is suitable primarily for larger volumes of data. The maximum permitted number of source registers to be read depends on the source area: A maximum of 128 registers (each 16-bit = 1 word) are permitted for reading from the SSC output register area. A maximum of 16 registers per block is to be read for all other source areas. Byte order:

All 16-bit wide register contents read are prepared by the module in such a way that the higher-value byte is transmitted as the first byte and the lower-value byte is transmitted as the second byte ("Big-Endian" or "Motorola format"). The register contents are sent in ascending address order, i.e. the start address first.

Transmission block with fixed length			
Command code (1 byte)	**Address area (2 bytes)**	**Data length (2 bytes)**	**Description**
0x20	0x0000-0xFFFF	1-16/128/256	READ_BULK
0x00	0xXXXX	0xXXXX	NO_OPERATION*

The Master sends a block of variable length with 0 bytes to the module.

The status response has almost the same structure and meaning as read access with 1 byte. In the event of an error, a 16-bit wide register address is at position 4 and 5 for this transmission type, at which the first error occurred. The status is transmitted in the first transmission block that follows the data block with variable length.

In the event of an error during bulk access (status "2"), the data transmitted by the module from the data block with variable length is invalid and must be discarded by the Master.

Transmission block with fixed length			
Status code (1 byte)	**Error code (2 bytes)**	**Address area (2 bytes)**	**Description**
0x00	0xXXXX	0xXXXX	NO_PREVIOUS_OPERATION
0x01	0x0000	0xXXXX	READ_SUCCESS
0x02	ERROR_CODE[1]	0x0000-0xFFFF	READ_FAILURE

[1]See Table “ Error code“

** Address where an error occurs

Transmission block with variable length (1-16/128 words)
Data area
0x0000-0xFFFF"

Simultaneous reading and writing of more than 2 bytes with one access (bulk read/ write)

In this access type, the number of source registers to be read or target registers to be written are transmitted with the first transmission block that has a fixed length of 5 bytes. After this block with metadata, the transmission of the read data follows in a separate transmission block with variable length. Unlike with Bulk-Read or Bulk-Write, no random start address can be defined for this access type. The start address for the block to be read is preset with 0x2001 (SSC input register) and with 0x1001 (SSC output register) for the block to be written. Byte order:

All 16-bit wide register contents to be read or written are prepared by the module in such a way that the higher-value byte is transmitted as the first byte and the lower-value byte is transmitted as the second byte ("Big-Endian" or "Motorola format"). The register contents are sent in ascending address order, i.e. the start address first.

Transmission block with fixed length			
Command code (1 byte)	**Not used (2 bytes)**	**Data length (2 bytes)**	**Description**
0x40	0xXXXX	1-128	READ_WRITE_BULK
0x00	0xXXXX	0xXXXX	NO_OPERATION*

Transmission block with variable length (1-16/128 words)
Data area
0x0000-0xFFFF"

The status response has almost the same structure and meaning as read access with 1 byte. In the event of an error, a 16-bit wide register address is at position 4 and 5 for this transmission type, at which the first error occurred when reading or writing. The status is transmitted in the first transmission block that follows the data block with variable length.

Transmission block with fixed length			
Status code (1 byte)	**Error code (2 bytes)**	**Not used (2 bytes)**	**Description**
0x00	0xXXXX	0xXXXX	NO_PREVIOUS_OPERATION
0x10	0x0000	0xXXXX	READ_WRITE_SUCCESS
0x20	ERROR_CODE[1]	0xXXXX	READ_WRITE_FAILURE

In the event of an error during bulk access (status "2"), the data transmitted by the module from the data block with variable length is invalid and must be discarded by the Master.

Error Codes

Error code	Designation	Description
0x01	INVALID_DATA_ ADDRESS	Invalid data address The master tries to access an invalid ad-dress. The slave ignores the instruction.
0x02	INVALID_DATA_ LENGTH	Invalid data length The data length predefined by the master is too great. The slave ignores the instruction.
0x04	INVALID_DATA	Invalid data The master tries to write data containing values outside a valid range. The slave ignores the instruction.
0x08	INVALID_ACCESS	Invalid access The master tries to access an invalid area or a valid address. The slave ignores the instruction.
0x10	INVALID_RANGE	Invalid range The master tries to write beyond the limits of an SSC input data area or to write bey-ond the limits of an SSC, SDI or FBS output data area. The slave ignores the instruction.
0x20	UNDEFINED_ERROR	Undefined error An undefined error has occurred. The slave ignores the instruction.

Table 1: Error Code

SSC Master Operating Mode

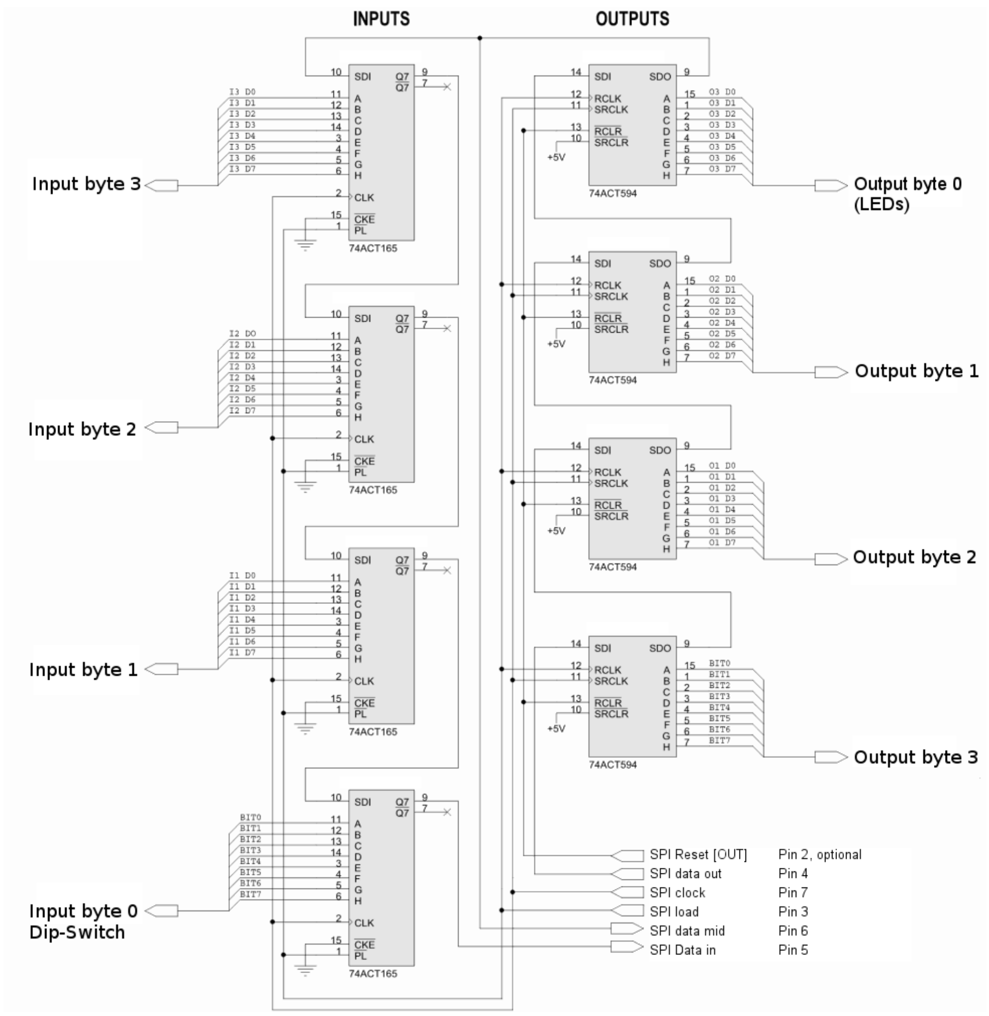


Illustration 6: Hardware shift register chain, example with 4 inputs and outputs

The advantage of such an interface is the possibility of forwarding input and output signals to the fieldbus without the need of microprocessor controlled application circuitry. Switches, contacts, relay coils or solenoid valves, for example, can therefore be connected directly via CANopen without using a microprocessor.

The IC-Modul with its clock clocks the output data into the input register of the chain via the MOSI line, where it is shifted bit by bit until the end. At the same time, the input data is shifted bit by bit via the MISO line into the IC-Modul with the same clock pulse. Prior to each such shift procedure, the module sets the LOAD line to high. In this way, the parallel outputs of all shift register modules receive the data from the input buffers in the previous cycle. The input shift registers, on the other hand, utilise the positive edges from the LOAD signal to copy all parallel input values simultaneously to their output buffers. From there, they are shifted bit by bit to the SSC input register area of the IC-Moduldruring the current cycle.

The clock rates of the IC-Moduls can be adapted manually or automatically in 3 levels and are about 300, 1200 or 4800 Kbit/s. The load impulse is between 5 and 15 µs long (active low). The delay between the load edge (positive edge of the load impulse) and the first clock edge (from high to low) is between 1 and 2 µs. These values are completely uncritical when using the shift register modules 74HC165 (Input) and 74HC594 (Output).

An optional SSC RESET line initialises the shift register modules during the starting process of the IC-Moduls (i.e. also during each reset of the module).

With an arrangement of the output and input shift register as shown in this example, all registers are switched in series so that the IC-Modulhas its own output data shifted back into the input register again for checking purposes. A test sample shifted through the complete chain without a LOAD signal allows the IC-Modulto detect how long the entire chain is by means of the necessary clock signals for such a shifting procedure. A centre pickoff between the output and input modules allows the IC-Modul to also detect the corresponding number of inputs and outputs of the shift register modules during this run of a test sample. If bit errors occur, the clock rate is reduced in automatic mode by one level. Hence, with such a structure the IC-Modulcan find the right setting for the chain lengths and maximum possible transmission rate independently. It is also possible, however, to assign the lengths and clock rates manually via the CDI menu. In this case, the centre pickoff can also be omitted (it is only needed for determining the allocation between inputs and outputs of the shift register modules). The entire chain length is monitored constantly during ongoing operation and must match the configured length. If the module detects a difference, then it shuts down the SSC communication and reports an error status via its status register.

The IC-Modulcan operate a maximum of 32 shift registers. You can use them as output or/and input shift registers.

Note on cycle time: The cycle time of the shift register interface is normally independent of its chain length since the IC-Modulin its work cycle only starts the transmission of a shift procedure. The shift procedure itself then takes place independently of the work cycle of the IC-Moduls. Its length is determined by the number of cycles as well as the clock rate. After completion of a shift procedure, the next shift cycle starts with the next work cycle of the module. The maximum delay between the completion and start of a shift cycle is 10 ms.

Note ! If the shift procedure is longer than a work cycle of the module, the cycle time is determined by the length and speed of the shift register chain.

4.8 Scripter

The IC-Modul includes a software component that allows you to set up customer-specific data exchange protocols for SDI or CDI serial interfaces. If, for example, the module is to communicate in an application with a serial-controlled servomotor, this servomotor expects a preset protocol to be processed in order to receive the actuator values or to return sensor values. With the aid of the Scripter you can load small executable program sequences into the module that are then executed there cyclically. With the appropriate data exchange protocol the module can receive such actuator values e.g. via CANopen and transmit these via the serial interface of the module (SDI or CDI) to the servomotor. The program sequences required are loaded once into the module in the form of a script via the CDI interface of the module and then always executed there cyclically. KUNBUS provides you with a PC tool for creating and testing such scripts. You can read all the necessary details in the separate manual on the Scripter.

NOTICE! Please note that when using the Scripter and activating a script the interface (CDI or SDI) selected for its communication is always assigned for the Scripter. If you choose the CDI interface, you can then no longer use this interface to check and enter module parameters ("CDI menus" are then no longer available). If you choose the SDI interface as a serial communication channel for the Scripter, you can then no longer process any Modbus protocol with access to the memory register via this interface.

5 Commissioning

5.1 Installation

The main board is connected to the device controller via a 32-pin connector strip. Thus, you have the option to plug the module directly into your DIL socket.

NOTICE

If the module is plugged in and unplugged frequently, mechanical stresses may damage the module.

Use a zero insertion force socket to prevent damage.

Pin assignment on the application interface

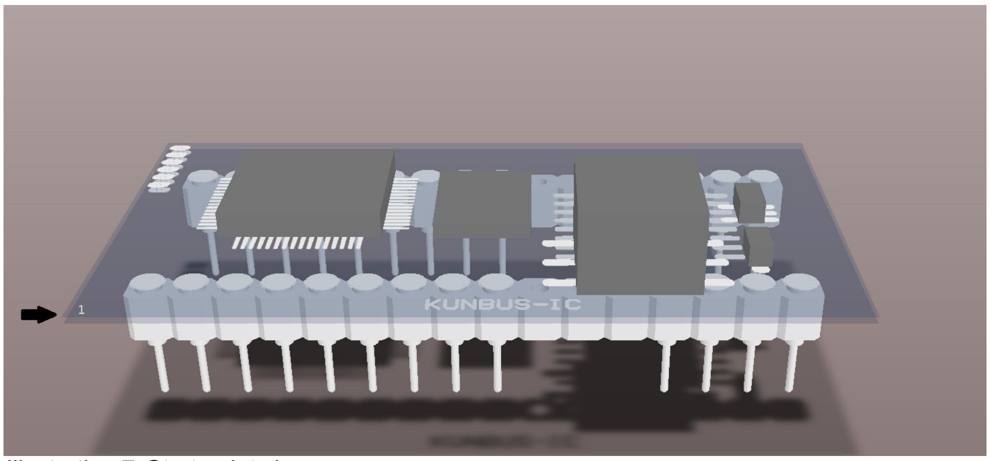


Illustration 7: Start point view

When you view your module from above, you will find a printed triangle in one corner.

The counting of the pins starts with the underlying pin which is then continued U-shaped and ends with the allocation 32 at the opposite pin.

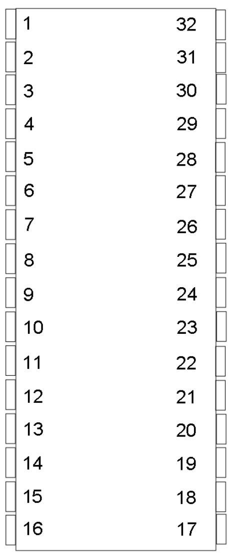


Illustration 8: Pin Assignment

In the table below, we have compiled the pin assignment on the application interface for you.

