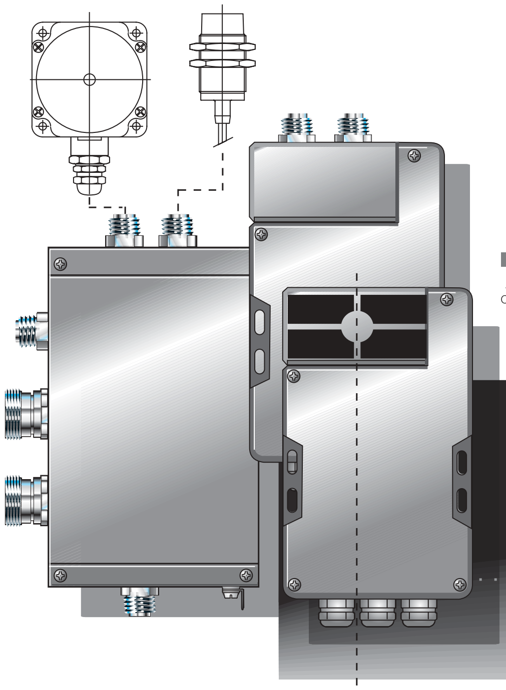
1



INTERBUS
Certified! No. 376

Manual

Electronic Identification Systems BIS
Processor BIS C-60_1
InterBus

Deutsch – bitte wenden!

2

No. 819 395 D/E • Edition 0105
Subject to modification.

http://www.balluff.de

Balluff GmbH
Schurwaldstrasse 9
73765 Neuhausen a.d.F.
Germany
Phone +49 (0) 71 58/1 73-0
Fax +49 (0) 71 58/50 10
E-Mail: balluff@balluff.de

Contents

Safety Considerations

Approved operation

Series BIS C-60_1 processors along with the other BIS C system components comprise an identification system and may only be used for this purpose in an industrial environment in conformity with Class A of the EMC Law.

Installation and Operation

Installation and operation should be carried out by technically trained personnel only. Unauthorized access and improper use will lead to loss of warranty and liability claims.

When installing the processor, consult the section on wiring diagrams carefully. Special caution must be used when wiring the processor to external controllers, particularly with respect to selection and polarity of the signals and power supply.

Only approved power supplies may be used with the processor. See the section on Technical Data for details.

Use and Checking

The relevant safety procedures must be followed when using the Identification System. In particular, steps must be taken to ensure that no danger to persons or equipment can arise should a fault occur in the Identification System.

This includes maintaining the published ambient operating conditions and regular checking of the functionality of the Identification System with all its associated components.

Fault Conditions

As soon as there is evidence that the Identification System is not functioning properly, it should be taken out of service and protected against unauthorized use.

Scope

This description is valid for series BIS C-6001-023...03-KL2 processors and both the ST8 and ST9 versions of series BIS C-6021-023-050-03-ST_.

Introduction
BIS C Identification System

This manual is designed to assist the user in setting up the control program and installing and starting up the components of the BIS C-60_1 Identification System, and to assure rapid, trouble-free operation.

Principles

The BIS C-60_1 Identification System belongs in the category of

non-contact systems for reading and writing.

This dual function permits applications for not only transporting information in fixed-programmed code tags, but also for gathering and passing along up-to-date information as well.

☞ If 2 read/write heads are connected to a BIS C-60_1 processor, both heads can be operated independently of each other. This means for example that you can read a code tag from one head while writing to another code tag at the other head.

Applications

Some of the notable areas of application include

– **for controlling material flow in production processes**
  (e.g. in model-specific processes),
  for workpiece conveying in transfer lines,
  in data gathering for quality assurance ,
  for gathering safety-related data,
– **in tool coding and monitoring;**
– **in equipment organization;**
– **in storage systems for monitoring inventory movement;**
– **in transporting and conveying systems;**
– **in waste management for quantity-based fee assessment.**

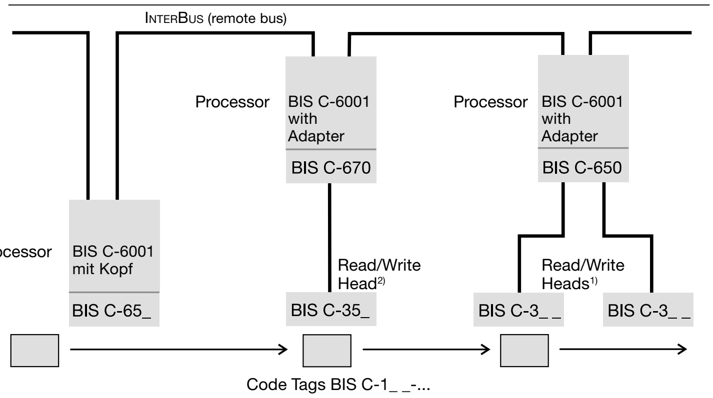
Introduction
BIS C Identification System

System Components

The main components of the BIS C Identification System are

– **Processor,**
– **Read/Write Heads and**
– **Code Tags**.

Configuration with BIS C-6001 processor



Schematic representation of an Identification System (example)

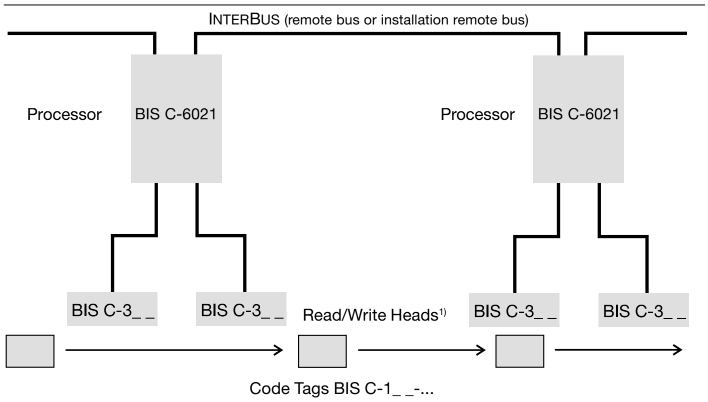
1) BIS C-3_ _ series, except BIS C-350 and -352

2) BIS C-350 or -352 only

7

Introduction
BIS C Identification System

Configuration with BIS C-6021 processor



Schematic representation of an Identification System (example)

[1)] BIS C-3_ _ series, except BIS C-350 and -352

8

BIS C-60_1 Processor
Basic knowledge for application

Selecting System Components

The **BIS C-6001** processor has a plastic housing. Connections are made through a terminal strip, with the cables secured by PG fittings. A single read/write head from BIS C-65_ series can be directly mounted to the processor, which creates a compact unit. If the BIS C-650 adapter is attached instead of the BIS C-65_ read/write head, two read/write heads may be cable connected. If the BIS C-670 adapter is attached, one read/write head may be cable connected.

The **BIS C-6021** processor has a metal housing. Connection is made through round connectors. Two read/write heads can be cable connected to the processor.

Series BIS C-60_1 processors have in addition a digital input. The input has various functions depending on the configuration (see Parametering).

Whether the compact version of the processor with integrated read/write head makes sense or whether the external solution is preferred depends primarily on the spatial arrangement of the components. There are no functional limitations. All read/write heads are suitable for both static and dynamic reading. Distance and relative velocity are based on which code tag is selected. Additional information on the read/write heads in series BIS C-65_ and series BIS C-3_ _ including all the possible code tag/read-write head combinations can be found in the manuals for the respective read/write heads.

The system components are electrically supplied by the processor. The code tag represents an free-standing unit and needs no line-carried power. It receives its energy from the read/write head. The latter constantly sends out a carrier signal which supplies the code head as soon as the required distance between the two is reached. The read/write operation takes place during this phase. Reading and writing may be dynamic or static.

BIS C-60_1 Processor
Basic knowledge for application

Control Function

The processor writes data from the host system to the code tag or reads data from the tag through the read/write head and prepares it for the host system. Host systems may include:

– **a host computer (e.g. industrial PC) or**
– **a programmable logic controller (PLC)**

Data checking

When sending data between the read/write head and the code tag a procedure is required for recognizing whether the data were correctly read or written.

The processor is supplied with standard Balluff procedure of double reading and comparing. In addition to this procedure a second alternative is available: CRC-16 data checking.

Here a test code is written to the code tag, allowing data to be checked for validity at any time or location.

Advantages of CRC-16	Advantages of double reading
Data checking even during the non-active phase (CT outside read/write head zone).	No bytes on the code tag need to be reserved for storing a check code.
Shorter read times since each page is read only once.	Shorter write times since no CRC needs to be written.

Since both variations have their advantages depending on the application, the user is free to select which method of data checking he wishes to use (see Parametering on 🗋 26 and 🗋 31).

☞ It is not permitted to operate the system using both check procedures!

BUS interface: INTERBUS

INTERBUS

Communication between the BIS C-60_1 processor and the host system is via INTERBUS.

The INTERBUS system consists of three components:

– the wiring module (rack card for industrial PC or PLC),
– Bus terminal as network node and/or
– the I/O modules (here the BIS C-60_1 processor).

Depending on the wiring module up to a maximum of 63 BIS C-60_1 processors can be connected.

The BIS C-6001 processor is used as a remote bus station. The BIS C-6021 can be used as a remote bus or installation bus station.

☞ **Important hints for use with PLC:**

In some control systems the PROFIBUS-DP data area is not synchronously transmitted with the updating of the input/output content. If more than 2 bytes of data are sent, a mechanism must be used which guarantees that the data in the PLC and the data in the BIS C are always identical!

2nd alternative: Set 2nd bit header
Data exchange between PLC and BIS is controlled by the so-called bit header. This is always the first byte of the respective read/write head in the data buffer. This bit header exists both in the input range (data from BIS to the PLC) and in the output range (data from the PLC to the BIS). llf this bit header is also sent as the last byte, a comparison of these two bytes can be used to guarantee the consistency of the transmitted data.

In this method the PLC cycle is unaffected nor is the bus access time changed. All that is required is that a byte in the data buffer be used for the 2nd bit header instead of for user data.

This 2nd alternative is the Balluff recommended setting (factory default).

BUS interface: INTERBUS

Address setting is done on the module (not on the I/O modules), i.e. not on the BIS C-60_1 processor). There are two types of addressing possible:

1. logical addressing, and
2. physical addressing.

Logical Addressing

Logical addressing permits free addressing of each module.

Advantage: high security and flexibility;
Disadvantage: more difficult at setup.

For the BIS C-60_1 Identification System use:

I/O Module Type	IDENT-No.	IN-Address (Byte)	OUT-Address (Byte)
Processor BIS C-60_1	03	16	16

Physical Addressing

Physical addressing is rigidly fixed to the system configuration. The address of each module depends on its location in the system.

Advantage: easy to configure at setup;
Disadvantage: changes in module location when power was off are recognized upon initialization, but are not made known to the user.

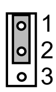
Compatibility with BIS C-6_1 processor

Setting compatibility

Compatibility with the BIS C-6_1 processors is established using terminal X5 and a jumper.

