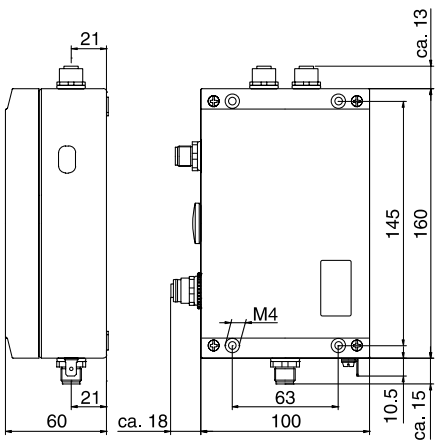
BALLUFF

sensors worldwide

BIS L-6026 EtherNet/IP

Technical Description, User‘s Manual

BIS L



english

www.balluff.com

1 User Notes

1.1 About this Manual

This manual describes the processor for the BIS L-6026 identification system and guides you through startup for immediate operation.

1.2 Manual layout

The manual is designed so that each section builds on the previous sections.
chapter 2: Basic information regarding safety.
chapter 3: The main steps in installing the identification system.
chapter 4: An introduction into the material.
chapter 5: Technical data for the processor.
chapter 6: Mechanical and electrical connections.
chapter 7: Logging the processor on to the network.
chapter 8: User-defined settings for the processor.
chapter 9: How the processor and host system work.

1.3 Conventions

The following conventions are used in this manual.

Enumerations

Enumerations are represented as a list with bullet points.
– Entry 1,
– Entry 2.

Actions

Action instructions are indicated by a preceding triangle. The result of an action is indicated by an arrow.
► Action instruction 1.
⇒ Result of action.
► Action instruction 2.

Notation

Numbers:
– Decimal numbers are represented without additional description (e.g. 123),
– hexadecimal numbers are represented by appending the abbreviation hex (e.g. 00hex).

Parameters:
Parameters are written in italics, e.g. *(CRC_16).*

Directory paths:
Paths in which data are or will be saved/stored are represented in small caps (e.g. Project:\Data Types\Userdefined).

Cross-references

Cross-references indicate where additional information on the topic can be found (see "Technical Data" starting page 12).

1.4 Symbols

Attention!
This symbol indicates a safety advisory which must be observed.

Note, tip
This symbol indicates general notes.

1 User Notes

1.5 Abbreviations

ARP	Address Resolution Protocol
BIS	Balluff Identification System
CIP	Common Industrial Protocol
CRC	Cyclic Redundancy Code
DHCP	Dynamic Host Configuration Protocol
EDS	Electronic Data Sheet
EEPROM	Electrical Erasable and Programmable ROM
EMV	Electromagnetic Compatibility
MAC-ID	Media Access Control Identifier
ODVA	Open DeviceNet Vendor Association
PC	Personal Computer
RPI	Requested Packed Interval
PLC	Programmable Logic Controller

2 Safety

2.1 Intended use

The BIS L-6026 processor is a component of the BIS L identification system. Within the identification system it is used for linking to a host computer (PLC, PC).
It is intended only for use only in this way and in an industrial environment complying with Class A of the EMC Law.
This description applies to processors in series BIS L-6026-034-….

2.2 General notes on device safety

Installation and startup
Installation and startup are to be carried out only by trained specialists. The manufacturer revokes the right to any warranty or liability claims resulting from unauthorized modifications or improper use. When connecting the processor to an external controller, be sure to observe proper polarity for all connections including the power supply (see section "Installation" on page 14). The processor must be operated only using approved power supplies (see section "Technical Data" on page 12).

Operation and testing
It is the responsibility of the operator to ensure that the locally applicable safety regulations are maintained.
In case of defects and faults in the identification system which cannot be remedied, take it out of operation and predect against unauthorized use.

2.3 Meaning of the warning notes

⚠ **Attention!**
The pictogram used with the word "Attention" warns of a possibly hazardous situation for the health of persons or equipment damage.
Disregarding these warnings may result in personal injury or equipment damage.
► Always observe the instructions given for avoiding this hazard.