Master Mode

Pin	Application side	Direction	Electrical specifications			Temperature range (-40 C to 85 C)
			Min.	Typical	Max.	Tolerance:
1[(1)]	VCC 5.0 V	[IN]	4.75 V	5.0 V	5.25 V	Imax = 150 mA
	VCC 3.3 V	[IN]	3.15 V	3.3 V	3.45 V	Imax = 200 mA
2	SPI Reset	[OUT]	-	-	3.3 V, 4 mA	
3	SPI Load	[OUT]	-	-	3.3 V, 4 mA	
4	SPI data out	[OUT]	-	-	3.3 V, 4 mA	
5	SPI data in	[IN]	-0.1 V	2.0 V [(2)]	5 V	
6	SPI data mid	[IN]	-0.1 V	2.0 V [(2)]	5 V	
7	SPI clock	[OUT]	-	-	3.3 V, 4 mA	
8	Module Reset	[IN]	-0.1 V	-	5 V	
9	VC DC Converter 5.0 V	[IN]	4.75 V	5.0 V	5.25 V	
	VC DC Converter 3.3 V	[IN]	3.15 V	3.3 V	3.45 V	
10-12	Reserved			Do not connect!		
13	CAN L	[IN/OUT]	(3)	(3)	(3)	
14	CAN H	[IN/OUT]	(3)	(3)	(3)	
15-18	Reserved			Do not connect!		
19	GND (isolated for bus connection)	-	-	-	-	
20	+ 5 V (isolated for bus connection)	-	-	-	-	
21-23	Reserved			Do not connect!		
24	GND	[IN]	-	-	-	
25	LED Module Status	[OUT]	-	-	3.3 V, 4 mA	
26	Reserved	[IN]	-	-	-	
27	CDI (UART) TX	[OUT]	-	-	3.3 V, 4 mA	
28	CDI (UART) RX	[IN]	-0.1 V	2 V [(2)]	5 V	
29	SDI (UART) RX	[IN]	-0.1 V	2 V [(2)]	5 V	
30	SDI (UART) TX	[OUT]	-	-	3.3 V, 4 mA	
31	RS-485 TX en	[OUT]	-	-	3.3 V, 4 mA	
32	GND	[IN]	-	-	-	

Table 2: SSC Master Mode

Slave Mode (Activation via Modbus Register)

Pin	Application side	Direction	Electrical specifications			Temperature range (-40 C to 85 C)
			Min.	Typical	Max.	Tolerance:
1[(1)]	VCC 5 V	[IN]	4.75 V	5.0 V	5.25 V	Imax = 150 mA
	VCC 3.3 V	[IN]	3.15 V	3.3 V	3.45 V	Imax = 200 mA
2	Reserved				Do not connect!	
3	SPI Slave Ready	[OUT]	-	-	3.3 V, 4 mA	
4	SPI data in	[IN]	-0.1 V	2 V [(2)]	5 V	
5	SPI data out	[OUT]	-	-	3.3 V, 4 mA	
6	SPI Slave Select	[IN]	-0.1 V	2 V [(2)]	5 V	
7	SPI clock	[IN]	-0.1 V	2 V [(2)]	5 V	
8	Module Reset	[IN]	-0.1 V	-	5 V	
9	VC DC Converter 5.0 V	[IN]	4.75 V	5.0 V	5.25 V	
	VC DC Converter 3.3 V	[IN]	3.15 V	3.3 V	3.45 V	
10-12	Reserved				Do not connect!	
13	CAN L	[IN/OUT]	(3)	(3)	(3)	
14	CAN H	[IN/OUT]	(3)	(3)	(3)	
15-18	Do not connect!				Do not connect!	
19	GND (isolated)	-	-	-	-	
20	+ 5 V (isolated)	-	-	-	-	
21-23	Reserved				Do not connect!	
24	GND	[IN]	-	-	-	
25	LED Module Status	[OUT]	-	-	3.3 V, 4 mA	
26	Reserved	[IN]	-0.1 V	2 V [(2)]	5 V	
27	CDI (UART) TX	[OUT]	-	-	3.3 V, 4 mA	
28	CDI (UART) RX	[IN]	-0.1 V	2 V [(2)]	5 V	
29	SDI (UART) RX	[IN]	-0.1 V	2 V [(2)]	5 V	
30	SDI (UART) TX	[OUT]	-	-	3.3 V, 4 mA	
31	RS-485 TX en	[OUT]	-	-	3.3 V, 4 mA	
32	GND	[IN]	-	-	-	

Table 3: SPI Slave Mode

(1) The module is available optionally with 5 volt or 3.3 volt.

(2) A logical High is detected from 2 Volts.

(3) According to ISO1189-2

Function

The individual pins establish the contact to the following functions:

- Pin 1 and 9: Power supply.
- Pin 2 – 7: SPI Slave interface.
- Pin 8: Module reset
- Pin 13 – 15 and 19 - 20: Fieldbus
- Pin 27 – 31: Serial communication (SDI/CDI)

NOTICE

An example of the connection options can be found in Appendix 2.

Connection options to the SDI interface

You can establish the connection to the SDI in RS232 or RS485 mode:

You need an RS232 level converter for operating in RS232 mode.

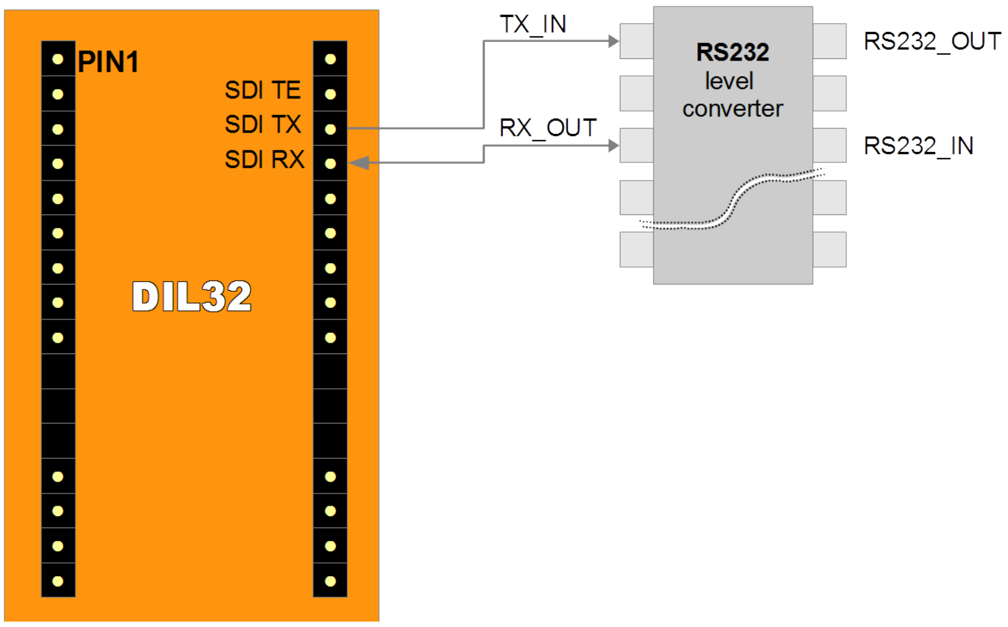


Illustration 9: Connection for the SDI via RS 232

The TX PIN is not used with this connection.

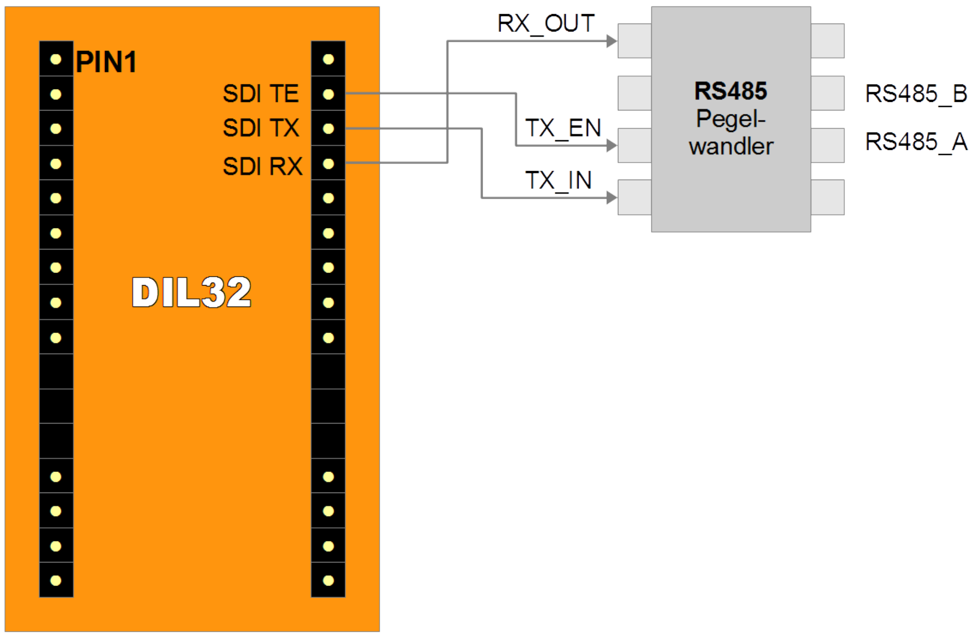


Illustration 10: Connection option for the SDI using RS485

You need an RS485 level converter for operating in RS485 mode.

Connection options to the CDI interface

You can establish the connection to the CDI in RS232 mode:

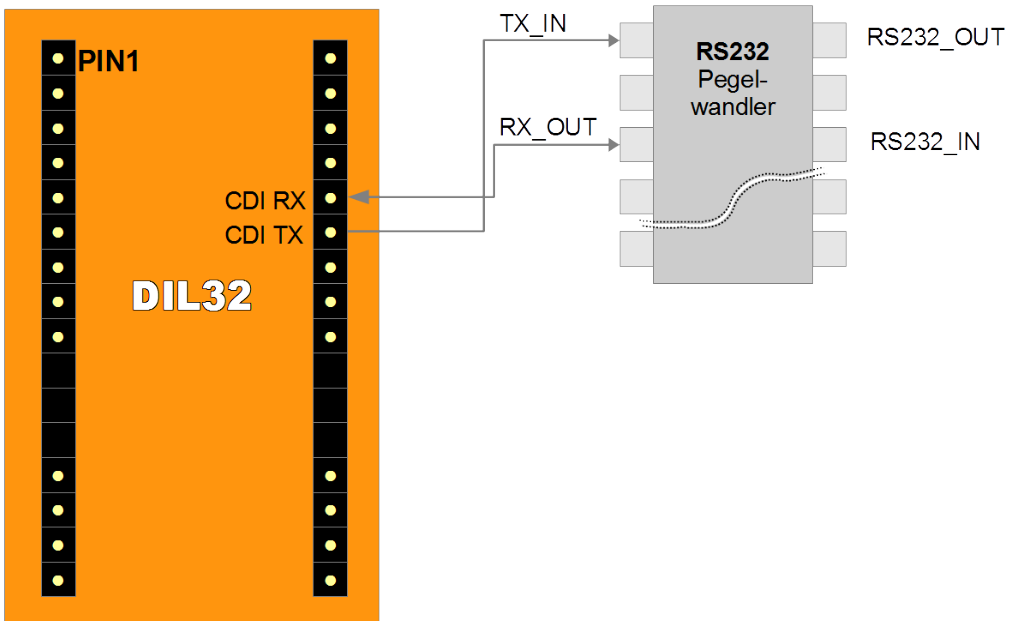


Illustration 11: Connection option for the CDI

5.2 Configuration

This section describes how to configure the module and associated components and applications.

NOTICE

- ➢ The module has no undo function.
- ➔ Changes are applied after a reset or start of the operating mode without any further confirmation.
- ⇨ If you want to reset all values, use the function "Reset to factory settings". [▸ 106] Please note that all previous settings made will be lost.

Configuration using the CDI

To configure the Modul and put it into operation using the CDI, you need a PC or notebook with a serial interface (RS-232) or USB/serial adapter. Make sure that the adapter drivers are installed.

Communication with the CDI (Configuration and Debug Interface) of the module takes place using a terminal program (e.g. PuTTY for Microsoft Windows®).

TIPP!: Das CDI is suitable for configuration during the development and for diagnostic purposes. To configure several modules automatically, we recommend performing the settings with Modpoll. You can find an introduction and example of this in the Appendix Configuration via Modpoll [▸ 119].

Configuration using the SDI

The IC-Modul has a UART interface with 3.3 V logic levels. Your main board must convert these lines to standardised RS-485 signals so that Modbus/RTU devices can access these. Conversion to standardised RS-232 signals is generally necessary for communication with a PC. The base board of the evaluation board has both interfaces that can each be selected via jumpers.

Configuration using the SDI requires a Modbus master device.

One of the following devices is suitable for this:

– Master computer,
– Control panel,
– Programming device,
– SPS with the possibility of Modbus-RTU communication.

To communicate with the SDI of the module using a PC, you need Modbus software (e.g. Modpoll).

5.3 Firmware Update

If a firmware update is required, please contact our support (support@kunbus.de). We will be delighted to provide you with all the information you need for your product.

6 Memory Register

6.1 Overview of the Memory Register

The storage unit is the central component for all functions of the IC-Module. It is subdivided into individual Memory Register with a width of 16 bits each. In these Memory registers the following information is stored:

- Input and output data
- Configuration settings
- Module Status
- Error states

The functionality of the addressing was incorporated from Modbus. The register assignment depends on the application and is not specified by the Modbus specification. A Memory Register according to this specification has a register number between 1 (0x0001) and a maximum of 65536 (0x10000), of which the module only uses a small part, however.

With 8-bit values, 1 byte remains unused. 32-bit values are stored in 2 registers.

The 16-bit values are stored internally in the memory in Little Endian order. This must be taken into account when you access data via the fieldbus interface, SDI or SSC.

NOTICE! In the description of the individual memory registers and CDI, the memory registers are also referred to as Modbus registers.

Bitwise access to input and output data

Optionally, you can address input and output data areas bitwise. The functions 01 Read Coil Status, 02 Read Input Status and 05 Force Single Coil are defined in Modbus for this purpose. Since each bit has a separate address, they are assigned to the bits in the registers as follows: Coil 0x0001 corresponds to the lowest value bit 0 of register 0x0001, Coil 0x0002 corresponds to bit 1, etc. coil 0x11 is the bit 0 from register 0x0001 etc.

The table below shows the start addresses of the data areas:

Area	Memory Register	Coil/Input Address
Input SSC	0x1001 - 0x1080	0x0001 – 0x0800
Input SDI	0x1401 – 0x1500	0x2001 – 0x4000
Input DPR	0x1c01 …	0x6001 …
Output SSC	0x2001 – 0x2080	0x8001 – 0x8800
Output SDI	0x2401 – 0x2500	0xa001 – 0xb000
Output FBS	0x2801 - 0x2880	0xc001 – 0xe001

Register assignment of the memory area

The following table contains a brief overview of the register assignment of the general memory area. You can find a detailed overview of the individual registers on the following pages.

Register number	**Assignment**	**Description**
0x0001 – 0x0100 [▹ 45]	General Device Parameters	e.g. Setting of the bitrates, mailbox sizes etc.
0x0101 – 0x0e00	Reserved	-
0x0e01 – 0x0ea0	Register for mapping the output data	Each channel occupies 2 x 8 registers
0x0f01 – 0x0f40 [▹ 68]	Register for mapping of the extended Data Broker	16 mappings occupy 4 registers each
0x1001 – 0x2000 [▹ 69]	Input memory of the communication channels	Each communication channel has a preallocated memory area of 128 - 256 registers.
0x2001 – 0x3000 [▹ 70]	Output memory of the communication channels	Each communication channel has a preallocated memory area of 128 - 256 registers.
0x3001 – 0x4000	Reserved	-
0x4001 – 0x5000	Fieldbus-specific (s. following table)	See the description of the individual fieldbus variants
0x5001 – 0x10000	Reserved	-

The following table contains a brief overview of the register assignment of the memory area for CANopen. You can find a detailed overview of the individual registers on the following pages.