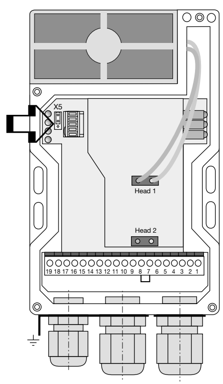
☞ If the BIS C-60_1 processor is set to be compatible with the BIS C-601 or BIS C-621, all settings for data exchange must be made as described in the sections on parametering, function description, protocol sequence and LED display in the user's manual for the BIS C-6_1 processor! This user's manual can be mailed on request, or you may download it from the Internet at `www.balluff.de`.

Jumper setting Terminal X5	Processor compatible with BIS C-6_1
1-2	no
2-3	yes



In the illustration compatibility with the BIS C-6_1 is not set.

To open the cover of the BIS C-6001 processor, see 58, and for BIS C-6021 see 71.



Terminal X5
(with cover removed)

Function Description
Communication with the processor

Basic Procedure

Communication between the host system and the processor takes place using a fixed protocol sequence. Data integrity from the control to the processor and vice-versa is indicated by a control bit. This bit is used to implement a handshake between the control and the processor.

Following is a simplified representation of the sequence of a job sent from the control to the processor:

1. The control sends a command designator to the processor together with the associated command parameters and sets a bit (AV bit). This bit indicates to the processor that the transmitted data are valid and that the job is now beginning.
2. The processor takes the job and sets a bit (AA bit), which indicates this to the control.
3. If an additional exchange of data between the control and the processor is required to carry out the job, each uses a bit (TI bit and TO bit) to indicate that the control / processor is now ready for additional data exchange or has accepted the received data.
4. Once the processor has carried out the job correctly, it sets a bit (AE bit).
5. Once the control has accepted all the important data, it indicates this to the processor by resetting the bit that was set at the beginning (AV bit).
6. The processor now in turn sets all the control bits that were set during the sequence (AA bit, AE bit) and is ready for the next job.

Function description
In- and output buffer on INTERBUS

Input and Output Buffers

To transmit the control bit, commands and data between the BIS C-60_1 processor and the host system, the latter must provide two fields. These two fields are:

– **the output buffer**
  For the control bit (bit header) and controller commands sent to the BIS identification system,
  For the data to be written, and
  For configuring the BIS C-60_1 processor
– **the input buffer**
  For the control bit (bit header) of the BIS C-60_1 processor,
  For the data to be read,
  For the ID's and error codes coming from the BIS identification system, and
  For reading out the configuration data.

The total buffer size of the BIS C-60_1 is 16 bytes for the input buffer and 16 bytes for the output buffer. This total buffer size is divided into 2 sectors:

Buffer sector 1 for Read/Write Head 1: 8 bytes input buffer, 8 bytes output buffer
Buffer sector 2 for Read/Write Head 2: 8 bytes input buffer, 8 bytes output buffer

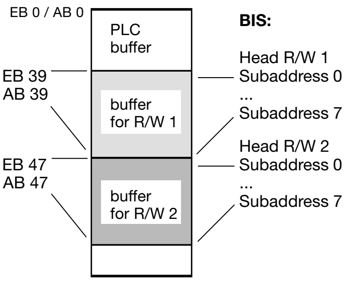
Please note the basic procedure on 13 and 28...34 and the examples on pages 35...52.

Function description
In- and output buffer on INTERBUS

Input and Output Buffers (continued)

Example: Using a PLC, the buffer sector for the BIS C-60_1 will start at input byte EB32 and output byte AB 32.

Memory map:	**PLC:**
Read/write head 1 (R/W1):	
Input buffer	from EB 32 to EB 39
Output buffer	from AB 32 to AB 39
Read/write head 2 (R/W2):	
Input buffer	from EB 40 to EB 47
Output buffer	from AB 40 to AB 47



Note that these buffers can be in two different sequences depending on the type of control.

The following description is based on sequence 1!

Sequence 1	Sequence 2
Subaddress 00	Subaddress 01
01	00
02	03
03	02
04	05
05	04
06	07
07	06

Please note the basic procedure on 13 and 28...34 and the examples on pages 35...52.

Function Description
Output buffer, configuration and explanation

Configuration of the output buffer for one (1) read/write head

Subaddress \ Bit No.	7	6	5	4	3	2	1	0	
00Hex = Bit Header	CT	TI				GR		AV	Bit Name
01Hex	Command Designator			or	Data	or	Config. 1st byte		
02Hex	Start Address (Low Byte)			or	Data	or	Config. 2nd Byte		
03Hex	Start Address (High Byte)			or	Data	or	Config. 3rd Byte		
04Hex	No. of Bytes (Low Byte)			or	Data	or	Config. 4th Byte		
05Hex	No. of Bytes (High Byte			or	Data	or	Config. 5th Byte		
06Hex					Data	or	Config. 6th Byte		
07Hex	2nd Bit Header (as above)			or	Data				

Description of Output Buffer

Sub-address	Bit Name	Meaning	Function Description	
00Hex Bit Header	CT	Code tag type	Select code tag type:	for code tag type:
		0	32 Byte block size	BIS C-1_ _-02, -03, -04, -05
		1	64 Byte block size	BIS C-1_ _-10, -11, -30
	TI	Toggle-Bit In	for reading:	Controller is ready to accept new/additional data.
			for writing:	Controller has prepared new/additional data.
	GR	Ground state	Causes the BIS system to go to ground state.	
	AV	Command	Signals the ID system that a job is waiting.	

(continued next)

Please note the basic procedure on 13 and 28...34 and the examples on pages 35...52.

Function Description
Output buffer, configuration and explanation

Description of Output Buffer (continued)

Sub-address	Meaning	Function Description
01Hex	Command designator	
	00Hex	No command present
	01Hex	Read code tag
	02Hex	Write to code tag
	04Hex	Configure processor
	05Hex	Read configuration data
	06Hex	Store program in the EEPROM for the Mixed Data Access function
	07Hex	Store the start address for the Auto-Read function in the EEPROM
	12Hex	Initialize the CRC-16 data check
	21Hex	Read code tag using Mixed Data Access function (corresponding to the program stored in the EEPROM)
	22Hex	Write to code tag using the Mixed Data Access function (corresponding to the program stored in the EEPROM)
or	Configuration 1st byte	
	00Hex	Default value (factory setting). Changes depending on the configuration.
or:	Data	for writing to the code tag

(continued next)

Please note the basic procedure on 13 and 28...34 and the examples on pages 35...52.

Function Description
Output buffer, configuration and explanation

Description of Output Buffer (continued)

Sub-address	Meaning	Function Description
02Hex	Start address (Low Byte)	Address at which reading from or writing to the code tag begins. (The Low Byte includes the address range from 0 to 255).
or:	Start address (Low Byte)	Address for the Auto-Read function, starting at which the code tag is to be read. The value is stored in the EEPROM. (The Low Byte covers the address range from 0 to 255).
or:	Program No	Number of the program to be stored in the EEPROM in conjunction with command ID 06Hex for Mixed Data Access function.
or:	Program No.	Number of the program stored in the EEPROM for read or write operations in conjunction with command ID 21Hex or 22Hex for the Mixed Data Access function.
or	Configuration 2nd byte	
	80Hex	Default value (factory setting) Changes depending on the configuration.
or:	Data	for writing to the code tag.

(continued next)

Please note the basic procedure on 13 and 28...34 and the examples on pages 35...52.

Function Description
Output buffer, configuration and explanation

Description of Output Buffer (continued)

Sub-address	Meaning	Function Description
03Hex	Start address (High Byte)	Address for reading from or writing to the code tag. (The High Byte is additionally used for the address range from 256 to 8,191)
or:	Start address (High Byte)	Address for the Auto-Read function, starting at which the code tag is to be read. The value is stored in the EEPROM. (The High Byte is also required for the address range from 256 to 8,191).
or:	Configuration 3rd byte	
	00Hex	Default value (factory setting) **This value must not be changed!**
or:	Data	for writing to the code tag
04Hex	No. of bytes (Low Byte)	Number of bytes to read or write beginning with the start address (the Low Byte includes from 1 to 255 bytes).
or:	Configuration 4th byte	
	82Hex	Default value (factory setting) Changes depending on the configuration.
or:	Data	for writing to the code tag

(continued next 🗋)

Please note the basic procedure on 🕮 13 and 28...34 and the examples on pages 🕮 35...52.

Function Description
Output buffer, configuration and explanation

Description of Output Buffer (continued)

Sub-address	Meaning	Function Description
05Hex	No. of bytes (High Byte)	Number of bytes to read or write beginning with the start address (the High Byte is additionally used for the range between 256 and 8,191 bytes).
or:	Configuration 5th byte	
	00Hex	Default value (factory setting) Changes depending on the configuration.
or:	Data	for writing to the code tag.
06Hex	Configuration 6th byte	
	00Hex	Default value (factory setting) **This value must not be changed!**
or:	Data	for writing to the code tag.
07Hex	2nd Bit header	The data are valid if the 1st and 2nd bit header are identical.
or:	Data	for writing to the code tag.

Please note the basic procedure on 🕮 13 and 28...34 and the examples on pages 🕮 35...52.

Function Description
Input buffer, configuration and explanation

Configuration of the input buffer for one (1) Read/Write head

Subaddress \ Bit No.	7	6	5	4	3	2	1	0	
00Hex = Bit Header	BB	HF	TO	IN	AF	AE	AA	CP	Bit name
01Hex	Error code		or		Data	or	Config. 1st byte		
02Hex	Data					or	Config. 2nd Byte		
03Hex	Data					or	Config. 3rd Byte		
04Hex	Data					or	Config. 4th Byte		
05Hex	Data					or	Config. 5th Byte		
06Hex	Data					or	Config. 6th Byte		
07Hex	2nd Bit Header (as above)					or	Data		

Description of Input Buffer

Sub-address	Bit Name	Meaning	Function Description
00Hex	BB	Ready	The BIS Identification System is in the Ready state.
Bit Header	HF	Head Error	Cable break from read/write head or no read/write head connected.
	TO	Toggle-Bit Out	for read: BIS has new/additional data ready. for write: BIS is ready to accept new/additional data.

(continued on next 🗋)

Please note the basic procedure on 🕮 13 and 28...34 and the examples on pages 🕮 35...52.

Function Description
Input buffer, configuration and explanation

Description of Input Buffer (continued)

Sub-address	Bit Name	Meaning	Function Description
00Hex	(continued)		
Bit Header	IN	Input	If the parameter "Input IN" is 1, this bit indicates the state of the Input.
	AF	Command Error	The command was incorrectly processed or aborted.
	AE	Command end	The command was finished without error.
	AA	Command start	The command was recognized and started.
	CP	Codetag Present	Code tag present within the active zone of the read/write head.