3 Getting Started

Mechanical connection

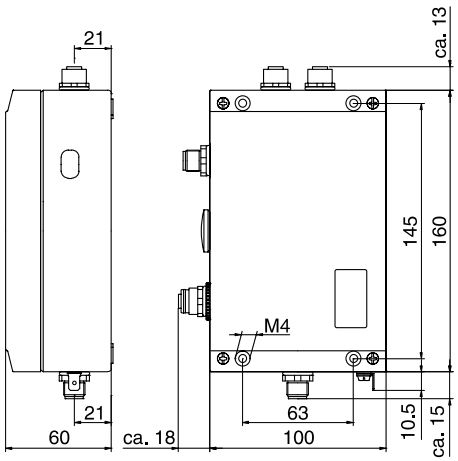


Fig. 1: Mechanical connection

► Attach processor using (4) M4 screws.

Electrical connection

Note
Route the ground wire to ground either directly or through an RC combination, depending on the system.
When connecting to the Ethernet, be sure that the connector shield is perfectly connected to the connector body.
Do not alter the factory setting for the DIL schwitches.

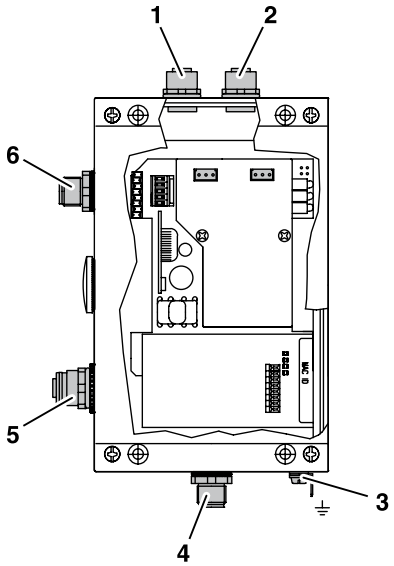
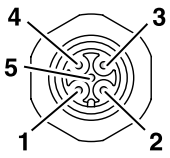


Fig. 2: Electrical connection

1 Head 2 - Read/write head 2
2 Head 1 - Read/write head 1
3 Function ground FE
4 X4 – Service port
5 X3 - Ethernet
6 X1 – Power supply