Register	Designation	Register	Designation
0x4001 [▹ 71]	Fieldbus Status	0x4002 [▹ 71]	Module Status
0x4003 [▹ 71]	Fieldbus Vendor ID, High Byte	0x4004 [▹ 71]	Fieldbus Vendor ID, Low Byte
0x4005	Reserved	0x4006	Reserved
0x4007 [▹ 72]	Fieldbus Version, High Byte	0x4008 [▹ 72]	Fieldbus Version, Low Byte
0x4009 [▹ 72]	Firmware Version	0x400a [▹ 72]	Serial number, High Byte
0x400b [▹ 72]	Serial number, Low Byte	0x400c [▹ 73]	current Fieldbus Node ID
0x400d	Configured Field-bus Address	0x400e	SSC Node ID
0x400f [▹ 73]	current Fieldbus Bitrate	0x4010 [▹ 74]	Configured Field-bus Bitrate
0x4011 [▹ 74]	Fieldbus Config-uration Bits, High Word	0x4012 [▹ 74]	Fieldbus Config-uration Bits, Low Word
0x4013	Reserved	0x4014 [▹ 76]	Product Code, High Word
0x4015 [▹ 76]	Product Code, Low Word	0x4016-0x4035 [▹ 77]	Product Name
0x4036 [▹ 77]	Size of input im-age	0x4037 [▹ 77]	Size of output image
0x4101 [▹ 78]	LSS Status	-	-

6.2 General Device Parameters

0x0001 Set operating mode

In this memory register you have the option, the operating mode to set

Modbus Register	0x0001
Value Range	0x0000-0x0003
Default Value	0x0000
Number of bytes available	2
Permanently stored	No
Access	Read/Write
Meaning	
0x0000 or 0x0001	Operation Cyclical data exchange takes place.
0x0002	Restoring default settings (Factory Reset) Resetting of all permanent parameters to their original respective settings. A module reset takes place automatically and does not have to be done manually here.
0x0003	Reset Implementing a reset. Your settings can first be applied after a reset.

0x0002-0x0003 Current module status

In these memory registers you will find information for the current module status.

Bit 5 indicates whether there is an error in the configuration of the SSC Master mode. It is only set, however, during the initialisation of the module. If an error occurs during ongoing operation, this is not displayed here.

Memory Register 0x0002 (bit 0-15) contains the Low Word, Memory Register 0x0003 (bit 16-31) contains the High Word.

Modbus Register	0x0002-0x0003
Value Range	-
Initial value	-
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
Bit 0	Fieldbus Run State 1: The field bus is in cyclical data exchange 0: The cyclical data connection is interrupted

Bit 1	SSC SSR Master Run State 1: The synchronous serial interface is in SSC mode and is exchanging data cyclically 0: No cyclical data exchange takes place.
Bit 2	SSC Mapping Configuration Error State 1: Configuration error in the mapping of the SCC interface. 0: Configuration is ok.
Bit 3	SDI Mapping Configuration Error 1: Configuration error in the mapping for the SDI. 0: Configuration is ok.
Bit 4	Fieldbus Communication Mapping Configuration Error 1: Configuration Error in the Mapping for the Fieldbus Interface. 0: Configuration is ok.
Bit 5	SSC SSR Master Configuration Error State 1: General Configuration Error in the SSC SSR Master Mode 0: Configuration is ok.
Bit 6	SDI Configuration Error 1: General Configuration Error of the SDI Interface. 0: Configuration is ok.
Bit 7	Fieldbus Communication Configuration Error 1: General configuration error of the FBS interface 0: Configuration is ok.
Bit 8-13	Reserved
Bit 14	Extended Mapping Error 1: Configuration error in the mapping 0: Configuration is ok
Bit 15	Script Run Status 1: Script was loaded successfully and is running cyclically. 0: Script is stopped
Bit 16	Script Error State 1: An error has occurred during execution of the script 0: Script runs without errors

0x0004 Set device address for the SDI interface

In this memory register you have the option, to set a unique device address for communication via the SDI interface (Modbus)

A Modbus network (RS485) can consist of several modules. Therefore, the Modbus protocol provides the unique addressing via device addresses. If you want to access the IC-Modulwith a Modbus Master (e.g. PC with Modpoll), the Master must use the device address set in this register as the first byte in the send telegram.

The new settings are applied after a reset (Power Off/On or write Memory Register 0x0001 with value 0x0003).

Modbus Register	0x0004
Value Range	0x01-0xF7
Default Value	0x01
Number of available bytes	1
Permanently stored	Yes
Access	Read/Write

0x0005 Set bitrate for the SDI interface

In this memory register you have the option, to define with which bitrate the SDI interface should communicate.

Automatic bitrate detection means that the module tests the following bitrates until it has received a correct Modbus-RTU telegram:

• 2400 bit/s

• 4800 bit/s

• 9600 bit/s

• 19200 bit/s

• 38400 bit/s

• 57600 bit/s

• 115200 bit/s

NOTICE! During automatic bitrate detection the module does not send a reply to the master until the correct bitrate has been detected. This procedure can require up to 40 polls of the master.

TIP: Set a fixed bitrate if the automatic bitrate detection lasts too long for you.

The new settings are applied after a reset (Power Off/On or write Memory Register 0x0001 with value 0x0003).

Modbus Register	0x0005
Value Range	0x00-0x07
Default Value	0x00
Number of available bytes	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x00	Automatic bitrate detection
0x01	2400 bit/s
0x02	4800 bit/s
0x03	9600 bit/s
0x04	19200 bit/s
0x05	38400 bit/s
0x06	56700 bit/s
0x07	115200 bit/s

0x0006 Set parity bits for the SDI interface

In this memory register you have the option, to set the parity bit for the data transmission of the SDI interface.

The number of stop bits is adjusted automatically to the parity to ensure that a transmission always contains the same number of bits.

The new settings are applied after a reset (Power Off/On or write Memory Register 0x0001 with value 0x0003).

Modbus Register	0x0006
Value Range	0x00-0x02
Default Value	0x00 (Even Parity)
Number of available bytes	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x0000	Even Parity, 1 Stop-Bit
0x0001	Odd Parity, 1 Stop-Bit
0x0002	No Parity, (2 Stop-Bits)

0x0007 Current bitrate of the SDI interface

In this memory register you will find information about the currently used bitrate of the SDI interface.

Modbus Register	0x0007
Value Range	0x0000-0x0007
Initial value	-
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
0x0000	The bitrate is unknown or has not yet been determined by the automatic bitrate detection.
0x0001	2400 bit/s
0x0002	4800 bit/s
0x0003	9600 bit/s
0x0004	19200 bit/s
0x0005	38400 bit/s
0x0006	57600 bit/s
0x0007	115200 bit/s

0x0012 Set bitrate for the CDI interface

In this memory register you have the option, to set the Bitrate for the CDI

The new settings are applied after a reset (Power Off/On or write Memory Register 0x0001 with value 0x0003).

Modbus Register	0x0012
Value Range	0x01-0x07
Default Value	0x07 (115200 bit/s)
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x01	2400 bit/s
0x02	4800 bit/s
0x03	9600 bit/s
0x04	19200 bit/s
0x05	38400 bit/s
0x06	56700 bit/s
0x07	115200 bit/s

NOTICE

Automatic bitrate detection with the CDI is not possible.

➢ If the configuration that was entered is invalid, the corresponding registers use the following settings in order not to block the interface by incorrect data:

➔ 115200 bit/s, 1 stop bit, even parity

0x0013 Set transmission format for the CDI interface

In this memory register you have the option, to set the format of the data transmission for the CDI interface

Modbus Register	0x0013
Value Range	0x00-0x07
Default Value	1
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
Bit 0	Parity Enable (PEN) 1: Activate parity check 0: Do not activate parity check
Bit 1	Even or Odd (EOP) Only relevant if parity check is activated. 1: Odd Parity 0: Even Parity
Bit 2	Stop Bit (STB) 1: Use synchronisation with 2 stop bits 0: Use synchronisation with 1 stop bit.

Example: The value "0x05" ("00000_101b") means:

- Bit 0: (1) Activate parity check.
- Bit 1: (0) Set Even Parity.
- Bit 2: (1) Use synchronisation with 2 stop bits.

Bit order:

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
													SBT	EOP	PEN

0x0014 Current bitrate of the CDI

In this memory register you will find information about the currently used bitrate for the CDI interface.

The new settings are applied after a reset (Power Off/On or write Memory Register 0x0001 with value 0x0003).

Modbus Register	0x0014
Value Range	0x01-0x07
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
0x01	2400 bit/s
0x02	4800 bit/s
0x03	9600 bit/s
0x04	19200 bit/s
0x05	38400 bit/s
0x06	57600 bit/s
0x07	115200 bit/s

0x0015 Current data transmission format of the CDI interface

In this memory register you will find information about the current format of a data byte for the CDI .

Modbus Register	0x0015
Value Range	0x00-0x07
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
Bit 0	Parity Enable (PEN) 1: Activate parity control 0: Do not activate parity control
Bit 1	Even or Odd (EOP) Only relevant if parity control is activated. 1: Odd Parity 0: Even Parity
Bit 2	Stop Bit (STB) 1: Use synchronisation with 2 stop bits 0: Use synchronisation with 1 stop bit.

Example: The value "0x05" ("00000_101b") means:

- Bit 0: (1) Parity control activated.
- Bit 1: (0) Even Parity Control set.
- Bit 2: (1) Synchronisation with 2 stop bits used.

Bit order:

b15	**b14**	**b13**	**b12**	**b11**	**b10**	**b9**	**b8**	**b7**	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**
													SBT	EOP	PEN

0x0016 Configure SSC mode

In this memory register you have the option, to set the SSC mode of the module

You can operate the module in slave mode or master mode.

Modbus Register	0x0016
Value Range	0x0000-0x0003
Default Value	0x01
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x00	SSC SSR Master Mode deactivated
0x01	SSC SSR Master Mode (shift register, automatic detection)
0x02	SSC SSR Mode (slide register, manual configuration)
0x03	SSC SPI Slave Mode

0x0017 Current SSC Mode

In this memory register you will find information on the current SPI/ SSC mode of the module.

You can find further information on this topic in section"Synchronous serial interface [▸ 23]".

Modbus Register	0x0017
Value Range	0x0000-0x0003
Initial value	0x01
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
0x00	SSC SSR Master Mode deactivated
0x01	SSC SSR Master Mode (shift register, automatic detection)
0x02	SSC SSR Mode (shift register, manual configuration)
0x03	SSC SPI Slave Mode
0x04	SSC SSR Master Mode Error Status

0x0018 Configure SPI mode

In this memory register you have the option, Clock and data level for the SPI interface to set

This setting is only used in the SPI slave mode. In SSC Master Mode the SPI Controller always uses setting 4: "lagging edge, CLK high, MSB first" (see also Synchronous serial interface [▸ 23])

Modbus Register	0x0018
Value Range	0x0001-0x0004
Default Value	4
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x0001	Leading edge (CPHA=0, CLK low (CPOL=0), MSB first
0x0002	Leading edge (CPHA=0, CLK high (CPOL=1), MSB first
0x0003	Lagging edge (CPHA=1, CLK low (CPOL=0), MSB first
0x0004	Lagging edge (CPHA=1, CLK high (CPOL=1), MSB first

0x0019 Current configuration of the SPI controller

In this memory register you will find information about the current configuration of the clock and data level of the SPI controller.

Modbus Register	0x0019
Value Range	0x0000-0x0004
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0x0000	SSC/SPI deactivated
0x0001	Leading edge (CPHA=0, CLK low (CPOL=0), MSB first
0x0002	Leading edge (CPHA=0, CLK high (CPOL=1), MSB first
0x0003	Lagging edge (CPHA=1, CLK low (CPOL=0), MSB first
0x0004	Lagging edge (CPHA=1, CLK high (CPOL=1), MSB first

0x001a Set bitrate on the SSC interface

In this memory register you have the option, to set the bitrate on the SPI controller

Modbus Register	0x001a
Value Range	0x00-0x03
Default Value	0x01
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x00	Deactivated
0x01	~300kbit
0x02	~1200 Kbit/s
0x03	~4800 Kbit/s

0x001b Current bitrate on the SSC interface

In this memory register you will find information about the current bitrate of the SSC interface.

These values are only significant if you operate the module in slave mode. In slave mode the CANopen- determines the bitrate.

Modbus Register	0x001b
Value Range	0x00-0x03
Default Value	-
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0x00	Bitrate not set or invalid
0x01	~300kbit
0x02	~1200 kbit/s
0x03	~4800 kbit/s

0x001c Configure number of SSC Outputs

Prerequisite: You have activated the Master Mode

In this memory register you have the option, to set the number of output shift register modules for the cyclical data exchange. The size of each shift register module is 8 bits.

When you carry out a manual configuration here, you must make sure that the automatic register detection is not set, since these values are given priority. If the number of connected shift register modules does not match this register, the SSC interface switches to error state.

Modbus Register	0x001c
Value Range	0-32
Default Value	0
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x00	0 Shift registers
0x01	1 Shift registers
0x02 – 0x1F	…..
0x20	32 Shift registers

0x001d Current number of output shift register modules

Prerequisite: You have activated the Master Mode.

In this memory register you will find information about the current number of output shift register modules for the cyclical data exchange on the SSC interface.

Modbus Register	0x001d
Value Range	0-32
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0x00	0 Shift registers
0x01	1 Shift registers
0x02 – 0x0F	…
0x20	32 Shift registers

0x001e Configure number of input shift register modules

Prerequisite: You have activated the Master Mode

In this memory register you have the option, to set the number of input shift register modules for the cyclical data exchange. The size of each shift register module is 8 bits.

When you carry out a manual configuration here, you must make sure that the automatic register detection is not set, since these values are given priority. If the number of connected shift register modules does not match this register, the SSC interface switches to error state.

Modbus Register	0x001e
Value Range	0-32
Default Value	0
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0x00	0 Shift registers
0x01	1 Shift registers
0x02-0x1F	…
0x20	32 Shift registers

0x001f Current number of input shift register modules

Prerequisite: You have activated the Master Mode.

In this memory register you will find information about the current number of output shift register modules for the cyclical data exchange on the SSC interface.

Modbus Register	0x001f
Value Range	0-32
Default Value	-
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0x00	0 Shift registers
0x01	1 Shift registers
0x02-0x1F	…
0x20	32 Shift registers

0x0020 Module type

This register contains the unique identification number for the module type of the KUNBUS-IC modules. This module type provides information regarding which product type it is and which fieldbus the module is used for.

Modbus Register	0x0020
Value Range	0x00-0xff
Initial value	
Number of bytes available	1
Permanently stored	Yes
Access	Read Only
Meaning	
	IC-Modul for CANopen

0x0021 Default values in the data communication configure

In this memory register you have the option, to specify the behaviour of the memory register in case no data from outside is received anymore on the SSC Modbus RTU or fieldbus interface.