Sub-address	Meaning	Function Description
01Hex	Error code	Error number is entered if command was incorrectly processed or aborted. Only valid with AF bit!
	00Hex	No error.
	01Hex	Reading or writing not possible because no code tag is present in the active zone of a read/write head.
	02Hex	Read error.
	03Hex	Code tag was removed from the active zone of the read/write head while it was being read.
	04Hex	Write error.
	05Hex	Code tag was removed from the active zone of the read/write head while it was being written.
	07Hex	AV bit is set but the command designator is missing or invalid.
	or:	Number of bytes is 00Hex.

(continued on next 🗋)

Please note the basic procedure on 🕮 13 and 28...34 and the examples on pages 🕮 35...52.

Function Description
Input buffer, configuration and explanation

Description of Input Buffer (continued)

Sub-address	Meaning	Function Description
01Hex	Error code (continued)	
	09Hex	Cable break to select read/write head, or head not connected.
	0CHex	The EEPROM cannot be read/programmed.
	0DHex	Faulty communication with the code tag.
	0EHex	The CRC of the read data does not coincide with the CRC of the code tag.
	0FHex	Contents of the 1st and 2nd bit header (1st and last bytes) of the output buffers are not identical (2nd bit header must be served).
	11Hex	Invoking a function that is not possible, since the processor is in "compatible with BIS C-6_1" mode.
or:	Configuration 1st byte	
	00Hex	Default value (factory setting). Changes depending on the configuration.
or:	Data	Data which was read from the code tag.

(continued on next)

Please note the basic procedure on pages 13 and 28...34 and the examples on pages 35...52.

Function Description
Input buffer, configuration and explanation

Description of Input Buffer (continued)

Sub-address	Meaning	Function Description
02Hex	Configuration 2nd byte	
	80Hex	Default value (factory setting). Do not change!
or:	Data	Data which was read from the code tag.
03Hex	Configuration 3rd byte	
	00Hex	Default value (factory setting). Changes depending on the configuration.
or:	Data	Data which was read from the code tag.
04Hex	Configuration 4th byte	
	82Hex	Default value (factory setting). Changes depending on the configuration.
or:	Data	Data which was read from the code tag.
05Hex	Configuration 5th byte	
	00Hex	Default value (factory setting). Changes depending on the configuration.
or:	Data	Data which was read from the code tag.
06Hex	Configuration 5th byte	
	00Hex	Default value (factory setting). Do not change!
or:	Data	Data which was read from the code tag.
07Hex	2nd Bit header	The data are valid if the 1st and 2nd bit headers are in agreement.
or:	Data	Data which was read from the code tag.

Please note the basic procedure on pages 13 and 28...34 and the examples on pages 35...52.

Function Description
Configuring the BIS C-60_1 processor

Configuration, Overview

The following functions can be activated / deactivated through the configuration:

- **CRC-16** data check**:**
  If this function is activated, the correctness of the read or written data is ensured by a CRC-16 data check (see 🗋 9).
- **Simultaneous data transmission for both read/write heads:**
  With simultaneous data transmission shorter read/write times can be achieved depending on the amount of data to be read/written and the type of controller.
- **Dynamic operation on Read/Write Head 1 or 2:**
  If dynamic operation is parametered, a read/write job can be sent even though there is no code tag in the active zone of the head. As soon as a code tag passes by the head, the command is immediately carried out.
- **"Auto-Read" for Read/Write Head 1 or 2:**
  If this function is activated, the processor reads out the first (max. 31) bytes from the code tag starting at a defined start address as soon as the tag enters the active zone of the read/write head. The start address must first have been stored in the processor's EEPROM with the command ID 07_{Hex}.
- **2nd bit header at end of in- and output buffer:**
  The 2nd bit header (factory setting) prevents data from being accepted by the bus as long as it is not fully updated.
- **Display state of the digital input in the bit header of the input buffer:**
  If this function is activated, the IN-bit displays the state of the digital input of the processor: IN = 0 → digital input low; IN = 1 → digital input high
- **Reset BIS C-60_1 processor through the digital input:**
  If this function is activated, the processor is reset when the digital input is set to high.

Function Description
Configuring the BIS C-60_1 processor

Configuration

The BIS C-60_1 processor is configured by the controller using the output buffer. The configuration data are arranged within 6 configuration bytes that are sent to the BIS C-60_1 processor using the command identifier 04_{Hex} (see Example 13 on 🗋 51). Command identifier 05_{Hex} is used to read out the current device configuration (see Example 14 on 🗋 52).

⚠ To input the configuration, all 6 bytes must be entered in Hex. Only the named bits are permitted to be changed. If any of the other bits are changed, there is no assurance that the BIS C-60_1 will function properly.

The default values of the 6 bytes are (factory setting):

	1st byte	**2nd byte**	3rd byte	**4th byte**	**5th byte**	6th byte
Hex	00	80	00	82	00	00
Binary	00000000	10000000	00000000	10000010	00000000	00000000
These are used for configuration:	**bit 5**, **bit 3**	**bit 5**, **bit 4**		**bit 8**, **bit 7**, **bit 2**	**bit 5**, **bit 4**	

Having the following functions:

1st byte, bit 5	Activate CRC-16 data checking
1st byte, bit 3	Activate simultaneous data transmission **for both read/write heads**
2nd byte, bit 5	Dynamic mode on read/write head 1 (for effects on read/write times, see 🗋 53/54)
2nd byte, bit 4	Activate Auto-Read function starting at specified address after CT-Present for Head 1 (the amount data read is 6 bytes for a double bit header or 7 bytes for a single bit header)

Bit state: 0 = no
1 = yes

Function Description
Configuring the BIS C-60_1 processor

Configuration
(continued)

Bit state: 0 = no
1 = yes

4th byte, bit 8,	Arrange a 2nd bit header at the end of the input and output buffers.
4th byte, bit 7	Display state of the digital input in the bit header of the input buffers
4th byte, bit 2	Reset the BIS C-60_1 processor through the digital input
5th byte, bit 5	Dynamic mode on read/write head 2 (for effects on read/write times, see 53/54)
5th byte, bit 4	Activate Auto-Read function starting at specified address after CT-Present for Head 2 (the amount data read is 6 bytes for a double bit header or 7 bytes for a single bit header)

Function Description
Processing code tags

Reading and writing

To carry out a read or write job, the code tag must be located in the active zone of the read/write head.

A read/write job has the following sequence (see examples on 37...43):

1. The host sends to the output buffer:
   - the command designator to subaddress 01_{Hex},
   - the start address for reading or writing to subaddress 02_{Hex}/03_{Hex},
   - the number of bytes for reading or writing to subaddress 04_{Hex}/05_{Hex},
   - the CT bit according to the code tag type (block size),
   - and sets the AV bit in the bit header to high.
2. The processor:
   - takes the request (AA in the bit header of the input buffer to high),
   - begins to transport the data;
     read = from code tag to input buffer,
     write = from output buffer to code tag.
     Larger data quantities are sent in blocks:
     block size of 6 bytes with 2nd bit header,
     block size of 7 bytes without 2nd bit header.
     The toggle bits in the two bit headers are used as a kind of handshaking between the host and the BIS C-60_1 processor.
3. The processor has processed the command correctly (AE bit in the bit header of the input buffer). If an error occurred during execution of the command, an error number will be written to subaddress 01_{Hex} of the input buffer and the AF bit in the bit header of the input buffer will be set.

Function Description
Processing code tags

Codetag Present

As soon as the code tag enters the active one of the read/write head, the processor indicates this by setting the CP bit (Codetag present).

> ☞ To accelerate the reading of small amounts of data, the ID system makes the first bytes of the code tag available in the input buffer of the respective read/write head as soon as the tag is detected (6 bytes for double bit header, 7 bites for single bit header).

The data are only valid after the rising edge of the CP bit in the bit header of the input buffer. They remain valid until the falling edge of the CP bit, or until the controller issues a new job.

Auto-Read

If the Auto-Read function is activated, data are automatically read (6 bytes for a double bit header / 7 bytes for a single bit header) beginning with a start address as soon as a code tag is recognized. The read process begins at the start address that was specified by command identifier 07_{Hex}. Each head can have its own start address assigned. The start addresses are stored in the EEPROM of the BIS C-60_1 processor.

> ☞ To obtain correct data output, use command identifier 07_{Hex} for each partial buffer Head 1 and/or Head 2.
>
> If the Auto-Read function is not activated, the processor runs in standard mode and sends 6 bytes for a double bit header or 7 bytes for a single bit header starting with code tag address 0.

Function Description
Processing code tags

Reading and writing in dynamic mode

In normal operation a read/write job is rejected by the processor BIS C-60_1 by setting the AF bit and an error number if there is no code tag in the active zone of the read/write head. If dynamic mode is configured, the processor accepts the read/write job and stores it. When a code tag is recognized, the stored job is carried out.

Reading and writing with simultaneous data transmission

Reading without simultaneous data transmission: In the case of a read job the processor first reads our all requested data from the code tag after receiving the start address and the desired number of bytes, and then sets the AE bit. Then the data read from the code tag are written to the input buffer. In the case of larger data amounts this is done in blocks, controlled by the handshake with the toggle bits as described on 🗋 28.

Reading with simultaneous data transmission: In the case of a read job the processor begins to send the data to the input buffer as soon as the first 6 bytes (or 7 bytes for a single bit header) have been read from the code tag beginning with the start address, and indicates this by inverting the TO bit. As soon as the controller inverts the TI bit, the processor sends the data that have been read in the meantime to the input buffer. This is repeated until the processor has read all the desired data from the code tag. Now the processor sets the AE bit and outputs the remaining data to the input buffer.

Writing without simultaneous data transmission: In the case of a write job the processor waits until it has received all the data that need to be written from the controller. Only then are the data written to the code tag as described on 🗋 28.

Writing with simultaneous data transmission: In the case of a write job the processor begins to write the data to the code tag as soon as it has received the first data to be written from the controller's output buffer. Once all the data have been written to the code tag, the AE bit is set.

Function Description
Processing code tags

CRC initialization

To be able to use the CRC check, the code tag must first be initialized with the command identifier 12Hex (see 35). The CRC initialization is used like a normal write job. The latter is rejected (with an error message) if the processor recognizes that the code tag does not contain the correct CRC. Code tags as shipped from the factory (all data are 0) can immediately be programmed with a CRC check.

If CRC-16 data checking is activated, a special error message is output to the interface whenever a CRC error is detected.

If the error message is not caused by a failed write request, it may be assumed that one or more memory cells on the code tag is defective. That code tag must then be replaced.

If the CRC error is however due to a failed write request, you must reinitialize the code tag in order to continue using it.

The checksum is written to the code tag as a 2-byte wide datum. Two bytes per page are 'lost', i.e., the page size becomes 30 bytes or 62 bytes depending on code tag type (setup of page size see 16). This means that the actual usable number of bytes is reduced:

Code tag type		Usable bytes
128 bytes	=	120 bytes
256 bytes	=	240 bytes
511 bytes *)	=	450 bytes
1023 bytes *)	=	930 bytes
2047 bytes *)	=	1922 bytes
2048 bytes	=	1984 bytes
8192 bytes	=	7936 bytes

*) The last code tag page for these EEPROM-based code tags is not available.