Modbus Register	0x0021
Value Range	0x00-0x3f
Default Value	0x00
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
Bit 1, Bit 0:	SS1 and SS0 (SSC interface) 00: Output data is set to 0 (default value) 01: Output data is set to 1 10: The data last written is retained
Bit 3, Bit 2:	FB1 and FB0 (fieldbus interface) 00: Output data is set to 0 (default value) 01: Output data is set to 1 10: The data last written is retained
Bit 5, Bit 4:	SS1 and SS0 (SDI interface) 00: Output data is set to 0 (default value) 01: Output data is set to 1 10: The data last written is retained

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										SD	SD	FB	FB	SS	SS

0x0022 Validity period of the process data on the SSC interface

In this memory register you have the option, to set the validity period of the process data on the SSC interface

The next production must take place within the specified period, otherwise the input data is marked as invalid. Output registers that are supplied with process data via the Data Broker from this input area then adjust themselves to the preselected safe values. You set these values in the Memory Register 0x0021.

You can find detailed information on this topic in section "Data Broker [▸ 13]".

Modbus Register	0x0022
Value Range	0-255
Default Value	0x00
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0	The data is valid indefinitely in acyclic operation.
1-255	Validity period in milliseconds (ms) The next production must follow within this time

0x0023 Validity period of the process data on the SDI interface

In this memory register you have the option, to define the validity period of the process data on the SDI interface.

The next production must take place within the specified period, otherwise the input data is marked as invalid. Output registers that are supplied with process data via the Data Broker from this input area then adjust themselves to the preselected safe values. You set these values in the Memory Register 0x0021.

You can find detailed information on this topic in section "Data Broker [▸ 13]".

Modbus Register	0x0023
Value Range	0-255
Default Value	0x00
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0	The data is valid indefinitely in acyclic operation.
1-255	Validity period in milliseconds (ms) The next production must follow within this time

0x0024 Validity period of the process data on the CANopen interface

In this memory register you have the option, to define the validity period of the process data on the CANopen interface.

The next production must take place within the specified period, otherwise the input data is marked as invalid. Output registers that are supplied with process data via the Data Broker from this input area then adjust themselves to the preselected safe values. You set these values in the Memory Register 0x0021.

You can find detailed information on this topic in section "Data Broker [▸ 13]".

Modbus Register	0x0024
Value Range	0-255
Default Value	0x00
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0	The data is valid indefinitely in acyclic operation.
1-255	Validity period in milliseconds (ms) The next production must follow within this time

0x0025 Configure shift chain

In this memory register you have the option, to use the first input and output register of the shift chain for connecting additional components (e.g. Status LEDs, Switches). If you do not want to use any further components, all registers can be used for the data transmission.

Modbus Register	0x0025
Value Range	0x00 – 0x07
Default Value	0x03
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
Bit 2, Bit 1	Input register of the shift chain 00: all registers are used for the data transmission. 01: Input register 0 is used for fieldbus switches. 10: Input register 0 and 1 are used for fieldbus switches.
Bit 0	Output register of the shift chain 0: all output registers are used for the data transmission. 1: Output register 0 is used for the status LEDs.

0x0026 Current assignment of the shift chain

In this memory register you will find information about using the first input and output register of the shift chain. You can detect whether additional components (e. g. status LEDs, switches) are used or whether all registers are used for the data transmission.

Modbus Register	0x0026
Value Range	0x00 – 0x07
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
Bit 0	Output register of the shift chain 0: all output registers are used for the data transmission. 1: Output register 0 is used for the status LEDs.
Bit 1, Bit 2	Input register of the shift chain 00: all registers are used for the data transmission. 01: Input register 0 is used for fieldbus switches. 10: Input register 0 and 1 are used for field-bus switches.

0x0032 Script Enable Register

In this register you have the option to activate or to deactivate the execution of a script.

You will find information about creating a script in the "KUNBUS-Scripter" documentation supplied.

Modbus Register	0x0032
Value Range	0x00 - 0xff
Default Value	0x00
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0	Deactivated
1	Activated

0x0033 Script Port Register

In this register you have the option to select the interface by which the script should communicate.

NOTICE

Please note that activation of a script influences the individual port.
E.g.: If the CDI port is used by the script, the CDI menu can no longer be used until you have deactivated the script.
If the CDI port is used by the script, you can only deactivate the script by writing the value 0 in the memory register 0x0032. You must restart the module to apply the deactivation.

Modbus Register	0x0033
Value Range	0x00-0x01
Default Value	0x01
Number of bytes available	1
Permanently stored	Yes
Access	Read/Write
Meaning	
0	CDI Interface
1	SDI Interface

0x0034 Script Status Register

In this register you will find information about the current status of the scripts.

Modbus Register	0x0034
Value Range	0x00-0xff
Default Value	0x00
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
0	The running script is in the initialisation phase
1	The script is running cyclically
2	The running script is waiting for data input or for a waiting period to elapse
3	The script was stopped or no script for running is loaded or running script is deactivated
4	Script cannot run due to a serious error

0x0035 Script Loop Register

In this register you have the option to monitor whether your script is running. During each run of the script, the value in the register is incremented.

Modbus Register	0x0035
Value Range	0x0000- 0xffff
Initial value	0
Number of bytes available	2
Permanently stored	No
Access	Read Only

0x0036 SSC Error Register

In this memory register you will find information about possible errors that have occurred when connecting the module to an external shift register chain.

Modbus Register	0x0036
Value Range	0x00-0x05
Default Value	-
Number of bytes available	1
Permanently stored	No
Access	Read Only
Meaning	
0	No Error
1	Centre tap is not receiving any data, line defective
2	Data In is not receiving any data
3	Number of input registers is not as expected
4	Number of output registers is not as expected
5	General error, e.g. electrical faults

Also see about this

Data Broker [13]

6.3 Register for the Mapping

In the following Memory registers you have the option to define the data mapping of the Data Broker for the input and output areas of the interfaces.

For each consumer (target register) there is a register area, in which all sources ("producers") are listed from which it obtains data. The data of the source areas is stored in the target area continuously in succession ("consumer") as well as the entries for this consumer. A maximum of 8 entries per consumer are possible. Each entry occupies 2 Memory Register. In the first Memory Register you can specify the base number of the Memory Register from which the data is copied. In the second Memory Register you determine the number of values you want to copy. Here, you can also exchange the High Byte and Low Byte (swap).

In the event of an invalid mapping, an error message flag is set in the status register 0x0002.

In section " Data Broker [▸ 13]" we will explain how a mapping works. You will also find an example of a mapping.

0x0e01-0x0e08 Output Data Mapping SSC

Modbus Register	0x0e01 – 0x0e08
Value Range	-
Default Value	0x00
Number of bytes available	16
Permanently stored	Yes
Access	Read/Write

0x0e21-0x0e28 Output Data Mapping SDI

Modbus Register	0x0e21 – 0x0e28
Value Range	-
Default Value	0x00
Number of bytes available	16
Permanently stored	Yes
Access	Read/Write

0x0e41-0x0e48 Output Data Mapping Fieldbus

Modbus Register	0x0e41 – 0x0e48
Value Range	-
Default Value	0x00
Number of bytes available	16
Permanently stored	Yes
Access	Read/Write

0x0f01-0xf40 Extended Output Mapping

In these memory registers have the option, to define a bit-accurate mapping. Individual bits can be mapped from any input data area to the output data area of your choice.

You can create a total of 16 different mappings.

Please note that a bit-accurate mapping requires very much run-time performance. Only use this function if you really need it.

To define a bit-accurate mapping, 4 registers must be defined in each case:

- In memory register 0xf01 enter the input memory register from which your data should originate.
- In memory register 0xf02 enter the output memory register in which you require the data.
- In memory register 0xf03 enter the source and target position of the bits that you want to map.
  - Define the source position via bit 0-3.
  - Define the target position via bit 4-7.
- In the memory register 0xf04 define the number of memory registers that you want to copy.
  - With bit 15 you can optionally swap the High Byte and Low Byte.

Modbus Register	0x0f01 – 0xf40
Value Range	-
Default Value	0x00
Number of bytes avail-able	128
Permanently stored	Yes
Access	Read/Write
Meaning	
Register 0, 4, 6, 8,…, 60	Source register number
Register 1, 5, 9,..., 61	Target register number
Register 2, 6, 10, ..., 62	Bit 0-3: Source bit position Bit 4-7: Target bit position
Register 3, 7, 11,..., 63	Bit 0-14: Number of registers to be copied. Bit 15: generated when setting a change from high and low byte (swap)

6.4 Memory of the Communication Channels

The following memory registers contain the input and output data of the communication channels. Cyclical process data is written there or read from there.

The Data Broker distributes this data cyclically according to the Mapping entries. At the same time, the input data of the producers is assigned to the output data of the consumers.

The data can be read from the input registers at any time via the Modbus protocol. It is only possible to write to input registers via the respective communication channels (only in the case of SDI is this the Modbus communication itself, of course). The initial value in the input registers is 0 until a register is written with process data.

Output registers that are not written by the Data Broker also contain the initial value 0 regardless of the setting for the drop-off value in the event of validity periods of the source data being exceeded. Output registers can solely be written by the Data Broker. Read access is not possible via Modbus, however.

0x1001 Input SSC

Modbus Register	0x1001-0x1080
Coil Address	0x0001 – 0x0800
Value Range	-
Initial value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read Only

0x1401 Input SDI

Modbus Register	0x1401-0x1480
Coil Address	0x2001 – 0x4000
Value Range	0x00-0xff
Initial value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read/Write

0x1801 Input Fieldbus

Modbus Register	0x1801-0x1880
Coil Address	0x4001 – 0x6001
Value Range	-
Initial value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read Only

0x2001 Output SSC

Modbus Register	0x2001-0x2080
Value Range	-
Coil Address	0x8001 – 0x8800
Initial value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read Only

0x2401 Output SDI

Modbus Register	0x2401-0x2480
Coil Address	0xa001 – 0xb000
Value Range	-
Default Value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read Only

0x2801 Output Fieldbus

Modbus Register	0x2801-0x2880
Coil Address	0xb001 – 0xe001
Value Range	-
Initial value	0x00
Number of bytes available	256
Permanently stored	No
Access	Read Only

6.5 Fieldbus specific Registers

0x4001 Fieldbus Status

In this memory register you will find information about the current communication status of the fieldbus interface.

Modbus Register	0x4001
Value Range	0x0000 - 0x0005
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0	Status is undefined
1	Bus in operation
2	Bus off error
3	Invalid node ID or invalid bitrate

0x4002 Module Status

In this memory register you will find information about the CANopen state of the module.

Modbus Register	0x4002
Value Range	0-4
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0	Initialisation
1	Error State
2	Fieldbus is stopped
3	Start-up phase
4	Operation

0x4003-0x4004 Manufacturer ID

In this memory register you have the option, to change the fieldbus specific manufacturer number for your application.

The Memory Register 0x4003 contains the High Word, Memory Register 0x4004 contains the Low Word of the manufacturer number.

Modbus Register	0x4003 - 0x4004
Value Range	-
Default Value	725 (0x000002d5)
Number of bytes available	4
Permanently stored	Yes
Access	Read/Write
Meaning	You can find the definition from the index 0x1018:01 in the CANopen object directory

0x4007-0x4008 Fieldbus Version

In this memory register you will find information about the fieldbus version.

The Memory Register 0x4007 contains the High Word, Memory Register 0x4008 contains the Low Word of the fieldbus version.

Modbus Register	0x4007 - 0x4008
Value Range	0x00000000 - 0xffffffff
Default Value	0
Number of bytes available	4
Permanently stored	Yes
Access	Read Only
Meaning	You can find the definition from the index 0x1018:03 in the CANopen object directory

0x4009 Firmware Version

In this memory register you will find information for the firmware version.

Modbus Register	0x4009
Value Range	0x0000 - 0xffff
Default Value	-
Number of bytes available	2
Permanently stored	Constant
Access	Read Only

0x400a-0x400b Serial Number

In these memory registers have the option, to change the serial number of the IC-Moduls.

In the delivered condition the KUNBUS serial number is stored.

The Memory Register 0x400a contains the High Word, register 0x400b contains the Low Word of the serial number.

Modbus Register	0x400a-0x400b
Value Range	0x00000000 - 0xffffffff
Default Value	KUNBUS Serial number
Number of bytes available	4
Permanently stored	Yes
Access	Read/Write

0x400c Current

In this memory register you will find information about the , with which the IC-Modul currently registers on the fieldbus.

You can set these addresses yourself or get them automatically via the LSS master.

NOTICE

If you have set the automatic assignment of the device address via the LSS master but no network is connected, this register displays the value "0". The "invalid setting" status appears in the CDI.

Modbus Register	0x400c
Value Range	0 - 127 (0x0000 - 0x007f)
Default Value	-
Number of bytes available	2
Permanently stored	No
Access	Read Only

0x400f Current Fieldbus Bitrate

In this memory register you will find information about the currently set fieldbusbitrate of the module.

Modbus Register	0x400f
Value Range	0x0001 - 0x0008, 0xffff
Default Value	-
Number of bytes available	2
Permanently stored	No
Access	Read Only
Meaning	
0x0000:	Bitrate not yet known
0x0001:	10 Kbit/s
0x0002:	20 Kbit/s
0x0003:	50 Kbit/s
0x0004:	125 Kbit/s
0x0005:	250 Kbit/s
0x0006:	500 Kbit/s
0x0007:	800 Kbit/s
0x0008:	1000 Kbit/s
0xffff:	Bus not connected

0x4010 Configured Fieldbus Bitrate

In this memory register you have the option, of storing a bitrate. Whether this bitrate is used depends on the selection you make in the Memory registers 0x4011-0x4012.

Modbus Register	0x4010
Value Range	0 - 8
Default Value	0
Number of bytes available	2
Permanently stored	Yes
Access	Read/Write
Meaning	
0	Automatic bitrate detection
1	10 kbit/s
2	20 kbit/s
3	50 kbit/s
4	125 kbit/s
5	250 kbit/s
6	500 kbit/s
7	800 kbit/s
8	1000 kbit/s

0x4011-0x4012 Fieldbus Configuration

In these registers you can define how the IC-Modul receives its device address.

Modbus Register	0x4011 - 0x4012
Value Range	
Default Value	0x00000009
Number of bytes available	4
Permanently stored	Yes
Access	Read/Write
Meaning	
Bit 1 , Bit 0	
	00: Memory register 0x400d and 0x4010 determine the node ID
	01: SSC determines the node ID (default, format acc. SC1- SC0)
	10: Node ID and bitrate are assigned via LSS from the fieldbus
	11: invalid
Bit 3, Bit 2	

	00	**Node ID and bitrate are set via coding switch** (requires 2 input shift registers) Setting of the node ID: (Bit 0-3 for setting the units digit, Bit 4-7 for setting the tens digit) One value range from 0-9 is allowed per bit. Values: 0: Node ID is assigned via LSS from the fieldbus 1-99: Node ID of the module Setting the bitrate: (bit 8-11) Value Range: 0: Automatic bitrate detection 1: 10 kBit/s 2: 20 kBit/s 3: 50 kBit/s 4: 125 kBit/s 5: 250 kBit/s 6: 500 kBit/s 7: 800 kBit/s 8: 1000 kBit/s 9-15: Invalid
	01	**Node ID and bitrate are set via the shift register** (requires 2 input shift registers) Setting of the node ID: (Bit 0-3 for setting the units digit, Bit 4-7 for setting the tens digit) One value range from 0-9 is allowed per bit. Other values are invalid. Values: 0: Node ID is assigned via LSS from the fieldbus 1-99: Node ID of the module The device address is set via the shift register. The automatic bitrate detection is accepted.

10	**Node Address and Bitrate are set via 8-Bit Binary Values** Settings for the bitrate: 00:Automatic bitrate detection 01: 250 KB 10: 500 KB 11: 1000 KB Settings for the node ID (bit 5-0) 000000: 1 000001: 2 ………… 111110: 61 111111: 62
11	**Invalid**

0x4014-0x4015 Product Code

In this memory register you will find information about the product number of the module. The product code of KUNBUS is stored in the delivered condition. You have the option to change the product code.

The Memory Register 0x4014 contains the High Word, Memory Register 0x4015 contains the Low Word of the product number.

Modbus Register	0x4014-0x4015
Value Range	0x00000000 - 0xffffffff
Default Value	-
Number of bytes available	24
Permanently stored	Yes
Access	Read/Write
Meaning	The default value is the KUNBUS product number of the IC-Moduls .

NOTICE

This value corresponds to the index 0x1018:02 in the CANopen object directory.

0x4016-0x4035 Product Name

In this memory register you have the option, to change the product name. The product name of the module assigned by KUNBUS is stored In the delivered condition.

You have up to 32 characters available in ASCII format for the name assignment.

NOTICE

If your product name is shorter than 32 characters, you have to fill up the remaining characters with the value "0".

Modbus Register	0x4016-0x4035
Value Range	32-Byte-String
Default Value	"KUNBUS IC CANopen "
Number of bytes available	32
Permanently stored	Yes
Access	Read/Write

0x4036 Size of Input Image

In this memory register you will find information about the number of bytes the Data Broker can receive via CANopen.

You define this value using the configuration parameters by the master.

Modbus Register	0x4036
Value Range	256
Default Value	256
Number of bytes available	2
Permanently stored	No
Access	Read Only

0x4037 Size of Output Image

In this memory register you will find information about the number of bytes the Data Broker can transmit via CANopen.

Modbus Register	0x4037
Value Range	256
Default Value	256
Number of bytes available	2
Permanently stored	No
Access	Read Only

0x4101 LSS-Status

In this memory register you will find information about the current des LSS.

Modbus Register		0x4101
Value Range		0x0000 - 0x0007
Default Value		0x0007
Number of bytes available		2
Permanently stored		No
Access		Read Only
Meaning		
	Bit 0	STP (Store Parameter) 0: current parameter stored permanently 1: current parameter stored temporarily
	Bit 1	LSN (LSS Node) 0: current device address was not con-figured by LSS master 1: current device address configured by LSS master
	Bit 2	LSB (LSS Bitrate) 0: current bitrate was not configured by LSS master 1: current bitrate configured by LSS master

Example: The value 0x0007 (0000_0111b) means:

Bit 0 = 1: current parameter stored temporarily

Bit 1 = 1: current device address configured by LSS

Bit 2 = 1: current bandwidth configured via LSS

7 Communication model

7.1 Object Directory

The standard objects are implemented according to specification DS302 (Version 4.02) of the CiA. You can find detailed information about the CANopen specification at www.can-cia.org.

You can configure some object values using the memory registers.

Index	Object Name	Sub Index	Description	Data Type	Access	Comment
1000h	Device Type	00h	Device Type	U32	RO	00000000: no profile
1001h	Error Register	00h	Error Register	U8	RO	
1003h	Predefined Error Field	00h	Number of errors occurred	U8	RW	
		01h-05h	Error memory	U32	RO	
1005h	COB-ID Sync	00h	SYNC Identifier	U32	RW	Default: 00000080
1006h	Communication Cycle Period	00h	Determines the sync interval in µs	U32	RW	
1008h	Manufacturer Device Name	00h	Device name	Visible string	RO	Modbus Register 4016 - 4025
100Ah	Manufacturer soft-ware version	00h	Software version	Visible string	RO	
100Ch	Guard time	00h	Duration (ms) between two guard telegrams	U16	RW	
100Dh	Life Time Factor	00h	Life Time Factor	U8	RW	
1010h	Store Parameters	00h	Max number of memory options	U8	RO	02h
		01h	Storage of all para-meters	U32	RW	Bitrate and node ID cannot be saved using this command.
		02h	Storage of the com-munication paramet-ers	U32	RW	Bitrate and node ID
1011h	Restore Parameters	00h	Max number of re-store options	U8	RO	02h
		01h	Restore all default parameters	U32	RW	
		02h	Only restore default communication parameters	U32	RW	
1014h	COB ID EMCY	00h	Emergency tele-gram identifier	U32	RO	
1015h	Inhibit Time EMCY	00h	Duration of emer-gency telegram	U32	RO	Default: 0000h
1016h	Consumer Heart-beat Time	00h	Number of entries	U8	RO	01h
		01h	Heartbeat cycle and node ID	U32	RW	The value of node ID and heart-beat time must be the multiple of 1 ms.
1017h	Producer Heartbeat Time	00h	Duration (ms) between two heart-beat telegrams	U16	RW	

1018h	Identity Object	00h	Number of entries	U8	RO	04h
		01h	Manufacturer ID	U32	RO	Memory register 4003 to 4004
		02h	Product Code	U32	RO	Memory register 4014 to 4015
		03h	Version number	U32	RO	Memory register 4007 to 4008
		04h	Serial number	U32	RO	Memory register 400a to 400b
1400h ... 1407h	Receive PDO Parameter	00h	Number of entries	U8	RO	02h
		01h	COB ID PDOx	U32	RW	
		02h	Transmission mode of the PDO	U8	RW	
1600h ... 1607h	Receive PDOx Mapping	00h	Number of entries	U8	RW	08h
		01h	Obj.-Mapping 1	U32	RW	
		02h	Obj.-Mapping 2	U32	RW	
		03h	Obj.-Mapping 3	U32	RW	
		04h	Obj.-Mapping 4	U32	RW	
		05h	Obj.-Mapping 5	U32	RW	
		06h	Obj.-Mapping 6	U32	RW	
		07h	Obj.-Mapping 7	U32	RW	
		08h	Obj.-Mapping 8	U32	RW	
1800h ... 1807h	Transmit PDO Parameter	00h	Number of entries	U8	RO	05h
		01h	COB ID PDOx	U32	RW	
		02h	Transmission mode of the PDO	U8	RW	
		03h	Repetition delay	U16	RW	
		05h	Event-Timer	U16	RW	
1a00h ... 1a07h	Transmit PDOx Mapping	00h	Number of entries	U8	RW	
		01h	Obj.-Mapping 1	U32	RW	
		02h	Obj.-Mapping 2	U32	RW	
		03h	Obj.-Mapping 3	U32	RW	
		04h	Obj.-Mapping 4	U32	RW	
		05h	Obj.-Mapping 5	U32	RW	
		06h	Obj.-Mapping 6	U32	RW	
		07h	Obj.-Mapping 7	U32	RW	
		08h	Obj.-Mapping 8	U32	RW	

You have the option to access the fieldbus data bytewise (8bit), wordwise (16bit) and double-wordwise (32bit).

Fieldbus Output Data (Direction Master)

Index	Object Name	Sub Index	Description	Data Type	Access	Comment
2000h	Output Buffer	00h	Number of entries	U8	RO	Access: Byte
		01h 02h ... 80h	Output Buffer Byte 0 Byte 1 ... Byte 127	U8	RO	Memory register 0x2801 to 0x2880
2001h	Output Buffer	00h	Number of entries	U8	RO	
		01h 02h ... 80h	Output Buffer Byte 128 Byte 129 ... Byte 255	U8	RO	
2010h	Output Buffer	00h	Number of entries	U8	RO	Access: Word
		01h 02h ... 40h	Output Buffer Word 0 Word 1 ... Word 63	U16	RO	Memory register 0x2801 to 0x2880
2011h	Output Buffer	00h	Number of entries	U8	RO	
		01h 02h ... 40h	Output Buffer Word 64 Word 65 ... Word 127	U16	RO	
2020h	Output Buffer	00h	Number of entries	U8	RO	Access: Doubleword
		01h 02h ... 20h	Output Buffer DWord 0 DWord 1 ... DWord 31	U32	RO	Memory register 0x2801 to 0x2880
2021h	Output Buffer	00h	Number of entries	U8	RO	
		01h 02 h ... 20h	Output Buffer DWord 32 DWord 33 ... DWord 63	U32	RO	

Fieldbus Input Data (from the Master)

Index	Object Name	Sub Index	Description	Data Type	Access	Comment
2100h	Input Buffer	00h	Number of entries	U8	RO	Access: Byte
		01h 02h ... 80	Input Buffer Byte 0 Byte 1 ... Byte 127	U8	RO	Memory register 0x1801 to 0x1880
2101h	Input Buffer	00	Number of entries	U8	RO	
		01h 02h ... 80h	Input Buffer Byte 128 Byte 129 ... Byte 255	U8	RO	
2110h	Input Buffer	00h	Number of entries	U8	RO	Access: Word
		01h 02h ... 40h	Input Buffer Word 0 Word 2 ... Word 63	U16	RO	Memory register 0x1801 to 0x1880
2111h	Input Buffer	00h	Number of entries	U8	RO	
		01h 02h ... 40h	Input Buffer Word 64 Word 65 ... Word 127	U16	RO	
2120h	Input Buffer	00h	Number of entries	U8	RO	
		01h 02h ... 20h	Input Buffer DWord 0 DWord 2 ... DWord 31	U32	RO	Access: Doubleword Memory register 0x1801 to 0x1880
2121h	Input Buffer	00h	Number of entries	U8	RO	
		01h 02h ... 20h	Input Buffer DWord 32 DWord 33 ... DWord 63	U32	RO	

8 CDI

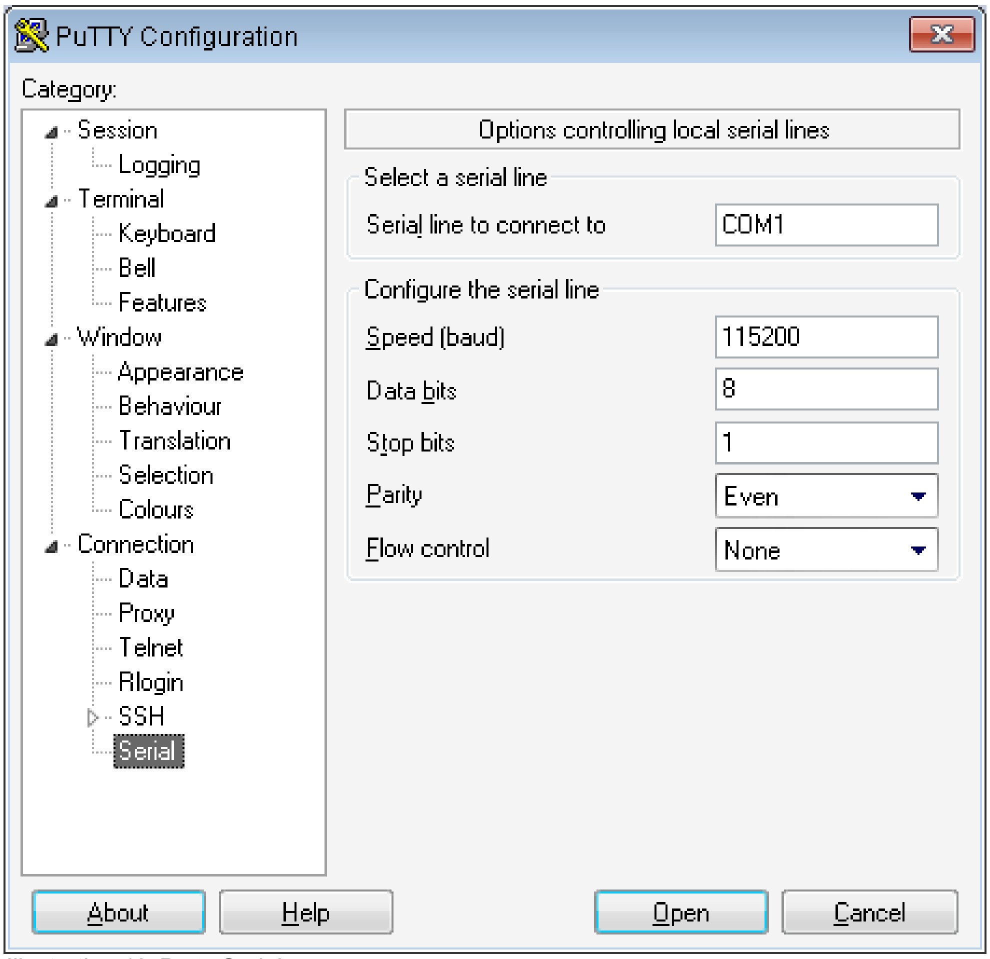
8.1 Setting up a Serial Connection

The CDI interface of the IC-Module is a UART interface (asynchronous serial interface) with 3.3V CMOS signal level.

If you connect a switch with level converter (e.g. "EXAR - SP3232EUEY") to the application connector via the RX and TX connections of this interface, an RS232 interface is available to you for connecting a terminal (you can find details from our sample circuit diagram under "RS232 Interface for CDI" in the Appendix).

You can connect the serial COMx interface of a PC to such a terminal interface (or a serial USB converter) and then access the CDI menus using a terminal emulation. We provide you with PuTTY as a terminal emulation on our support webpage.