Function Description
Processing code tags

Mixed Data Access

Small read/write programs can be stored in the BIS C-60_1 processor's EEPROM.

The Mixed Data Access function is useful when the required information is stored on the code tag at various addresses. This function makes it possible to read out this “mixed”, i.e. non-contiguously stored data from the code tag in a single procedure and using just one command.

Up to 10 programs with up to 25 instructions can be stored. Each program instruction contains a “start address” and a “number of bytes” specification. The amount of data for reading may not exceed 2 kB.

Storing a program:
The command identifier 06Hex is used to send the read/write program to the BIS C-60_1 processor. One program per command can be stored. All 25 program records plus an additional 2 bytes with FFHexFFHex as a terminator must always be sent. This means a total of **104 bytes** of information per program must be sent (including the command identifier and program number).

☞ The individual program records must all be contiguous. They must be sent one after the other and be terminated with FFHexFFHex as a terminator. It is recommended that the remaining, unused memory sector be filled with FFHexFFHex.

If an address range is selected twice, the data will also be output twice.

Function Description
Processing code tags

Mixed Data Access (cont.)

The following shows the structure of a program:

Program structure	Subaddress	Value	Range
Command designator	01_{Hex}	06_{Hex}	
1. Program record			
Program number	02_{Hex}	01_{Hex}	01_{Hex} to $0A_{Hex}$
1st data record:			
Start address Low Byte	03_{Hex}		
Start address High Byte	04_{Hex}		
Number of bytes Low Byte	05_{Hex}		
Number of bytes High Byte	06_{Hex}		
2nd data record:			
...			
25th data record:			
Start address Low Byte	03_{Hex}		
Start address High Byte	04_{Hex}		
Number of bytes Low Byte	05_{Hex}		
Number of bytes High Byte	06_{Hex}		
Terminator	FF_{Hex} FF_{Hex}		

To store a second program, repeat this process.

The procedure for writing these settings to the EEPROM is described in the 9th example on 45...47.

Replacing the EEPROM is described on 66 for BIS C-6001 and on 77 for BIS C-6021.

Function Description
Processing code tags

Read from code tag, with program

The command identifier 21_{Hex} can be used to read out the program records stored in the program from the code tag. The user must document exactly which data are to be read from where and with what number of bytes for the respective program (see example 10 on 48)

Write to code tag, with program

The command identifier 22_{Hex} can be used to write the program records stored in the program to the code tag. The user must document exactly which data are to be written from where and with what number of bytes for the respective program (see example 11 on 49)

Function Description
Examples for protocol sequence

Example No. 1

For configuring with double bit header!

Initializing the code tag for the CRC-16 data checking

The processing of this command is similar to a write command. Start address and number of bytes have to correspond to the maximum number of data to be used.
In this example the complete memory range of a code tag with 128 bytes shall be used (BIS C-1_ _-03/L with 32 byte block size). Because 2 bytes are used for the CRC only 120 bytes can be used as data bytes, hence: start address = 0, number of bytes = 120.

Host:

1.) Process subaddresses of the output buffer in order shown:

01Hex	Command designator 12Hex
02Hex	Start address 00Hex
03Hex	Start address 00Hex
04Hex	No. of bytes 78Hex
05Hex	No. of bytes 00Hex
00Hex/07Hex	Set AV-Bit, CT-Bit to 0

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in order shown:

00Hex/07Hex	Set AA-Bit, invert TO-Bit

3.) Process subaddresses of the output buffer:

01...06Hex	Enter first 6 bytes of data
00Hex/07Hex	Invert TI-Bit

4.) Process subaddresses of the output buffer:

01...06Hex	Copy first 6 data bytes

Process subaddress of the input buffer:

00Hex/07Hex	Invert TO-Bit

5.) Process subaddresses of the output buffer:

01...06Hex	Enter the second 6 data bytes
00Hex/07Hex	Invert TI-Bit

6.) Process subaddresses of the output buffer:

01...06Hex	Copy second 6 data bytes

Process subaddress of the input buffer:

00Hex/07Hex	Invert TO-Bit

...To be continued until the complete memory range is written. See next 🗋.

Function Description
Examples for protocol sequence

Example No. 1
(continued)

For configuring with double bit header !

41.) Process subaddresses of the output buffer:

01...06Hex	Enter the remaining data byte
00Hex/07Hex	Invert TI-Bit

42.) Process subaddresses of the output buffer:

01...06Hex	Copy the remaining data byte

Process subaddress of the input buffer:

00Hex/07Hex	Set AE-Bit

43.) Process subaddresses of the output buffer:

00Hex/07Hex	Reset AV-Bit

44.) Process subaddresses of the input buffer:

00Hex/07Hex	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 2

For configuring with double bit header!

Read 17 bytes starting at code tag address 10 (code tag type with 32 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in order shown:

01 Hex	Command designator 01 Hex
02Hex	Start address Low Byte 0A Hex
03Hex	Start address High Byte 00 Hex
04 Hex	No. of bytes Low Byte 11 Hex
05Hex	No. of bytes High Byte 00 Hex
00 Hex/07 Hex	CT-Bit to 0 (32 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in order shown:

00Hex/07Hex	Set AA-Bit
01...06Hex	Enter first 6 bytes of data
00Hex/07Hex	Set AE-Bit

3.) Process subaddresses of the input buffer:

01...06Hex	Copy first 6 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Invert TI-Bit

4.) Process subaddresses of the input buffer:

01...06Hex	Enter the second 6 data bytes
00Hex/07Hex	Invert TO-Bit

5.) Process subaddresses of the input buffer:

01...06Hex	Copy second 6 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Invert TI-Bit

6.) Process subaddresses of the input buffer:

01...05Hex	Enter the remaining 5 data bytes
00Hex/07Hex	Invert TO-Bit

7.) Process subaddresses of the input buffer:

01...05Hex	Copy the remaining 5 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Reset AV-Bit

8.) Process subaddresses of the input buffer:

00Hex/07Hex	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 3 like 2nd example but with simultaneous data transmission

For configuring with double bit header!

Read 17 bytes starting at code tag address 10, with simultaneous data transmission (code tag type with 32 byte block size)**:**

While the read job is being carried out and as soon as the input buffer is filled, the first data are sent. The AE bit is not set until the "Read" operation is completed by the processor.

The reply "Job End" = AE bit is reliably set no later than before the last data are sent. The exact time depends on the requested data amount, the input buffer size and the timing of the controller. This is indicated in the following by the note *Set AE-Bit* (in italics).

Host:

1.) Process subaddresses of the output buffer in order shown:

01 Hex	Command designator 01 Hex
02Hex	Start address Low Byte 0A Hex
03Hex	Start address High Byte 00 Hex
04 Hex	No. of bytes Low Byte 11 Hex
05Hex	No. of bytes High Byte 00 Hex
00 Hex/07 Hex	CT-Bit to 0 (32 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in order shown:

00Hex/07Hex	Set AA-Bit
01...06Hex	Enter first 6 bytes of data
00Hex/07Hex	Invert TO-Bit
00Hex/07Hex	*Set AE-Bit*

3.) Process subaddresses of the input buffer:

01...06Hex	Copy first 6 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Invert TI-Bit

4.) Process subaddresses of the input buffer:

01...06Hex	Enter the second 6 data bytes
00Hex/07Hex	Invert TO-Bit
00Hex/07Hex	*Set AE-Bit*

Continued on next 🗋.

Function Description
Examples for protocol sequence

Example No. 3
(continued)

like 2nd example but with simultaneous data transmission

For configuring with double bit header !

5.) Process subaddresses of the input buffer:

01...06Hex	Copy second 6 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Invert TI-Bit

6.) Process subaddresses of the input buffer:

01...05Hex	Enter the remaining 5 data bytes
00Hex/07Hex	Invert TO-Bit
00Hex/07Hex	*Set AE-Bit*

7.) Process subaddresses of the input buffer:

01...05Hex	Copy the remaining 5 data bytes

Process subaddress of the output buffer:

00Hex/07Hex	Reset AV-Bit

8.) Process subaddresses of the input buffer:

00Hex/07Hex	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 4

For configuring with double bit header !

Read 30 bytes starting at code tag address 10 with read error
(code tag type with 64 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in the order shown:

01Hex	Command designator 01Hex
02Hex	Start address Low Byte 0AHex
03Hex	Start address High Byte 00 Hex
04Hex	No. of bytes Low Byte 1E Hex
05Hex	No. of bytes High Byte 00 Hex
00Hex/07Hex	Set CT-Bit to 1 (64 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

If an error occurs right away:

00Hex/07Hex	Set AA-Bit
01Hex	Enter error number
00Hex/07Hex	Set AF-Bit

3.) Process subaddress of the input buffer:

01Hex	Copy error number

Process subaddress of the output buffer:

00Hex/07Hex	Reset AV-Bit

4.) Process subaddress of the input buffer:

00Hex/07Hex	Reset AA-Bit and AF-Bit

Function Description
Examples for protocol sequence

Example No. 5, like 4th example but with simultaneous data transmission

For configuring with double bit header!

Read 30 bytes starting at code tag address 10, with read error and simultaneous data transmission (code tag type with 64 byte block size)**:**

If an error occurs, the AF bit is set instead of the AE-Bit, with a corresponding error number. When the AF-BIT is set the job is interrupted and declared to be ended.

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 01_{Hex}
02_{Hex}	Start address Low Byte $0A_{Hex}$
03_{Hex}	Start address High Byte 00_{Hex}
04_{Hex}	No. of bytes Low Byte $1E_{Hex}$
05_{Hex}	No. of bytes High Byte 00_{Hex}
$00_{Hex}/07_{Hex}$	Set CT-Bit to 1 (64 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

If an error occurs right away:

$00_{Hex}/07_{Hex}$	Set AA-Bit
01_{Hex}	Enter error number
$00_{Hex}/07_{Hex}$	Set AF-Bit

3.) Process subaddress of the input buffer:

01_{Hex}	Copy error number

Process subaddress of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AV-Bit

4.) Process subaddress of the input buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AF-Bit

An error can also occur after the data have already been sent (see example on the next □).

Function Description
Examples for protocol sequence

Example No. 6

For configuring with double bit header and 8-byte buffer size!

Read 30 bytes starting at code tag address 10, with read error and simultaneous data transmission (code tag type with 64 byte block size)**:**

If an error occurs after data have started to be sent, the AF-Bit is set instead of the AE-Bit along with the corresponding error number. The error message AF is dominant. It cannot be specified which data are incorrect. When the AF-BIT is set the job is interrupted and declared to be ended.