How to start PuTTY:



Options controlling local serial lines

Select a serial line

Serial line to connect to: COM1

Configure the serial line

Speed (baud): 115200

Data bits: 8

Stop bits: 1

Parity: Even

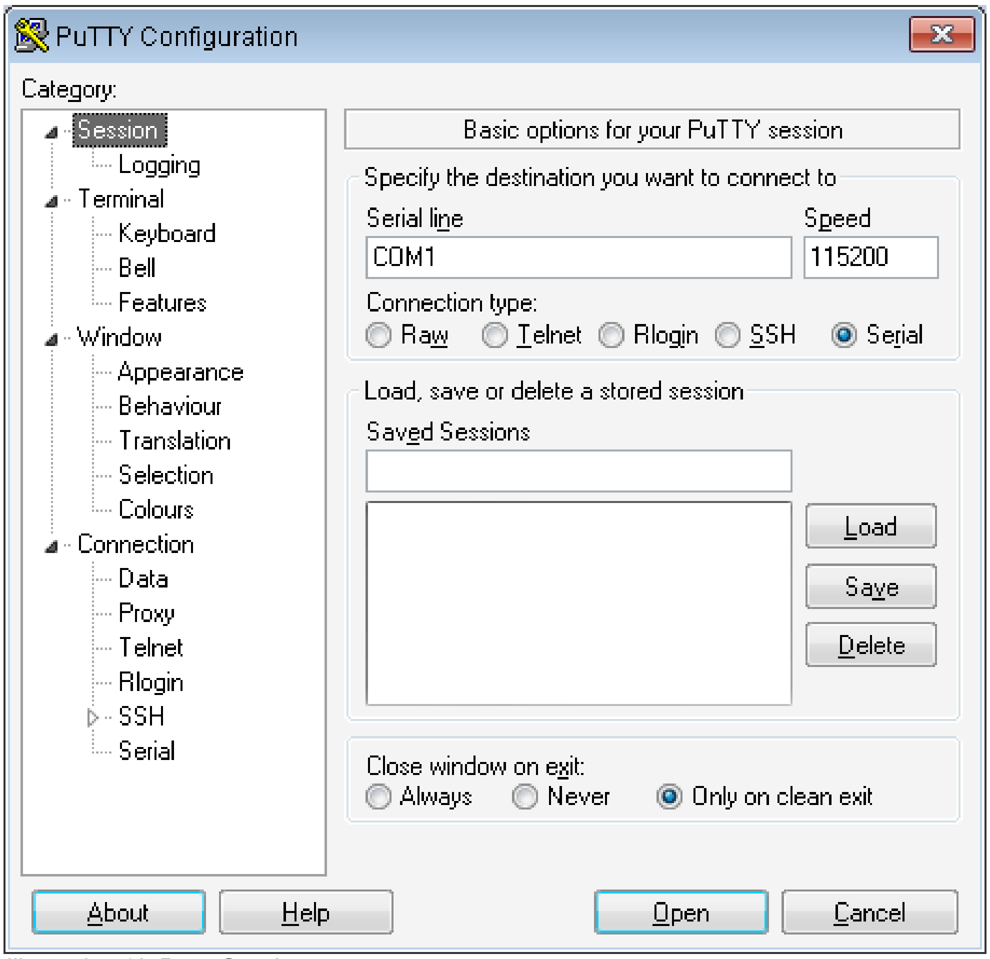
Flow control: None

Illustration 12: Putty Serial

- Change to the "Connection > Serial" view
- Select the serial interface that you want to access the CDI with (here: COM1)
- Configure the interface with the default settings of the CDI interface (for values see Fig above). Deactivate the dataflow control as well.
- Change to the "Session" view.
- Activate "Serial" as connection type. The "Serial line" and "Speed" fields are already preallocated with the settings you specified beforehand.
- Assign a name under "Saved Sessions" to save these settings.
- Click on "Save".
- Click on "Open".

⇨ The main menu will open in the terminal window.

NOTICE! If the terminal window remains black, click on the [ESC] key to start the data transmission via the CDI.


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM1

Speed: 115200

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

Load, save or delete a stored session

Saved Sessions:

Close window on exit: Always Never [✓ Only on clean exit]

Illustration 13: Putty Session

8.2 CDI Menus

Main Menu

The main menu is your access point for operating the module using the CDI. After a reset, the module transmits this main menu to the terminal.

```
--------------------------------------------------
KUNBUS-IC - Main Menu
--------------------------------------------------
1 - Module Information
2 - Interface Configuration
3 - Monitor Communication
4 - Module Status
--------------------------------------------------
>
```

Menu 1 – "Module Information"

1.Module Information

In this menu you will find general information about the module:

– Software revision
– Checksum of the firmware
– Serial number of the module
– Module type (unique throughout KUNBUS)

Menu 2 – "Interface Configuration"

2. Interface Configuration

In this menu you have the option to define the operational parameters for the different interfaces.

Here, you can set the mapping for the data broker.

Changes that you make in these menus are first activated after a restart.

The selection "`8 – Set Arbitrary Register`" allows you write access to all writeable Modbus registers of the module.

```
--------------------------------------------------
KUNBUS-IC – Interface Configuration
--------------------------------------------------
1  - SDI Communication
2  - CDI Communication
3  - SSC Communication
4  - SDI Output mapping
5  - SSC Output mapping
6  - Fieldbus Output mapping
7  - Fieldbus Specific
8  - Set Arbitrary Register
9  - Reset Module
10 - Extended Databroker
11 - Script Interpreter
12 - Reset to Factory Settings
--------------------------------------------------
```

2.1 SDI Communication

In this menu you will find information about the settings for the Modbus RTU communication. You can configure the values in the submenus.

```
--------------------------------------------------
KUNBUS-IC - SDI Communication
--------------------------------------------------
1 - Bitrate: Automatic Bitrate detection
2 - Parity: Even Parity, 1 Stopbit
--------------------------------------------------
>
```

2.1.1 SDI Communication Bitrate

In this menu you have the option, to select the appropriate Bitrate for your application.

The default value is "Automatic Bitrate detection".

```
--------------------------------------------------
KUNBUS-IC – SDI Communication: Set Bitrate
--------------------------------------------------
1 - Automatic Bitrate detection
2 - 2400   Bit/s
3 - 4800   Bit/s
4 - 9600   Bit/s
5 - 19200  Bit/s
6 - 38400  Bit/s
7 - 57600  Bit/s
8 - 115200 Bit/s
--------------------------------------------------
>
```

2.1.2 SDI Communication-Set Parity

In this menu you have the option to select the appropriate parity for your application from the displayed values.

The default value is "Even Parity (1 stop-bit)".

The number of stop-bits is based automatically on the parity setting. This ensures that a transmission always contains the same number of bits per byte.

```
--------------------------------------------------
KUNBUS-IC - SDI Communication: Set Parity
--------------------------------------------------
1 - Even Parity (1 Stopbit)
2 - Odd Parity (1 Stopbit)
3 - No Parity (2 Stopbits)
--------------------------------------------------
>
```

2.1.3 Set Modbus Node Address

In this menu you have the option, to enter the Modbus Node Address.

Permitted input values: 1-247

```
--------------------------------------------------------
KUNBUS-IC - SDI Communication: Set Modbus Node Address
--------------------------------------------------------
Enter a Modbus Node Address between 1 and 247:
--------------------------------------------------------
>
```

2.2 CDI Communication

In this menu you will find information about the currently set values of bitrate and parity. In the submenus you can configure the values.

```
--------------------------------------------------
KUNBUS-IC - CDI Communication
--------------------------------------------------
1 - Bitrate: 115200 Bit/s
2 - Parity: Even Parity, 1 Stopbit
--------------------------------------------------
>
```

2.2.1 CDI Communication - Set Bitrate

In this menu you have the option, to select the appropriate Bitrate for your application.

The default value is 115200 bit/s.

Automatic bitrate detection via the CDI is not possible.

```
--------------------------------------------------
KUNBUS-IC - CDI Communication: Set Bitrate
--------------------------------------------------
1 - 2400 Bit/s
2 - 4800 Bit/s
3 - 9600 Bit/s
4 - 19200 Bit/s
5 - 38400 Bit/s
6 - 57600 Bit/s
7 - 115200 Bit/s
--------------------------------------------------
>
```

2.2.2 CDI Communication - Set Parity

In this menu you have the option to select the appropriate parity for your application from the displayed values.

The default value is "Even Parity (1 stop-bit)".

The number of stop-bits is based automatically on the parity setting. This ensures that a transmission always contains the same number of bits per byte.

```
--------------------------------------------------
KUNBUS-IC - CDI Communication: Set Bitrate
--------------------------------------------------
1 - Even Parity, 1 Stopbit
2 - Even Parity, 2 Stopbit
3 - Odd Parity, 1 Stopbit
4 - Odd Parity, 2 Stopbit
5 - No Parity, 1 Stopbit
6 - No Parity, 2 Stopbit
--------------------------------------------------
>
```

SSC Mode

In this menu you have the option, to configure the shift register chain.

2.3 Select SSC SSR Master Mode

In this menu you have the following options:

- SSC SSR Master Mode, disabled: With this option, you switch off the SSC.
- SSC SSR Master Mode, auto detect shift registers: With this option, the number of input/output shift registers and the bitrate is determined automatically.
- SSC SSR Master Mode, configured shift registers: With this option, you can define the number of input/output shift registers and the bitrate yourself.
  - Enter [3] + [Enter] to select this mode as the configuration source.
  - With the option [9] you open the configuration menu.
- SSC SPI Slave Mode: With this selection you can make the settings for the SPI Slave Mode
  - Enter [4] + [Enter] to select this mode as the configuration source.
  - With the option [9] you open the configuration menu.

```
--------------------------------------------------
KUNBUS-IC- Select SSC SSR Master Mode
--------------------------------------------------
Mode: SSC SSR Master Mode, configured shift registers

1 - SSC SSR Master Mode, disabled
2 - SSC SSR Master Mode, auto detect shift registers
3 - SSC SSR Master Mode, configured shift registers
4 - SSC SPI Slave Mode

9 - Configure actual selection
--------------------------------------------------
>
```

2.3.2 SSC SSR Master Mode, Auto detect shift registers

In this menu, you can define how many shift registers are available for the integration of configuration switches and status LEDs.

```
-------------------------------------------------------------
KUNBUS-IC - SSC SSR Master Mode, auto detect shift registers
--------------------------------------------------
1 - No. of Fieldbus Switch Registers: 0
2 - No. of Fieldbus Status-LED Registers: 0
--------------------------------------------------
>
```

2.3.2.1 Number of Fieldbus Switch Registers

In this menu, you can define how many input shift registers are available for configuration switches.

Permissible input values: 0-2

```
------------------------------------------------------
KUNBUS-IC - SSC No. of Fieldbus Switch registers
------------------------------------------------------
>
```

2.3.2.2 Number of Status LED Registers

In this menu, you can define how many output shift registers are available for status LEDs.

Permissible input values: 0-1

```
------------------------------------------------------
KUNBUS-IC - SSC No. of Fieldbus Status-LED registers
------------------------------------------------------
>
```

2.3.3 SSC SSR Master Mode, configure shift registers

Prerequisite: In menu "2.3 - Select SSC SSR Master Mode" you have selected the configuration source "SSC SSR Master Mode, configure shift registers".

In this menu you have the following options:

– Number of input shift registers for configuration switches
– Number of output shift registers for status LEDs
– Number of input shift registers
– Number of output shift registers
– Bitrate

```
--------------------------------------------------
KUNBUS-IC- SSC SSR Master Mode, configure shift registers
--------------------------------------------------
3 - No of Fieldbus Switch Registers: 0
4 - No of Fieldbus Status-LED Registers: 0
3 - Number of overall Input Registers: 0
4 - Number of overall Output Registers: 0
5 - Configured Bitrate: 300 kBit/s
--------------------------------------------------
>
```

2.3.3.1 Number of Fieldbus Switch Registers

In this menu, you can define how many input shift registers are available for configuration switches.

Permissible input values: 0-2

```
------------------------------------------------------
KUNBUS-IC - SSC No. of Fieldbus Switch registers
------------------------------------------------------
>
```

2.3.3.2 Number of Fieldbus Status LED Registers

In this register, you can define how many output shift registers are available for status LEDs.

Permissible input values: 0-1

```
------------------------------------------------------
KUNBUS-IC - SSC No. of Fieldbus Status-LED registers
------------------------------------------------------
>
```

2.3.3.3 Number of overall Input Registers

In this menu you have the option to define the number of input shift registers. Valid input values are between 0 and 32.

```
--------------------------------------------------
KUNBUS-IC - SSC SSR Number of input shift registers
--------------------------------------------------
>
```

2.3.3.4 Number of overall Output Registers

In this menu you have the option to define the number of output shift registers. Valid input values are between 0 and 32.

```
--------------------------------------------------
KUNBUS-IC - SSC SSR Number of output shift registers
--------------------------------------------------
>
```

2.3.3.5 SSC SSR Bitrate

In this menu you have the option to set the bitrate. With option [1] "Auto detect Bitrate" the bitrate is determined automatically.

The values specified are to be regarded as reference values for "fast", "medium" and "slow". In SSC master mode these values are fallen short of by approx. 10%.

In slave mode the CANopen- determines the bitrate.

```
--------------------------------------------------
KUNBUS-IC - SSC SSR Bitrate
--------------------------------------------------
1 - Auto detect Bitrate
2 - 300 kBit/s
3 - 1200 kBit/s
4 - 4800 kBit/s
--------------------------------------------------
>
```

2.3.4 SSC SPI Slave Mode

In this menu you have the option to make settings for the SSC SPI Slave Mode.

In the first line you will see the current settings.

Using the option [1] "Configure settings" you can configure the settings.

```
--------------------------------------------------
KUNBUS-IC - SSC SPI Slave Settings
--------------------------------------------------
Settings: Polarity Normal, Falling Edge, CLK High, MSB first

1 - Configure settings
--------------------------------------------------
>
```

2.3.4.1 SSC SPI Slave Settings

In this menu you have the option to select the following combination of clock and data level.

You can find more information on this topic in section"Synchronous serial interface [▸ 23]".

```
--------------------------------------------------
KUNBUS-IC - SSC SPI Slave Settings
--------------------------------------------------
1 - Polarity Normal, Rising Edge, CLK Low, MSB first
2 - Polarity Normal, Rising Edge, CLK High, MSB first
3 - Polarity Normal, Falling Edge, CLK Low, MSB first
4 - Polarity Normal, Falling Edge, CLK High, MSB first
--------------------------------------------------
>
```

Note! Unfortunately, the representation in this CDI menu is not completely correct. In the memory register 0x0018 [▸ 55] you can see the correct representation.

2.4 SDI Output Mapping

In this menu you have the option to configure the Data-Broker-Mapping for the SDI.

Here you can specify up to four different register areas (start address and number of successive registers) as data sources.

Optionally, you can activate a timeout and specify default data that should be valid in the event of a timeout.

```
--------------------------------------------------
KUNBUS-IC - SDI Outputmapping
--------------------------------------------------
Src Register Number
1 - 1 (0x0001) | 0
2 - 1 (0x0001) | 0
3 - 1 (0x0001) | 0
4 - 1 (0x0001) | 0

5 - Default Data: all zero
6 - Valid Time: disabled
--------------------------------------------------
>
```

Configuring Output Mapping

```
--------------------------------------------------
KUNBUS-IC - Edit one map entry
--------------------------------------------------
Source Register:
```

✓ First determine the start address of the source register.
You can get an overview of the start addresses in the menuRegister for the Mapping [▷ 67].

- ◦ To do this, select a register from the input data area of the communication channels and then enter the number of registers to be used.
  - ⇨ After entering the number of registers, you will return automatically to the mapping overview.
- ◦ Restart the module.

⇨ Your mapping is now configured and will be applied in the operating mode.

```
--------------------------------------------------
KUNBUS-IC- Edit one map entry
--------------------------------------------------
Source Register: 0x1001
Number of Registers:
```

NOTICE! In the event of an invalid mapping (e.g. due to a register address outside the permitted input range or a register number that is too high) an error message appears.

Default data in Data Broker

In this menu you have the option, to define the default data for consumers used in the event of invalid production data.

If the data in the input registers should be invalid e.g. because no data has arrived anymore from outside, the Data Broker uses the following values:

– All bits are 1
– All bits are 0
– All bits remain unchanged

```
--------------------------------------------------
KUNBUS-IC - Default data in data broker
--------------------------------------------------
1 - all data bit are zero
2 - all data bit are one
3 - old production values are used
--------------------------------------------------
>
```

Production valid time

In this menu you have the option,to determine the validity period of the register contents to be read out.

The value range of the validity periods can be defined between 0 and 255 ms.

If no new data from the producer should arrive in the input registers within the predefined period, the Data Broker uses the data set among the default values.

```
--------------------------------------------------
KUNBUS-IC – Production valid time (0 = disabled)
--------------------------------------------------
>
```

2.5 SSC Output Mapping

In this menu you have the option to configure the Data-Broker-Mapping for the SDI.