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 01_{Hex}
02_{Hex}	Start address Low Byte $0A_{Hex}$
03_{Hex}	Start address High Byte 00_{Hex}
04_{Hex}	No. of bytes Low Byte $1E_{Hex}$
05_{Hex}	No. of bytes High Byte 00_{Hex}
$00_{Hex}/07_{Hex}$	Set CT-Bit to 1 (64 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

$00_{Hex}/07_{Hex}$	Set AA-Bit
$01...06_{Hex}$	Enter the first 6 data bytes
$00_{Hex}/07_{Hex}$	Invert TO-Bit

3.) Process subaddress of the input buffer:

$01...06_{Hex}$	Copy first 6 data bytes

Process subaddress of the output buffer:

$00_{Hex}/07_{Hex}$	Invert TI-Bit

4.) Process subaddresses of the input buffer:

If an error has occurred:

01_{Hex}	Enter error number
$00_{Hex}/07_{Hex}$	Set AF-Bit

5.) Process subaddress of the input buffer:

01_{Hex}	Copy error number

Process subaddress of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AV-Bit

6.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AF-Bit

Function Description
Examples for protocol sequence

Example No. 7

For configuring with double bit header!

Write 16 bytes starting at code tag address 20 (code tag type with 32 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 02_{Hex}
$02_{Hex}/03_{Hex}$	Start address 14_{Hex} / 00_{Hex}
$04_{Hex}/05_{Hex}$	No. of bytes 10_{Hex} / 00_{Hex}
$00_{Hex}/07_{Hex}$	CT-Bit to 0 (32 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

$00_{Hex}/07_{Hex}$	Set AA-Bit, invert TO-Bit

3.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Enter the first 6 data bytes
$00_{Hex}/07_{Hex}$	Invert TI-Bit

4.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Copy the first 6 data bytes

Process subaddress of the input buffer:

$00_{Hex}/07_{Hex}$	Invert TO-Bit

5.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Enter the second 6 data bytes
$00_{Hex}/07_{Hex}$	Invert TI-Bit

6.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Copy the second 6 data bytes

Process subaddress of the input buffer:

$00_{Hex}/07_{Hex}$	Invert TO-Bit

7.) Process subaddresses of the output buffer:

$01...04_{Hex}$	Enter the remaining 4 data bytes
$00_{Hex}/07_{Hex}$	Invert TI-Bit

8.) Process subaddresses of the output buffer:

$01...04_{Hex}$	Copy the remaining 4 data bytes

Process subaddress of the input buffer:

$00_{Hex}/07_{Hex}$	Set AE-Bit

9.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AV-Bit

10.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 8 with Auto-Read function

For configuring with double bit header!

Programming start address 50 (code tag type with 32 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 07_{Hex}
02_{Hex}	Start address Low Byte 32_{Hex}
03_{Hex}	Start address High Byte 00_{Hex}
$00_{Hex}/07_{Hex}$	CT-Bit to 0 (32 Byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Set AA-Bit and AE-Bit

3.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AV-Bit

4.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 9
Mixed Data Access

For configuring with double bit header!

Storing a program for reading out 3 data records:

1st data record	Start address	5	Number of bytes	7
2nd data record	Start address	75	Number of bytes	3
3rd data record	Start address	312	Number of bytes	17

Total number of bytes exchanged in the operation: 27 bytes

All 104 bytes are written for the programming.

Host:

1.) Process subaddresses of the output buffer in the order shown:

01Hex	Command designator 06Hex
02Hex	Program number 01Hex
00Hex/07Hex	CT-Bit to 0 or 1 (depending on block size), set AV-Bit

Host:

2.) Process subaddresses of the input buffer:

00Hex/07Hex	Set AA-Bit, invert TO-Bit

3.) Process subaddresses of the output buffer:

01Hex 02Hex	1st start address	(Low Byte) 05Hex (High Byte) 00Hex
03Hex 04Hex	1st number of bytes	(Low Byte) 07Hex (High Byte) 00Hex
05Hex 06Hex	2nd start address	(Low Byte) 4BHex (High Byte) 00Hex
00Hex/07Hex	Invert TI-Bit	

4.) Process subaddresses of the input buffer:

00Hex/07Hex	Invert TO-Bit

Continued on next 🗋.

Function Description
Examples for protocol sequence

Example No. 9
Mixed Data Access
(continued)

For configuring with double bit header!

5.) Process subaddresses of the output buffer:

01Hex 02Hex	2nd number of bytes	(Low Byte) 03Hex (High Byte) 00Hex
03Hex 04Hex	3rd start address	(Low Byte) 38Hex (High Byte) 01Hex
05Hex 06Hex	3rd number of bytes	(Low Byte) 11Hex (High Byte) 00Hex
00Hex/07Hex	Invert TI-Bit	

6.) Process subaddresses of the input buffer:

00Hex/07Hex	Invert TO-Bit

7.) Process subaddresses of the output buffer:

01Hex/02Hex	Terminator	FFHex/FFHex
03Hex/04Hex	(not used)	FFHex/FFHex
05Hex/06Hex	(not used)	FFHex/FFHex
00Hex/07Hex	Invert TI-Bit	

8.) Process subaddresses of the input buffer:

00Hex/07Hex	Invert TO-Bit

Fill all unused start addresses and number of bytes with FFHex!

Continued on next 🗋.

Function Description
Examples for protocol sequence

Example No. 9
Mixed Data Access
(continued)

For configuring with double bit header!

35.) Process subaddresses of the output buffer:

$01_{Hex}/02_{Hex}$	(nicht verwendet)	FF_{Hex}/FF_{Hex}
$03_{Hex}/04_{Hex}$	(nicht verwendet)	FF_{Hex}/FF_{Hex}
$05_{Hex}/06_{Hex}$	(nicht verwendet)	FF_{Hex}/FF_{Hex}
$00_{Hex}/07_{Hex}$	TI-Bit invertieren	

36.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	AE-Bit setzen

37.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	AV-Bit rücksetzen

38.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	AA-Bit und AE-Bit rücksetzen

☞ We recommend that you carefully document which parameters are used for start addresses and number of bytes for writing/reading the desired data records.

The data are sequenced in the exact order specified in the program.

Function Description
Examples for protocol sequence

Example No. 10
Mixed Data Access

For configuring with double bit header!

Read code tag using Program No. 1 (code tag type with 32 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 21_{Hex}
02_{Hex}	Program number 01_{Hex}
$00_{Hex}/07_{Hex}$	CT-Bit to 0 (32 byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

$00_{Hex}/07_{Hex}$	Set AA-Bit
$01...06_{Hex}$	Enter first 6 bytes of data
$00_{Hex}/07_{Hex}$	Set AE-Bit

3.) Process subaddresses of the input buffer:

$01...06_{Hex}$	Copy first 6 data bytes

Process subaddress of the output buffer:

$00_{Hex}/07_{Hex}$	Invert TI-Bit

4.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Enter the second 6 data bytes
$00_{Hex}/07_{Hex}$	Invert TO-Bit

... A total of 27 bytes of data are exchanged.
For the remainder of the procedure, see Example 2 on 🗋 37.

☞ Dynamic mode is turned off while the Mixed Data Access program is being run.

Function Description
Examples for protocol sequence

Example No. 11
Mixed Data Access

For configuring with double bit header!

Write code tag using Program No. 1 (code tag type with 32 byte block size)**:**

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 22_{Hex}
02_{Hex}	Program number 01_{Hex}
$00_{Hex}/07_{Hex}$	CT-Bit to 0 (32 byte block size), set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

$00_{Hex}/07_{Hex}$	Set AA-Bit
$01...06_{Hex}$	Enter first 6 bytes of data
$00_{Hex}/07_{Hex}$	Set AE-Bit

3.) Process subaddresses of the input buffer:

$01...06_{Hex}$	Copy first 6 data bytes

Process subaddress of the output buffer:

$00_{Hex}/07_{Hex}$	Invert TI-Bit

4.) Process subaddresses of the input buffer:

$01...06_{Hex}$	Enter the second 6 data bytes
$00_{Hex}/07_{Hex}$	Invert TO-Bit

... A total of 27 bytes of data are exchanged.
For the remainder of the procedure, see Example 7 on 43.

Dynamic mode is turned off while the Mixed Data Access program is being run.

Function Description
Examples for protocol sequence

Example No. 12

For configuring with double bit header!

Put the relevant read/write head into ground state:

Both read/write heads can be independently set to the ground state.

Host:

1.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Set GR-Bit

BIS C-60_1 Identification System:

2.) Go to ground state;
Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Reset BB-Bit

3.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Reset GR-Bit

4.) Process subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Set BB-Bit

Function Description
Examples for protocol sequence

Example No. 13

For configuring with double bit header!

Program configuration data:
Configuration data can be programmed in both buffers - for Head 1 and Head 2 - as desired.

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 04_{Hex}
$00_{Hex}/07_{Hex}$	Set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown.

$00_{Hex}/07_{Hex}$	Set AA-Bit, invert TO-Bit

3.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Enter the 6 configuration bytes
$00_{Hex}/07_{Hex}$	Invert TI-Bit

4.) Process subaddresses of the output buffer:

$01...06_{Hex}$	Enter the 6 configuration bytes
Process subaddresses of the input buffer	
$00_{Hex}/07_{Hex}$	Set AE-Bit

5.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AV-Bit

6.) **Process** subaddresses of the input buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AE-Bit

Function Description
Examples for protocol sequence

Example No. 14

For configuring with double bit header!

Read-out programmed configuration data:

Host:

1.) Process subaddresses of the output buffer in the order shown:

01_{Hex}	Command designator 05_{Hex}
$00_{Hex}/07_{Hex}$	Set AV-Bit

BIS C-60_1 Identification System:

2.) Process subaddresses of the input buffer in the order shown:

00_{Hex}	Set AA-Bit
$01...06_{Hex}$	Enter the 6 configuration bytes
00_{Hex}	Set AE-Bit

3.) **Process** subaddresses of the input buffer::

$01...06_{Hex}$	Copy the 6 configuration bytes
Process subaddress of the output buffer:	
$00_{Hex}/07_{Hex}$	Reset AV-Bit

4.) Process subaddresses of the output buffer:

$00_{Hex}/07_{Hex}$	Reset AA-Bit and AE-Bit

Read/Write Times

Read times from code tag to processor in static mode (parametering: 2nd byte, bit 5 = 0, without CRC-16 data check)

For double read and compare:

Code tag with 32 byte blocks	
No. of bytes	Read time [ms]
from 0 to 31	110
for each additional 32 bytes add	120
from 0 to 255	= 950

Code tag with 64 byte blocks	
No. of bytes	Read time [ms]
from 0 to 63	220
for each additional 64 bytes add	230
from 0 to 2047	= 7350

Write times from processor to code tag in static mode (parametering: 2nd byte, bit 5 = 0, without CRC-16 data check)

Including readback and compare:

Code tag with 32 byte blocks	
No. of bytes	Write time [ms]
from 0 to 31	110 + n * 10
for 32 bytes or more	y * 120 + n * 10

Code tag with 64 byte blocks	
No. of bytes	Write time [ms]
from 0 to 63	220 + n * 10
for 64 bytes or more	y * 230 + n * 10

n = number of contiguous bytes to write
y = number of blocks to be processed

Example: 17 bytes from address 187 have to be written. Code tag with 32 bytes per block. The blocks 5 and 6 will be processed since the start address 187 is in block 5 and the end address 203 in block 6. t = 2 * 120 + 17 * 10 = 410 ms

☞ The indicated times apply after the code tag has been recognized. If the code tag is not yet recognized, an additional 45 ms for building the required energy field until the code tag is recognized must be added.

Read/Write Times

Read times from code tag to processor in dynamic mode (parametering: 2nd byte, bit 5 = 1, without CRC-16 data check)

Read times within the 1st block for dual read and compare:

The indicated times apply after the code tag has been recognized. If the code tag is not yet recognized, an additional 45 ms for building the required energy field until the code tag is recognized must be added.

Code tag with 32 byte blocks	
No. of bytes	Read time [ms]
from 0 to 3	14
for each additional byte add	3.5
from 0 to 31	112

Code tag with 64 byte blocks	
No. of bytes	Read time [ms]
from 0 to 3	14
for each additional byte add	3.5
from 0 to 63	224

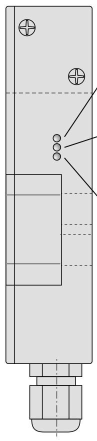
m = highest address to be read

Formula: t = (m + 1) * 3.5 ms

Example: Read 11 bytes starting at address 9, i.e. the highest address to be read is 19. This corresponds to 70 ms.

LED Display

Function displays on BIS C-60_1



The BIS C-60_1 uses the three side-mounted LED's to indicate important conditions of the identification system.

Status	LED	Meaning
Ready / Bus active	red green	Supply voltage OK; no hardware error, however, bus not active. Supply voltage / hardware OK, bus active.
CT1 Present / operating	green yellow yellow flashes off	Code tag read/write-ready at read/write head 1. Read/write command at read/write head 1 in process. Cable break to read/write head or not connected. No code tag in read/write range of read/write head 1.
CT2 Present / operating	green yellow yellow flashes off	Code tag read/write-ready at read/write head 2. Read/write command at read/write head 2 in process. Cable break to read/write head or not connected. No code tag in read/write range of read/write head 2.

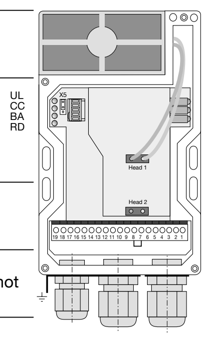
If all three LED's are synchronously flashing, it means a hardware error. Return the unit to the factory.

Function displays

Operating state
INTERBUS

4 diagnostic LEDs on the motherboard inside the processor report the key operating states on the INTERBUS.

LED	State	INTERBUS message
UL	green / off	Reset Protocol chip is / is not supplied with power.
RC / CC Remotebus Check Cable Check	green / off	Communication Ready Communication with the IBS-Master is / is not possible, the application has not however yet started to exchange data.
BA Bus Active	green / off	Communication active or also Run Master is / is not exchanging user data.
RD Remotebus Disable	yellow / off	Remotebus Disable The extended bus interface is / is not turned off.



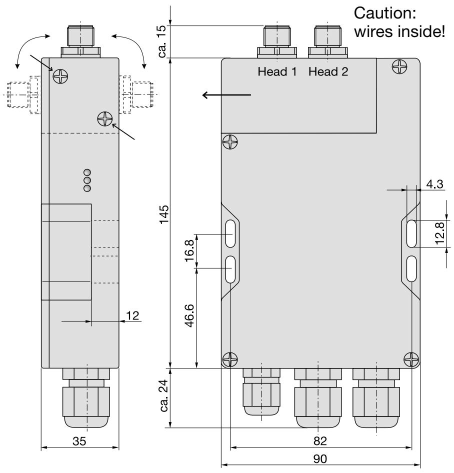
BIS C-6001
Mounting Head / Processor

Mounting the read/write head or adapter

Depending on model, the processor is equipped with a read/write head or the adapter for offset read/write heads. Both the read/write head and the adapter can be rotated by the user by + or -90 deg. to the desired position (see drawing). Be sure that power is off first. Loosen both screws (indicated with arrows). Carefully pull the head or adapter out towards the side (direction of arrow, right drawing).

Caution: wires inside!

Reattach at the desired orientation and screw tight again.



Mounting the BIS C-6001 processor

The processor is attached using 4 lateral mounting holes.

BIS C-6001
Opening the Processor

Opening the Processor BIS C-6001

The BIS C-6001 processor must be opened to perform the following steps:

– Set/change compatibility mode

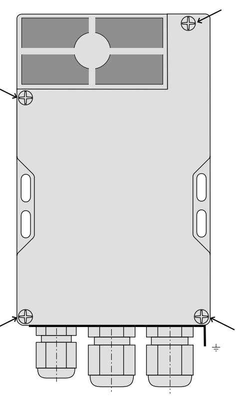
– Replace EEPROM

– Make electrical connections (supply voltage, in-/output, INTERBUS connections).

Be sure that the unit is disconnected from power before opening.

Remove the 4 screws on the BIS C-6001 and lift off the cover.

Perform the desired action. To make the electrical connections, push the cables through the fittings. For additional wiring details, see the following 🕮.



Mounting of the cover (4 screws), max. permissible tightening torque: 0.15 Nm

Opening the processor

BIS C-6001
Installing the connection cables

Make connections on the BIS C-6001 processor

The BIS C-6001 processor must be opened in order to make the connections for the supply voltage, the digital input and the INTERBUS connections (see 🗋 58).

First be sure that the unit is disconnected from power.

Remove the 4 screws on the BIS C-6001 and lift off the cover.

Guide the two INTERBUS cables through the PG 11 fittings (see 🗋 60). For additional information on wiring, see the following 🗋🗋.

Push the cable for supply voltage and for the digital input through the PG 9 fitting.

Close up the processor.

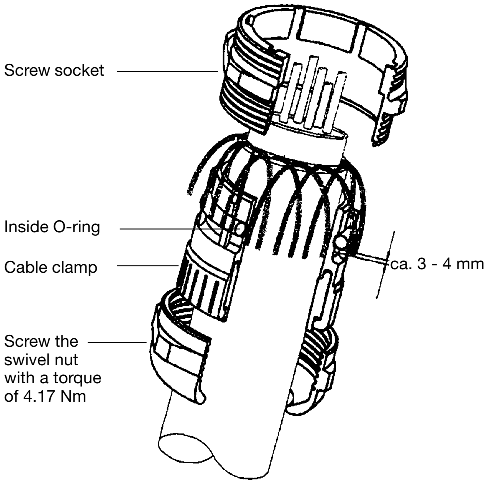
If the processor is equipped with an adapter:

– BIS C-650: Connect the read/write heads to terminals Head 1 and Head 2.
– BIS C-670: Connect the read/write head to terminal Head 1.

BIS C-6001
Mounting the PG Connection

Mounting the PG Connection on the processor BIS C-6001

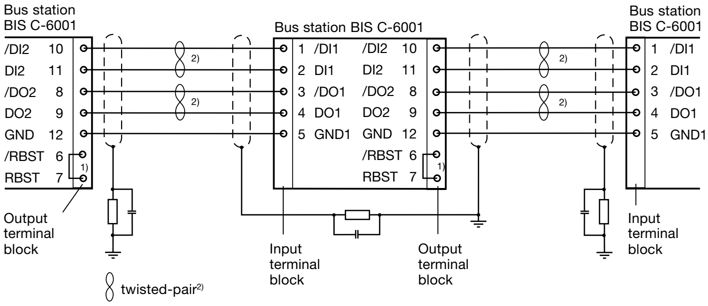
After connecting the (field) bus leads to the termional block, make sure that the shield has proper connection to the PG housing.



BIS C-6001
Interface Information / Wiring Diagrams

Remote bus cable and interfaces for INTERBUS

To insert the BIS C-6001 processors into the serial INTERBUS, the terminal strip provides terminals 1...5 for the input interface and terminals 8...12 for the output interface of the INTERBUS. The following drawing shows the wiring when the BIS C-6001 processors need to be connected together.



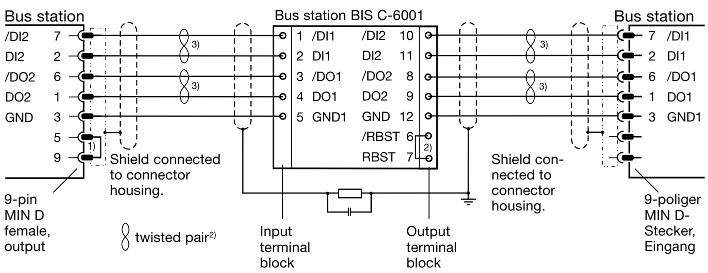
[1]) Leave the jumper in the BIS C-6001 if an additional station is to follow.
Remove it if no additional station follows.

[2]) The differential signals DO and /DO as well as DI and /DI must be twisted-pair.
Recommended cable: LiYCY 3x2x0.25 mm² (AWG 24); maximum cable capacitance: 120 pF/m

BIS C-6001
Interface Information / Wiring Diagrams

Remote bus cable and interfaces for INTERBUS (continued)

To insert the BIS C-6001 processors into the serial INTERBUS, the terminal strip provides terminals 1...5 for the input interface and terminals 8...12 for the output interface of the INTERBUS. The following drawing shows the wiring when the BIS C-6001 interface needs to be connected using a 9-pin terminal (e.g. to a BIS C-6021).



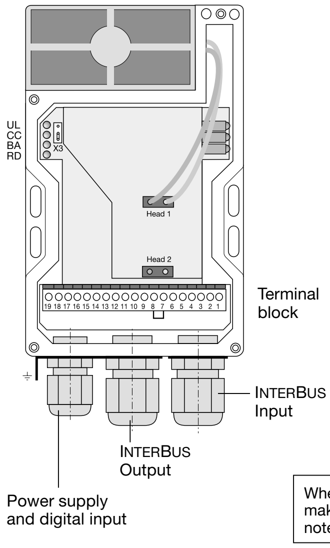
[1]) Connect the jumper in the connector if another station is to follow.
Remove it if no additional station follows.

[2]) The jumper remains in the BIS C-601 if an another station is to follow.
Remove it if no additional station follows.

[3]) The differential signals DO and /DO as well as DI and /DI must be twisted-pair.
Recommended cable: LiYCY 3x2x0.25 mm² (AWG 24); maximum cable capacitance: 120 pF/m

BIS C-6001
Interface Information / Wiring Diagrams

Wiring diagram for BIS C-6001 processor with integrated read/write head



Terminal location and designation

Terminal block assignments

5	4	3	2	1
GND	DO1	/DO1	DI1	/DI1
INTERBUS Input				

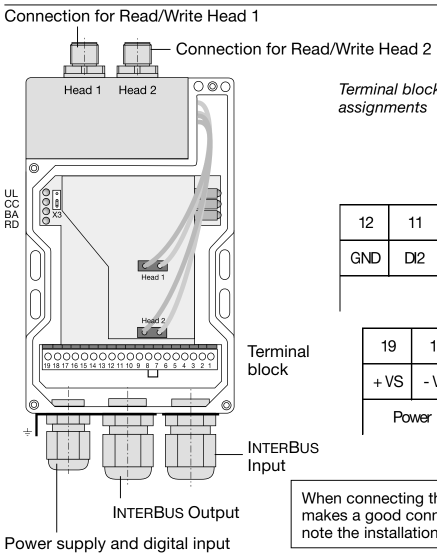
12	11	10	9	8	7	6
GND	DI2	/DI2	DO2	/DO2	RBST	/RBST
INTERBUS Output						

19	18	17	16	15	14	13
+VS	- VS	TxD	RxD	GND	+IN	-IN
Power		RS 232			Digital Input	

When connecting the bus lines, be sure that the shield makes a good connection with the PG housing. Please note the installation instructions on 🗋 60.