Here you can specify up to four different register areas (start address and number of successive registers) as data sources.

Optionally, you can activate a timeout and specify default data that should be valid in the event of a timeout.

```
--------------------------------------------------
KUNBUS-IC - SSC Outputmapping
--------------------------------------------------
Src Register Number
1 - 1 (0x0001) | 0
2 - 1 (0x0001) | 0
3 - 1 (0x0001) | 0
4 - 1 (0x0001) | 0

5 - Default Data: all zero
6 - Valid Time: disabled
--------------------------------------------------
>
```

Configuring Output Mapping

```
--------------------------------------------------
KUNBUS-IC - Edit one map entry
--------------------------------------------------
Source Register:
```

✓ First determine the start address of the source register.
  You can get an overview of the start addresses in the menuRegister for the Mapping [▸ 67].

◦ To do this, select a register from the input data area of the communication channels and then enter the number of registers to be used.

  ⇨ After entering the number of registers, you will return automatically to the mapping overview.

◦ Restart the module.

⇨ Your mapping is now configured and will be applied in the operating mode.

```
--------------------------------------------------
KUNBUS-IC- Edit one map entry
--------------------------------------------------
Source Register: 0x1001
Number of Registers:
```

NOTICE! In the event of an invalid mapping (e.g. due to a register address outside the permitted input range or a register number that is too high) an error message appears.

Default data in Data Broker

In this menu you have the option, to define the default data for consumers used in the event of invalid production data.

If the data in the input registers should be invalid e.g. because no data has arrived anymore from outside, the Data Broker uses the following values:

– All bits are 1
– All bits are 0
– All bits remain unchanged

```
--------------------------------------------------
KUNBUS-IC - Default data in data broker
--------------------------------------------------
1 - all data bit are zero
2 - all data bit are one
3 - old production values are used
--------------------------------------------------
>
```

Production valid time

In this menu you have the option,to determine the validity period of the register contents to be read out.

The value range of the validity periods can be defined between 0 and 255 ms.

If no new data from the producer should arrive in the input registers within the predefined period, the Data Broker uses the data set among the default values.

```
--------------------------------------------------
KUNBUS-IC - Production valid time (0 = disabled)
--------------------------------------------------
>
```

2.6 Fieldbus Output Mapping

In this menu you have the option, to configure the Data-Broker-Mapping for the fieldbus interface.

Here you can specify up to four different register areas (start address and number of successive registers) as data sources.

Optionally, you can activate a timeout and specify default data that should be valid in the event of a timeout.

```
KUNBUS-IC - Fieldbus Outputmapping
--------------------------------------------------
Src Register Number
1 - 1 (0x0001) | 0
2 - 1 (0x0001) | 0
3 - 1 (0x0001) | 0
4 - 1 (0x0001) | 0

5 - Default Data: all zero
6 - Valid Time: disabled
--------------------------------------------------
>
```

Configuring Output Mapping

```
--------------------------------------------------
KUNBUS-IC - Edit one map entry
--------------------------------------------------
Source Register:
```

✓ First determine the start address of the source register.
You can get an overview of the start addresses in the menuRegister for the Mapping [▸ 67].

- ◦ To do this, select a register from the input data area of the communication channels and then enter the number of registers to be used.
  - ⇨ After entering the number of registers, you will return automatically to the mapping overview.
- ◦ Restart the module.

⇨ Your mapping is now configured and will be applied in the operating mode.

```
--------------------------------------------------
KUNBUS-IC- Edit one map entry
--------------------------------------------------
Source Register: 0x1001
Number of Registers:
```

NOTICE! In the event of an invalid mapping (e.g. due to a register address outside the permitted input range or a register number that is too high) an error message appears.

Default data in Data Broker

In this menu you have the option, to define the default data for consumers used in the event of invalid production data.

If the data in the input registers should be invalid e.g. because no data has arrived anymore from outside, the Data Broker uses the following values:

– All bits are 1

– All bits are 0

– All bits remain unchanged

```
--------------------------------------------------
KUNBUS-IC - Default data in data broker
--------------------------------------------------
1 - all data bit are zero
2 - all data bit are one
3 - old production values are used
--------------------------------------------------
>
```

Production valid time

In this menu you have the option,to determine the validity period of the register contents to be read out.

The value range of the validity periods can be defined between 0 and 255 ms.

If no new data from the producer should arrive in the input registers within the predefined period, the Data Broker uses the data set among the default values.

```
--------------------------------------------------
KUNBUS-IC – Production valid time (0 = disabled)
--------------------------------------------------
>
```

2.7 CANopen settings

In the following menu items, you have the option to make field-specific settings.

Using the selection [1]-[3] + [Return] select the required configuration source.

Using the selection [9] + [Return] you can make adjustments to the configuration source.

```
--------------------------------------------------
KUNBUS-IC - CANopen Settings
--------------------------------------------------
Actual Config Source: Shift register

1 - Modbus Register
2 - Switch attached to shift register
3 - LSS Setting from CANopen master

9 - Configure Actual Selection
--------------------------------------------------
>
```

2.7.1 Modbus Register

In this menu, you can assign node ID and bitrate via the storage register.

```
--------------------------------------------------
KUNBUS-IC - CANopen Mbus Register setting
--------------------------------------------------
Node ID (0x400d): 1
Bitrate (0x4010): automatic detection

1 - Change Node ID
2 - Change Bitrate
--------------------------------------------------
>
```

2.7.1.1 Set Node-ID

In this menu, you can enter the required node ID directly and save it permanently.

Possible input values: 1-127

```
--------------------------------------------------
KUNBUS-IC - CANopen Mbus Register. Set Node ID
--------------------------------------------------
>
```

2.7.1.2 Change Bitrate

Here, you can set the bitrate.

```
--------------------------------------------------
KUNBUS-IC - CANopen Mbus Register. Set Bitrate
--------------------------------------------------
1 - Automatic bitrate detection
2 - 10 kbit
3 - 20 kbit
4 - 50 kbit
5 - 125 kbit
6 - 250 kbit
7 - 500 kbit
8 - 800 kbit
9 - 1 Mbit
--------------------------------------------------
>
```

2.7.2 Switch attached to shift Register

If you want to assign node ID and bitrate via the SSC, enter [2] + [Return] in the "CANopen settings" menu. The SSC is now selected as the current configuration source. By entering [9] + [Return] the displayed dialog window will open.

Here, you can now select the required settings.

```
--------------------------------------------------
KUNBUS-IC - CANopen Switch Settings
--------------------------------------------------
Actual: Node-ID 6 Bit, Bitrate 2 Bit

1 - Node-ID 2x BCD, Bitrate automatic detection
2 - Node-ID 2x BCD, Bitrate 1x BCD
3 - Node-ID 6 Bit, Bitrate 2 Bit
--------------------------------------------------
>
```

2.7.3 LSS Settings from CANopen master

With this selection the address is obtained automatically via the CANopen-Master.

2.8 Set Arbitrary Register

In this menu you have the option,to make adjustments to the writeable registers that do not have their own menu item.

First enter the register number:

```
--------------------------------------------------
KUNBUS-IC - Set Arbitray Register
--------------------------------------------------
Register Number:
```

If the register number does not exist or is write-protected, you will receive a corresponding error message.

```
--------------------------------------------------
KUNBUS-IC - Set Arbitray Register
--------------------------------------------------
Register Number: 0x5001 <<Register doesn‘t exist or no write access
.Register Number:
```

If the register number is valid, the dialog first shows you the set value in decimal, hexadecimal and binary notation.

```
--------------------------------------------------
KUNBUS-IC - Set Arbitray Register
--------------------------------------------------
Register Number: 0x0013
Current Value : 1 0x0001 0000_0000_0000_0001b
Enter New Value:
```

Now specify a new value for the register and press [Return] to confirm your entry.

```
--------------------------------------------------
KUNBUS-IC - Set Arbitray Register
--------------------------------------------------
Register Number: 0x0013
Current Value : 1 0x0001 0000_0000_0000_0001b
Enter New Value: 2
New Value Set : 2 0x0002 0000_0000_0000_0010b

Register Number:
```

The configuration interface detects whether it is a decimal, hexadecimal or binary value based on the notation and displays the saved value once again for confirmation.

⚠ CAUTION

Please note that some settings could lead to functional incapacity of the CDI interface.

You can only remedy such an inaccurate setting with a functional SDI interface.

- ➢ When correcting the error, all settings made previously are reset.
- ➔ Write the value 0x0002 in the register 0x0001
- ⇨ You have set the module to the original settings. The CDI interface is now available again.

2.9 Reset Module

With this option you can restart the module. The module hereby interrupts the fieldbus communication temporarily.

A restart is necessary for the module to apply changes of the parameters.

2.10 Extended (Bit)Mapping

In this menu you will find information about the current settings of the extended Data Broker. You can configure the individual values in the corresponding submenus.

The Extended Mapping is executed after the general Mapping. You can combine both mapping functions by copying the larger areas with the standard mapping and change individual bits with the extended mapping afterwards.

You can define a total of 16 mappings. A mapping can be up to 1024 bits long.

You must store the following information for each mapping:

– Address of the source register
– Bit position within the source register
– Address of the target register
– Bit position within the target register
– Number of bits to be copied

Info!:

– Bitwise copying requires performance.
The module works cyclically. During each cycle, the different interfaces are operated in series and the data from the Data Broker is distributed between the input and output data areas. Since all interfaces process the data traffic independently of each other (asynchronous) and store the data produced or used in a buffer, they are independent of the module's cycle. In the case of very fast interfaces and a long cycle time, it is possible, however, that the Data Broker does not distribute all incoming data completely if several data packets arrive at the interface within a cycle. Conversely, it may happen in the case of slow interfaces that the 2nd cycle will already proceed and the output values of an interface will be changed by the Data Broker before the values from the first cycle have been transmitted via the interface.
For this reason, the cycle times of the module can be relevant. These

are normally less than 1 ms. Individual cycles, however, can also last more than 5 ms. If you use the Extended Mapping very extensively, the cycle times can increase considerably.

  – Therefore, only use it if necessary.
  – Only use it for small areas.

– Bear in mind that the mappings are processed sequentially. If target areas overlap, this can cause problems.

```
--------------------------------------------------
KUNBUS-IC - Extended (Bit)Mapping
--------------------------------------------------
Source | Bitp | Dest. | Bitp | Length

1  - 0x0001 | 0 | 0x0001 | 0 | 0
2  - 0x0001 | 0 | 0x0001 | 0 | 0
3  - 0x0001 | 0 | 0x0001 | 0 | 0
4  - 0x0001 | 0 | 0x0001 | 0 | 0
5  - 0x0001 | 0 | 0x0001 | 0 | 0
6  - 0x0001 | 0 | 0x0001 | 0 | 0
7  - 0x0001 | 0 | 0x0001 | 0 | 0
8  - 0x0001 | 0 | 0x0001 | 0 | 0
9  - 0x0001 | 0 | 0x0001 | 0 | 0
10 - 0x0001 | 0 | 0x0001 | 0 | 0
11 - 0x0001 | 0 | 0x0001 | 0 | 0
12 - 0x0001 | 0 | 0x0001 | 0 | 0
13 - 0x0001 | 0 | 0x0001 | 0 | 0
14 - 0x0001 | 0 | 0x0001 | 0 | 0
15 - 0x0001 | 0 | 0x0001 | 0 | 0
16 - 0x0001 | 0 | 0x0001 | 0 | 0

--------------------------------------------------
>
```

The submenu for entering the mapping is displayed by entering a number from 1 to 16:

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1001, Bitpos: 2
Destination: 0x2001, Bitpos: 0
(Bit)Length: 8

1 - change mapping
2 - clear mapping
--------------------------------------------------
>
```

Any mapping that might exist is deleted by entering [2]+[Enter].

After entering [1]+[Enter], the Source Address, Bitpos, Destination Address, Bitpos and Bit length are polled sequentially:

Source

– Specify here which register the data should originate from.

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1401, Bitpos: 14
Destination: 0x2401, Bitpos: 2
(Bit)Length: 0x2400, Bitpos: 2

1 - change mapping
--------------------------------------------------
>1
Source Register >
```

Source Bit Position

– Specify here the exact position of the bit that you want to map.

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1401, Bitpos: 14
Destination: 0x2401, Bitpos: 2
(Bit)Length: 0x2400, Bitpos: 2

1 - change mapping
--------------------------------------------------
>1
Source Register >0x1810
Source Bit Position >
```

Destination Register

– Specify here the register where you want to output the data.

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1401, Bitpos: 14
Destination: 0x2401, Bitpos: 2
(Bit)Length: 0x2400, Bitpos: 2

1 - change mapping
--------------------------------------------------
>1
Source Register >0x1810
Source bit position >5
Destination Register >
```

Destination Bit Position

– Specify the exact position of the bit that you want to map the data on.

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1401, Bitpos: 14
Destination: 0x2401, Bitpos: 2
(Bit)Length: 0x2400, Bitpos: 2

1 - change mapping
--------------------------------------------------
>1
Source Register >0x1810
Source bit position >5
Destination Register >0x2405
Destination Bit Position >
```

Bit Length

– Specify here the number of bits you want to map.

```
--------------------------------------------------
KUNBUS-IC - Extended Mapping, 1 Entry
--------------------------------------------------
Entry : 1
Source : 0x1401, Bitpos: 14
Destination: 0x2401, Bitpos: 2
(Bit)Length: 0x2400, Bitpos: 2

1 - change mapping
--------------------------------------------------
>1
Source Register >0x1810
Source bit position >5
Destination Register >0x2405
Destination bit position >3
Number of bits to map >
```

Main menu with mapping entry

After confirming the number of bits with the enter key, you will return automatically to the main menu.

Here, you will now see the mapping you created in the previous steps:

```
--------------------------------------------------
KUNBUS-IC - Extended (Bit)Mapping
--------------------------------------------------
Source | Bitp | Dest. | Bitp | Length

1  - 0x1810 | 5 | 0x2405 | 3 | 27
2  - 0x0001 | 0 | 0x0001 | 0 | 0
3  - 0x0001 | 0 | 0x0001 | 0 | 0
4  - 0x0001 | 0 | 0x0001 | 0 | 0
5  - 0x0001 | 0 | 0x0001 | 0 | 0
6  - 0x0001 | 0 | 0x0001 | 0 | 0
7  - 0x0001 | 0 | 0x0001 | 0 | 0
8  - 0x0001 | 0 | 0x0001 | 0 | 0
9  - 0x0001 | 0 | 0x0001 | 0 | 0
10 - 0x0001 | 0 | 0x0001 | 0 | 0
11 - 0x0001 | 0 | 0x0001 | 0 | 0
12 - 0x0001 | 0 | 0x0001 | 0 | 0
13 - 0x0001 | 0 | 0x0001 | 0 | 0
14 - 0x0001 | 0 | 0x0001 | 0 | 0
15 - 0x0001 | 0 | 0x0001 | 0 | 0
16 - 0x0001 | 0 | 0x0001 | 0 | 0

--------------------------------------------------
>
```

The extended mapping is stored permanently in the module, but is first executed after a restart.

2.11 Script Interpreter

In this menu you have the option, to activate or deactivate a script and to select the desired port.

NOTICE

Please note that activation of a script influences the individual port.

Never activate a script with the setting "Port used by script: CDI" if you do not have a fully functional SDI interface for accessing the parameter memory registers! In such a case, your module is no longer configurable!
You can switch the status between disabled and enabled by entering [1]+[Return].

– You can define the SDI or CDI as port. You switch between both options by entering [2]+[Return].