BIS C-6001
Interface Information / Wiring Diagrams

Wiring diagram for BIS C-6001 processor with BIS C-650 adapter



Terminal location and designation

Terminal block assignments

5	4	3	2	1
GND	DO1	/DO1	DI1	/DI1
INTERBUS Input				

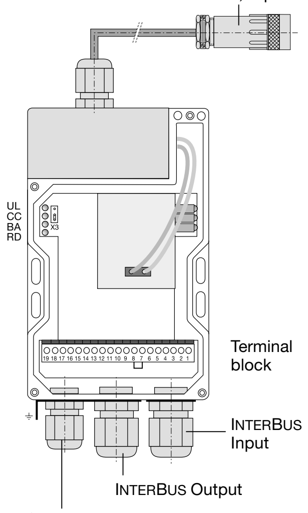
12	11	10	9	8	7	6
GND	DI2	/DI2	DO2	/DO2	RBST	/RBST
INTERBUS Output						

19	18	17	16	15	14	13
+VS	- VS	TxD	RxD	GND	+IN	-IN
Power		RS 232			Digital Input	

When connecting the bus lines, be sure that the shield makes a good connection with the PG housing. Please note the installation instructions on 🗋 60.

BIS C-6001
Interface Information / Wiring Diagrams

Wiring plan for BIS C-6001 using BIS C-670 adapter



Connection location and name

Terminal block assignments

5	4	3	2	1
GND	DO1	/DO1	DI1	/DI1
INTERBUS Input				

12	11	10	9	8	7	6
GND	DI2	/DI2	DO2	/DO2	RBST	/RBST
INTERBUS Output						

19	18	17	16	15	14	13
+ VS	- VS	TxD	RxD	GND	+IN	-IN
Power		RS 232			Digital Input	

When connecting the bus lines, be sure that the shield makes a good connection with the PG housing. Please note the installation instructions on 🗋 60.

BIS C-6001
Changing the EEPROM

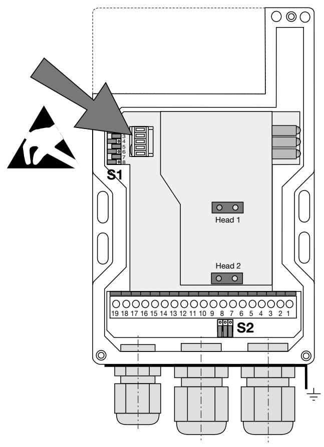
Changing the EEPROM in the BIS C-6001 processor

To replace the EEPROM, open up the processor as described on 🗋 58.

Be sure before opening that the unit is discon-nected from power..

To avoid damaging the EEPROM, please observe the requirements for handling electrostatically sensitive components.

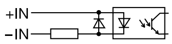
The EEPROM is replaced by unplugging and plugging back into the socket.



Location of the EEPROM

BIS C-6001
Technical Data

Dimensions, Weight	**Housing**	Plastic PS
	Dimensions with read/write head BIS C-652	169 x 90 x 35 mm
	Dimensions with adapter BIS C-650	184 x 90 x 35 mm
	Weight	400 g
Operating Conditions	Ambient Temperature	0 °C to +60 °C
Connections	Terminal Block	19-pin
	Cable Entry for supply voltage	1 x PG 9 fitting (metal)
	for INTERBUS, in-/output	2 x PG 11 fittings (metal)
	Cable Diameter	4 to 8 mm for PG 9
		5 to 10 mm for PG 11
	Wire gauge	0.14 to 1 mm²
	with end crimps	0.25 to 0.34 mm²
Enclosure Rating	Enclosure Rating	IP 65 (when connected)
Electrical Connections	**Supply Voltage V_S, input**	DC 24 V ± 20 %
	Ripple	≤ 10 %
	Current Draw	≤ 400 mA
	INTERBUS, In-/ and Outputs	serial interface for remote bus station, Ident-No. 03, 16 bytes IN, 16 bytes OUT
	Digital Input (+IN, –IN)	Terminal block, Optocoupler isolated
	Control voltage active	4 V to 40 V
	Control voltage inactive	1.5 V to –40 V
	Input current at 24 V	11 mA
	Delay time, typ.	5 ms



BIS C-6001
Technical Data

Electrical Connections (continued)	**Service interface**		RS 232
	Read/Write Head*		integrated, BIS C-65_ and following;
	option for mounted adapter BIS C-650*		2 x connectors 4-pin (male) for all read/write heads BIS C-3_ _ with 4-pin connector (female), not BIS C-350 and BIS C-352
	option for mounted adapter BIS C-670*		1 x connector 4-pin (male) for read/write heads BIS C-350 / BIS C-352
Function Displays	BIS operating states (LED in housing)	LED red / green	Ready / Bus active
		LED green / yellow	CT1 Present / Operating
		LED green / yellow	CT2 Present / Operating
	INTERBUS state (LED on side of housing)	LED green	Reset
		LED green	Cable Check
		LED green	Bus active
		LED yellow	Remotebus Disable

The CE-Mark is your assurance that our products are in conformance with the EC-Guideline

89/336/EEC (EMC-Guideline)

and the EMC Law. Testing in our EMC Laboratory, which is accredited by the DATech for Testing of Electromagnetic Compatibility, has confirmed that Balluff products meet the EMC requirements of the Generic Standard

EN 50081-2 (Emission) and EN 50082-2 (Noise Immunity).

* rotatable by 90 degrees

BIS C-6001
Ordering Information

Ordering Code

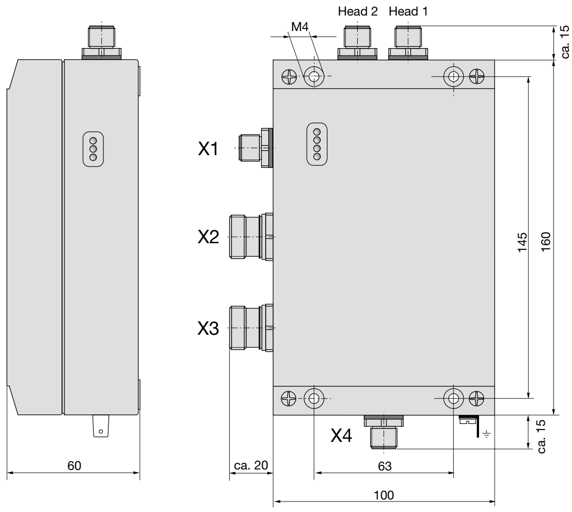
BIS C-6001-023-_ _ _-03-KL2

- Balluff Identification System
- Type C Read/Write System
- Hardware Type
  6001 = INTERBUS (remote bus)
- Software Type
  023 = INTERBUS
- Read/Write Head
  000 = no read/write head
  651 = with read/write head Type 651 (with circular antenna on top)
  652 = with read/write head Type 652 (with circular antenna on front)
  653 = with read/write head Type 653 (with rod antenna)
  650 = with two connections for external read/write heads BIS C-3_ _ (except BIS C-350 and -352)
  670 = with offset connection for an external read/write head BIS C-350 or BIS C-352
- Interface
  BUS versions
- User Connection
  KL2 Clamp connection via 1 x PG 9 and 2 x PG 11

BIS C-6021
Mounting Processor

Mounting the BIS C-6021 processor

The processor is mounted using 4 M4 screws.



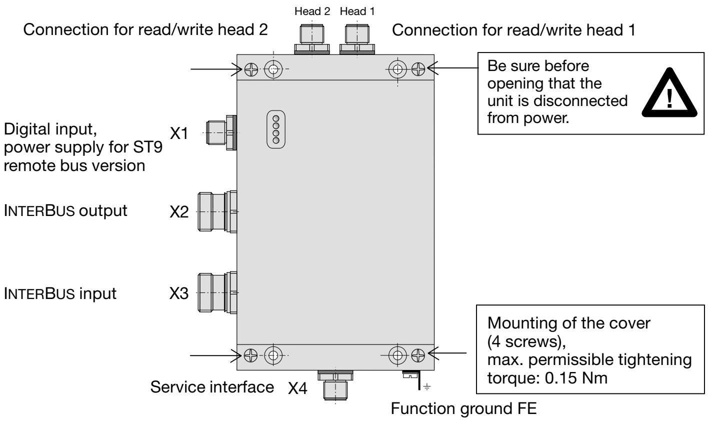
BIS C-6021
Opening the processor / Interface information

Opening the BIS C-6021 processor

To convert the processor for the power supply via the installation remote bus instead of supplying it via X1, the internal connections have to be changed.

Ensure that the device is turned off. Remove the 4 screws on the BIS C-6021 and lift off the cover. Conversion see the following 🕮.

Connection diagram for BIS C-6021 processor



Connection locations and names

BIS C-6021
Interface Information / Wiring Diagrams

To make the connections for the InterBus, the supply voltage and the digital input, connect the pre-assembled cable to the processor. For additional wiring information, see the following 🕮.
Connect the read/write heads to the terminals for Head 1 and Head 2.

Connecting on the remote bus or installation remote bus

The BIS C-6021...ST8 processor is intended for use on the installation remote bus, which provides the supply voltage over the bus. The BIS C-6021...ST9 processor is intended for use on the remote bus, whereby the supply voltage for the processor is brought in through X1.

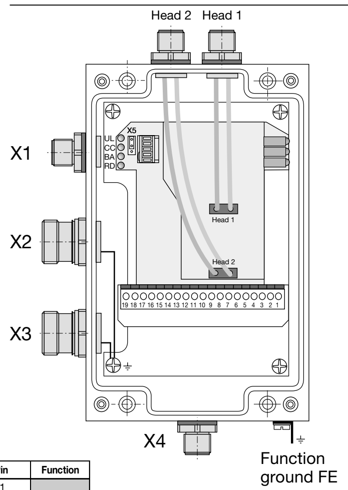
Connect the incoming INTERBUS cable to the INTERBUS input X3. Connect the outgoing INTERBUS cable to the INTERBUS output X2.

If this remote bus station is the last one on the bus, the INTERBUS output X2 must be closed off with a threaded cap to maintain the enclosure rating.

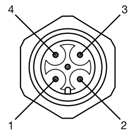
☞ Please note the load capacity of the INTERBUS cable and verify during operation that the supply voltage is maintained at the processor (see Technical Data for specifications).

BIS C-6021...ST8
Interface Information / Wiring Diagrams (installation remote bus)

Wiring diagram for BIS C-6021...ST8 processor on installation remote bus

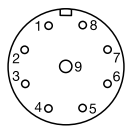


X1, supply voltage and digital input



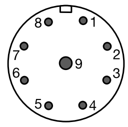
Pin	Function
1	not connected
3	not connected
2	–IN
4	+IN

X2, INTERBUS output



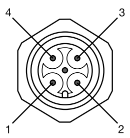
Pin	Function
1	DO2
2	/DO2
3	DI2
4	/DI2
5	GND
9	/RBST
6	FE
7	+24 V
8	0 V

X3, INTERBUS input



Pin	Function
1	DO1
2	/DO1
3	DI1
4	/DI1
5	GND1
6	FE
7	+24 V
8	0 V

X4, Service interface

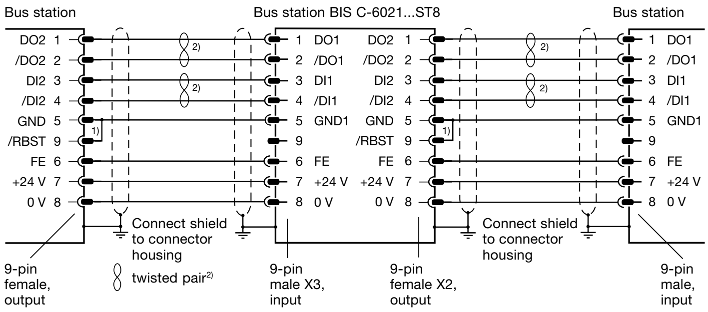


Pin	Function
1	
2	TxD
3	GND
4	RxD

BIS C-6021...ST8
Interface Information / Wiring Diagrams (installation remote bus)

Wiring diagram for BIS C-6021...ST8 processor on installation remote bus (continued)

To integrate BIS C-6021 processor into the serial INTERBUS, two terminals are provided on the housing, X2 as INTERBUS output and X3 as INTERBUS input. For installation remote bus, the stations are supplied through the bus.



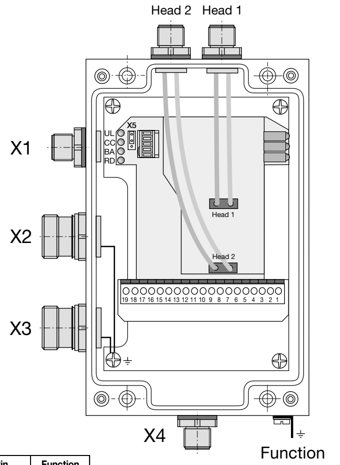
[1]) Connect the jumper in the connector if another station is to follow. Remove it if no additional station follows.

[2]) The differential signals DO and /DO as well as DI and /DI must be twisted-pair.
Recommended cable: LiYCY 3x2x0.25 mm² (AWG 24); maximum cable capacitance: 120 pF/m

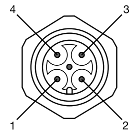
75

BIS C-6021...ST9
Interface Information / Wiring Diagrams (remote bus)

Wiring diagram for BIS C-6021...ST9 processor on remote bus

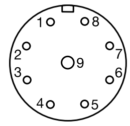


X1, supply voltage and digital input



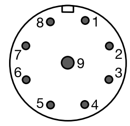
Pin	Function
1	+Vs
3	–Vs
2	–IN
4	+IN

X2, INTERBUS output



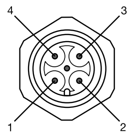
Pin	Function
1	DO2
2	/DO2
3	DI2
4	/DI2
5	GND
9	/RBST
6	FE
7	not connected
8	not connected

X3, INTERBUS input



Pin	Function
1	DO1
2	/DO1
3	DI1
4	/DI1
5	GND1
6	FE
7	not connected
8	not connected

X4, Service interface



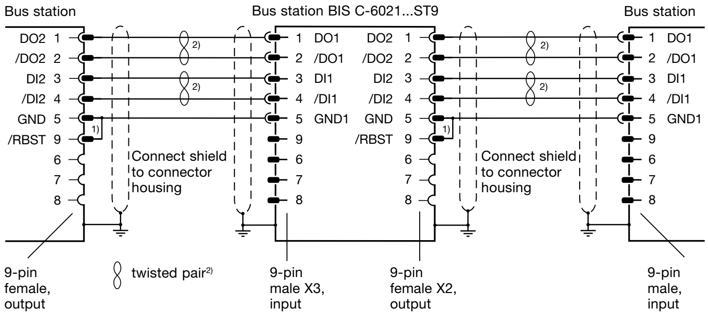
Pin	Function
1	
2	TxD
3	GND
4	RxD

76

BIS C-6021...ST9
Interface Information / Wiring Diagrams (remote bus)

Wiring diagram for BIS C-6021...ST9 processor on remote bus (continued)

To integrate BIS C-6021 processor into the serial INTERBUS, two terminals are provided on the housing, X2 as INTERBUS output and X3 as INTERBUS input. For remote bus, the stations are not supplied through the bus.



[1]) Connect the jumper in the connector if another station is to follow. Remove it if no additional station follows.

[2]) The differential signals DO and /DO as well as DI and /DI must be twisted-pair.
Recommended cable: LiYCY 3x2x0.25 mm² (AWG 24); maximum cable capacitance: 120 pF/m

BIS C-6021
Changing the EEPROM

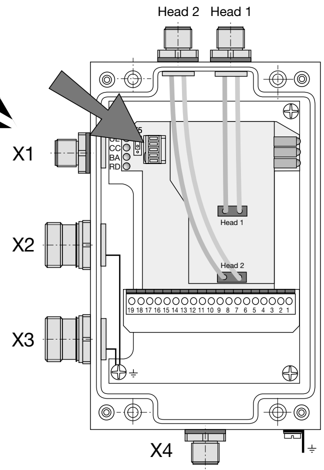
Changing the EEPROM in the BIS C-6021 processor

To change the EEPROM, open the processor as described on 71.

Be sure before opening that the unit is disconnected from power.

To avoid damaging the EEPROM, please observe the requirements for handling electrostatically sensitive components.

The EEPROM is replaced by unplugging and plugging back into the socket.



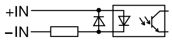
Location of the EEPROM

BIS C-6021
Technical Data

Dimensions, weight	**Housing**	Metal
	Dimensions	190 x 120 x 60 mm
	Weight	820 g
Operating conditions	Ambient temperature	0 °C to +60 °C
Connection type	Integral connector X1	5-pin (male)
	Integral connector Head 1, Head 2	4-pin (male)
	Round connector for X2	9-pin (female)
	Round connector for X3	9-pin (male)
	Integral connector X4	4-pin (male)
Enclosure	Protection class	IP 65 (when connected)
Electrical connections	**Supply voltage V_S**	DC 24 V ± 20 %
	Ripple	≤ 10 %
	Current draw	≤ 400 mA
	Connections for supply voltage V_S	
	with installation remote bus	at INTERBUS input X3, output X2
	with remote bus	at input X1
	Output X2, input X3, INTERBUS	serial interface for remote bus station, Ident-No. 03, 16 bytes IN, 16 bytes OUT (with BIS C-621 mode: 8 Byte IN, 8 Byte OUT)
	Head 1, Head 2, Read/Write Head	via integrated adapter with 2 x connectors for all read/write heads BIS C-3_ _ with 4-pin connector (female), **excluding** BIS C-350 and BIS C-352

BIS C-6021
Technical Data

Electrical Connections (continued)	**Digital input X1 (+IN, –IN)**		galvanically isolated (optocoupler)
	Control voltage active		4 V to 40 V
	Control voltage inactive		1.5 V to –40 V
	Input current at 24 V		11 mA
	Delay time, typ		5 ms
	Service interface X4		RS 232
Function Displays	BIS operating states (LED in housing)	LED red / green	Ready / Bus active
		LED green / yellow	CT1 Present / Operating
		LED green / yellow	CT2 Present / Operating
	INTERBUS state (LED on side of housing)	LED green	Reset
		LED green	Cable Check
		LED green	Bus active
		LED yellow	Remotebus Disable



The CE-Mark is your assurance that our products are in conformance with the EC-Guideline

89/336/EEC (EMC-Guideline)

and the EMC Law. Testing in our EMC Laboratory, which is accredited by the DATech for Testing of Electromagnetic Compatibility, has confirmed that Balluff products meet the EMC requirements of the Generic Standard

EN 50081-2 (Emission) and EN 50082-2 (Noise Immunity).

BIS C-6021
Ordering Information

Ordering code

BIS C-6021-023-050-03-ST_

Balluff Identification System

Type C Read/Write System

Hardware Type
6021 = metal housing, INTERBUS
(remote bus or installation remote bus)

Software Type
023 = INTERBUS

Adapter
050 = with two connections for external read/write heads BIS C-3_ _
(except BIS C-350 and -352)

Interface
03 = BUS versions

User Connection
ST8 = Connector version (installation remote bus)
ST9 = Connector version (remote bus)
(2 round connector for power supply/digital input and service interface, 2 round connectors for INTERBUS)

Accessory (optional, not included)

Type		Ordering code
Mating connector	for X1	BKS-S79-00
	for X2	BKS-S83-00
	for X3	BKS-S84-00
Mating connector	for X4	BKS-S 10-3
Protective cap	for X1, Head 1, Head 2, X4	BES 12-SM-2
Protective cap	for X2	115 475

Appendix, ASCII Table

Deci-mal	Hex	Control Code	ASCII
0	00	Ctrl @	NUL
1	01	Ctrl A	SOH
2	02	Ctrl B	STX
3	03	Ctrl C	ETX
4	04	Ctrl D	EOT
5	05	Ctrl E	ENQ
6	06	Ctrl F	ACK
7	07	Ctrl G	BEL
8	08	Ctrl H	BS
9	09	Ctrl I	HT
10	0A	Ctrl J	LF
11	0B	Ctrl K	VT
12	0C	Ctrl L	FF
13	0D	Ctrl M	CR
14	0E	Ctrl N	SO
15	0F	Ctrl O	SI
16	10	Ctrl P	DLE
17	11	Ctrl Q	DC1
18	12	Ctrl R	DC2
19	13	Ctrl S	DC3
20	14	Ctrl T	DC4
21	15	Ctrl U	NAK

Deci-mal	Hex	Control Code	ASCII
22	16	Ctrl V	SYN
23	17	Ctrl W	ETB
24	18	Ctrl X	CAN
25	19	Ctrl Y	EM
26	1A	Ctrl Z	SUB
27	1B	Ctrl [	ESC
28	1C	Ctrl \	FS
29	1D	Ctrl]	GS
30	1E	Ctrl ^	RS
31	1F	Ctrl _	US
32	20		SP
33	21		!
34	22		"
35	23		#
36	24		$
37	25		%
38	26		&
39	27		'
40	28		(
41	29		)
42	2A		*
43	2B		+

Deci-mal	Hex	ASCII
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@

Deci-mal	Hex	ASCII
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U

Deci-mal	Hex	ASCII
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j

Deci-mal	Hex	ASCII
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C		
125	7D	}
126	7E	~
127	7F	DEL