```
-------------------------------------------------
KUNBUS-IC - Script Interpreter
-------------------------------------------------
1 - State: disabled
2 - Port used by script: SDI
-------------------------------------------------
>
```

2.12 Reset to Factory Settings

With this option, you reset all parameters of the IC module to the default values.

You have to confirm the selection by entering the number "36" for reasons of security. Afterwards, the module restarts.

NOTICE

Loss of all settings

Bear in mind that with this function you will delete all settings that you have previously made.

Menu 3 – "Monitor Communication"

3 Monitor Communication

In this menu you have the option to view the current values of the Memory Register.

```
--------------------------------------------------
KUNBUS-IC - Modbus Register Monitor
--------------------------------------------------
1 - SSC In
2 - SDI In
3 - Fieldbus In (from Master)
4 - SSC Out
5 - SDI Out
6 - Fieldbus Out (to Master)
7 - Arbitrary Register
--------------------------------------------------
>
```

3.1 Monitor SSC Input Registers

In this menu you will get an overview of the current values of the input areas of the shift register chain.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC - Monitor SSC Input Registers
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x1001: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1009: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1011: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1019: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1021: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1029: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1031: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1039: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
>
```

3.2 Monitor SDI Input Registers

In this menu you will get an overview of the input area's current values of the SDI interface.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC - Monitor SDI Input Registers
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x1401: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1409: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1411: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1419: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1421: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1429: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1421: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1439: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
```

3.3 Monitor Fieldbus Input (from Master)

In this menu you will get an overview of the input area's current values of the CANopen interface.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC – Monitor Fieldbus Input (from Master)
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x1801: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1809: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1811: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1819: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1821: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1829: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1831: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x1839: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
>
```

3.4 Monitor SSC Output Registers

In this menu you will get an overview of the output area's current values of the shift register chain.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC – Monitor SSC Output Registers
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x2001: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2009: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2011: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2019: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2021: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2029: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2031: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2039: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
>
```

3.5 Monitor SDI Output Registers

In this menu you will get an overview of the output area's current values of the SDI interface.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC - Monitor SDI Output Registers
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x2401: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2409: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2411: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2419: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2421: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2429: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2431: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2439: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
>
```

3.6 Monitor Fieldbus Output (to Master)

In this menu you will get an overview of the output area's current values of the CANopen interface.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC - Monitor Fieldbus Output (to Master)
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x2801: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2809: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2811: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2819: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2821: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2829: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2831: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
0x2839: 0x0000 0x0000 0x0000 0x0000 0x0000 0x0000
>
```

3.7 Arbitrary Register

In this menu you will get an overview of all registers. Use this function to view registers that do not have their own menu item.

First enter the register number (e.g. “0x4001”):

```
--------------------------------------------------
KUNBUS-IC - Monitor Arbitray Register
--------------------------------------------------

Register Number: 0x4001
```

The CDI then displays 64 registers from the specified register value.

In the first row you see the setting values for the display. Here, you can select a binary [b], hexadecimal [h] or decimal [d] format.

```
--------------------------------------------------
KUNBUS-IC - Monitor Arbitray Register
--------------------------------------------------

b=binary, h=hex, d=decimal, n=next, p=previous <cr>=refresh

0x4001: ...... ...... 0x0490 ...... 0x002b ...... 0x0102
0x4009: 0x0ef5 0x0001 0xe240 ...... ...... 0x0000 ......
0x4011: ...... 0x0001 ...... ...... 0x0004 0x0055 0x004e
0x4019: 0x0042 0x0055 0x0053 0x002d 0x0043 0x004f 0x004d
0x4021: 0x0020 0x0045 0x0074 0x0068 0x0065 0x0072 0x004e
0x4029: 0x0074 0x002f 0x0049 0x0050 0x0000 0x0017 0x0000
0x4031: 0x0000 0x0008 0x00e1 0x0000 0x0020 ...... ......
0x4039: ...... ...... ...... ...... ...... ...... ......
>
```

The CDI represents registers that do not exist by dots.

Menu 4 – "Module Status"

4 Module Status

In the following submenu you will get an overview of the current operating state of the module and of all interfaces.

```
--------------------------------------------------
KUNBUS-COM – Module Status
--------------------------------------------------
1 – Common Status
2 – SDI Status
3 – SSC Status
4 – CDI Status
5 – Fieldbus Status
7 – Error Stack
8 – Script Status
--------------------------------------------------
>
```

4.1 Common Status

In this menu you will find information about the current operating state of all communication channels.

The CDI displays the states of the interfaces, connection mode for the main board, configuration errors as well as errors in the Data Broker Mapping.

```
-------------------------------------------------
KUNBUS-IC - Common Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

Fieldbus State: is stopped
Fieldbus Configuration: is ok
Fieldbus Mapping: is ok

Synchron serial communication: is stopped
Sync. serial comm. Configuration: is ok
Sync. serial comm. Mapping: is ok

Modbus RTU (SDI) Mapping: is ok
Modbus RTU (SDI) Configuration: is ok
--------------------------------------------------
>
```

4.2 SDI Status

In this menu you will find information about the current interface parameters and node ID.

If the interface is running in "Automatic Bitrate detection" mode and no bitrate has been detected yet, the message "Actual Bitrate: undefined" will be displayed instead of the interface parameters.

```
--------------------------------------------------
KUNBUS IC - SDI Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

Modbus Node Address: 1
Actual Bitrate: 38400 Bit/s
Actual Parity: even
Actual Stopbits: 1 Stopbit
--------------------------------------------------
>
```

4.3 SSC Status

In this menu you will find information about the status of the SSC interface. The representation is dependent on the selected mode and configurations:

SSC SSR Master Mode, auto detect shift registers

- You have selected "auto detect shift registers" as the configuration source for the SSC.

In this menu you will find the following information:

- Number of shift registers used fieldbus specifically
- Number of status LEDs used
- Number of shift registers
- Bitrate

If the shift register chain is not working, the text "Shift Register chain not working" will be displayed instead of the interface parameters.

```
--------------------------------------------------
KUNBUS-IC - SSC Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

SSC SSR Master Mode, auto detect shift registers

Act. No. of Fieldbus Switch regs: 0
Act. No. of Fbus Status-LED regs: 0
Actual Number of Input registers: 1/1
Actual Number of Output registers: 1/1
Actual Bitrate: 4800 kBit/s
--------------------------------------------------
>
```

SSC SSR Master Mode, configured registers

– You have selected "configured shift registers" as the configuration source for the SSC.

In this menu you will find the following information:

– Number of shift registers used fieldbus specifically

– Number of status LEDs used

– Number of shift registers

– Bitrate

If the number of configured registers does not match the number of connected registers or the shift register chain is not working, you will see the message "Shift Register Chain not working“.

```
--------------------------------------------------
KUNBUS-IC- SSC Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

SSC SSR Master Mode, configured shift registers

Act. No. of Fieldbus Switch regs: 0
Act. No. of Fbus Status-LED regs: 0
Actual Number of Input registers: 1/1
Actual Number of Output registers: 1/1
Actual Bitrate: 4800 kBit/s
--------------------------------------------------
>
```

SSC SSR Master Mode, disabled

You will find this representation in one of the following cases:

– SSC SSR Master is switched off via the menu

– Invalid configuration of the number of input or output registers

– Transmission error due to unsuitable bitrate

– Other transmission errors (e.g. line interruption)

```
--------------------------------------------------
KUNBUS-IC - SSC Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

SSC SSR Master Mode, disabled
--------------------------------------------------
>
```

SSC SPI Slave Mode

– You have selected "SSC SPI Slave Mode" as the configuration source.

In this menu you will find information about clock and data level.

```
--------------------------------------------------
KUNBUS-IC - SSC Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

SSC SPI Slave Mode

Actual SPI Settings: Polarity Normal, Falling Edge, CLK High, MSB
first
--------------------------------------------------
>
```

4.4 CDI Status

In this menu you will find information about the current parameters of the CDI.

```
--------------------------------------------------
KUNBUS- IC-Modul - CDI Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

Actual Bitrate: 115200 Bit/s
Actual Parity: even
Actual Stopbits: 1 Stopbit
--------------------------------------------------
>
```

4.5 CANopen Status

In this menu you will find information about the current status of the fieldbus interface.

```
-------------------------------------------------
KUNBUS-IC - CANopen Status
--------------------------------------------------
<cr> = refresh; <Esc> = return

Vendor Id: 0x000002d5
Device Type: 0x00000002
Revision: 0x00000000
Serial number: 4294967295
Node ID: 17
Bitrate: 250 kBit/s
Busstate: Bus is running
CANopen: Operational State
--------------------------------------------------
>
```

4.6 Error Stack

In this menu you will find information about the errors that last occurred. Please have these entries ready at hand when you contact our support. We can help you faster by evaluating the entries.

If the value is 0x00000000, no error has occurred.

The Error Stack occupies the register 0x000a to 0x0011 and provides space for 4 entries at a length of 32 bits each.

The register x+0 (0x….0000) contains the High-Word of the error.
The register x+1 (0x….0000) contains the Low-Word of the error.

If all four entries are occupied and another error occurs, this error will replace the oldest error in the Error Stack. The latest error is always displayed first.

```
------------------------------------------------
KUNBUS-IC – Error Stack
------------------------------------------------
<cr> = refresh; <Esc> = return

Entry 0: = 0x00000000
Entry 1: = 0x00000000
Entry 2: = 0x00000000
Entry 3: = 0x00000000
------------------------------------------------
>
```

4.7 Script Interpreter Status

In this menu you will find information about the current Script-Status.

```
------------------------------------------------
KUNBUS-IC - Script Interpreter Status
------------------------------------------------
<cr> = refresh; <Esc> = return

Port used by script: CDI
Execution status: Stopped
Loop counter: 0
------------------------------------------------
>
```

4.8 Emergency Menu

In the unlikely event of a serious error, you can open this menu.

- You can try to start the module by entering `"R"`.
- You can reset the EEPROM to the factory settings with the `"F"` key. This can be useful, for example, if a permanently stored value triggers a serious software error after a firmware update.

If the error cannot be remedied, please contact our support. Have the number of the error entry ready for this purpose.

```
--------------------------------------------------
KUNBUS-COM - Fatal Error Handler
--------------------------------------------------
A severe error occurred. Error Stack:

Entry 0: = 0x34030000
Entry 1: = 0x00000000
Entry 2: = 0x00000000
Entry 3: = 0x00000000
--------------------------------------------------
press 'R' for Reset or
press 'F' for Reset with Factory Defaults
--------------------------------------------------
>
```

9 Disposal

9.1 Dismantling and Disposal

This section contains important information explaining how to safely dismantle, replace and dispose of the IC-Modul correctly.

Dismantling

⚠ **DANGER**

Danger of electric shock

Before dismantling, make sure that the device in which your module is installed is no longer connected to the power supply.

➔ Remove the mains plug of the device from the power supply.

➔ Proceed in accordance with the device manufacturer's documentation to ensure that the device is disconnected from the power supply and no damage will result when dismantling.

⚠ **WARNING**

Fault due to electrostatic discharge

Observe all regulations necessary for working in electrostatically protected areas in order to avoid any faults on the module.

➔ Make sure that you yourself and the module are earthed.

✓ The device, in which your module is installed, has been disconnected from the power supply.

✓ You have removed the cover of the housing, if applicable, in accordance with the manufacturer's instructions.

◦ Release the safety lever, if applicable, and remove your module carefully out of the socket.

◦ If you want to insert a new module, proceed as described in section Installation [▸ 35].

Disposal

Dispose of any defective module at a collection point for waste electronic equipment in accordance with EU Directive 2002/96/EC. Do not dispose of the module in your household waste.

Dispose of any defective module in accordance with the local regulations for waste electronic equipment.

10 Technical data

10.1 Technical data

Dimensions

Length	25.2 mm
Width	45.4 mm
Height	14.2 mm
Weight	8 g

Environmental Conditions

Operating temperature	0 °C to +60 °C
Humidity	0% not 95%, non-condensing
Mechanical shock load	15 G
Permanent mechanical stress	5 G
Condensing	not allowed
Storage temperature	-30 °C to +85 °C

11 Appendix

11.1 Configuration via Modpoll

Prerequisite: You have made the SDI interface on the application connector suitable for a PC by your application hardware (e.g. a level converter).

Tip!: With the KUNBUS-IC Evaluation Board you have a Sub-D connector at your disposal with RS-232 levels that can be connected to a serial COM interface of the PC or to a serial USB converter.

Complete Application Configuration

To configure the module conveniently for application, we recommend that you make the setting adjustments with the aid of the Modpoll program supplied.

✓ To do this, open the command line prompt in the first step and then change to the directory where Modpoll is located. Drag and drop the modpoll.exe file into the input panel.

◦ Always begin new commands with the input "Modpoll".

⇨ Define your application by specifying the protocol type, register address, baudrate etc. We have compiled the relevant Modpoll commands for you in the following table.

Example for the Configuration of the SSC Output Register

modpoll –m rtu –r 0x1401 –t 4:hex –b 19200 COM1 0x1234

modpoll –m rtu –r 0x2401 –t 4:hex –b 19200 COM1

Overview of the Modpoll Commands

Start	modpoll.exe
Help	-h
Protocol selection	
Modbus ASCII Protocol	-m ascii
Modbus RTU Protocol	-m rtu
Modbus TCP Protocol	-m tcp
Nested Modbus RTU via TCP	-m enc
Addresses	
Slave address	-a #
Register address	-r #
Number of registers	-c #
I/O	
Discrete output (coil)	-t 0
Discrete input	-t 1
16-bit input register	-t 3
16-bit input register with hexadecimal display	-t 3:hex
32-bit integer data type in the input register table	-t 3:int
32-bit module 1000 data type in the input register table	-t 3:mod

32-bit float data type in the input register table	-t 3:float
16-bit output register (holding) with default value	-t 4
16-bit output register (holding) with hexadecimal display	-t 4:hex
32-bit integer data type in the output register (holding) table	-t 4:int
32-bit module 1000 data type in the output register (holding) table	-t 4:mod
32-bit float data type in the output register table	-t 4:float
Slave works onBig-Endian 32-Bit integers	-i
Slave works onBig-Endian 32-Bit floats	-f
Only poll once (instead of every second)	-1
Use Daniel/Enron single register 32-bit mode	-e
First reference is 0 instead of 1 (PDU addressing)	-0
Options for Modbus TCP	
TCP Port number (default value: 502)	-p #
Options for ASCII and Modbus RTU	
Baudrate (9600, 19200,... Default value: 9600)	-b #
Databits (7 or 8 for ANSCCI, 8 for RTU)	-d #
Stop bits (1 or 2, default value:1)	-s #
Parity: None	-p none
Parity: Even	-p even
Parity: Odd	-p odd
RS-485 Mode	-4 #
Timeout in seconds (0.01-10.0 default value: 0)	-o #