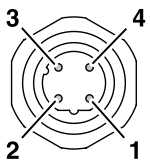
3 Getting Started

X1 - Power supply



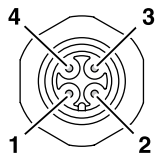
PIN	Function
1	+Vs
2	
3	-Vs
4	
5	

X3 - Ethernet



PIN	Function
1	TD+
2	RD+
3	TD-
4	RD-

X4 - Service port



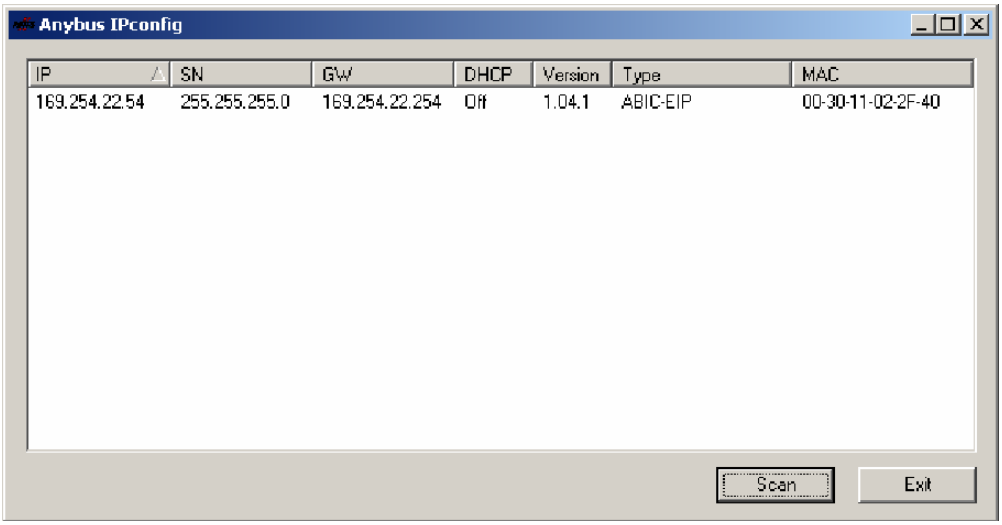
PIN	Function
1	
2	TxD
3	GND
4	RxD

Bus connection

There are two ways of making the bus connection:
- Setting the IP address, subnet mask, gateway address and selecting DHCP using the application "Anybus IPconfig" on a Windows PC having an Ethernet network card.
- Using a DHCP server or BOOTP software.

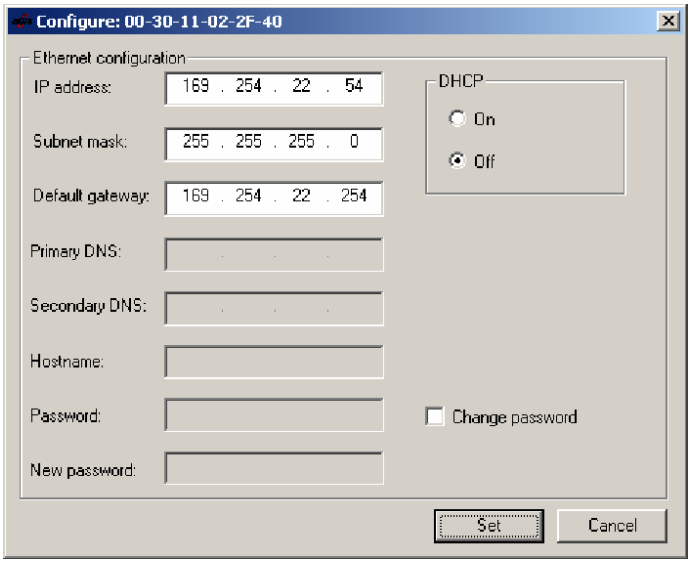
Use the "Anybus IPconfig" program to make the bus connection. The "Anybus IPconfig" application is included on the BIS SD which comes with the processor.

► Start "Anybus IPconfig".
⇒ The subnet is scanned for a connected BIS _-6026. The result of the scan is displayed in the "Anybus IPconfig" window.



► Select the device from the scan list and double-click on it.
⇒ The "Configure" window is opened.

3 Getting Started



- ► Assign the IP address, subnet mask and gateway address.
- ► Turn DHCP on/off.
- ► Confirm your settings by clicking on Set.

4 Basic Knowledge

4.1 Identification system principles of operation

The BIS L identification system belongs to the category of non-contact systems having a read and write function. This enables you to not only read data contained in the data carriers, but also to write new data to them at any point in the process.

The main components of the BIS L identification system are:
– Processor,
– read/write heads,
– data carriers.

The main areas of application are:
– In production for controlling material flow (e.g. for variant-specific processes, workpiece transport using conveying systems, for collecting safety-related data),
– in inventory systems for monitoring inventory movements,
– in transport and conveying technology.

4.2 Product description

BIS L-6026 processor:
– Metal enclosure,
– round connectors for making plug connections,
– capacity for two read/write heads,
– read/write heads are suitable for both dynamic and static operation,
– processor provides power for system components,
– Carrier signal from the read/write heads provides power for the data carrier.

4.3 Control function

The processor represents the link between the data carrier and the host control system.
It manages two-way data transfer between the data carrier and read/write head and provides a buffer storage function. The processor writes data from the host signal to the data carrier through the read/write head, or reads data from the data carrier and makes the data available to the host system.

Host systems may be:
– A control computer (e.g. industrial PC),
– a PLC.

Dual bit-header:
In order to ensure consistency of the transmitted data, the control bits in the first and last byte (bit-header) of the data buffer for each read/write head are sent and compared. If the two bit-headers are identical, then the data were fully updated and may be accepted. This means that the data for each read/write head are only valid if both bit-headers are identical. The host system must therefore also compare the bits in the bit-headers.

4.4 Data integrity

To ensure data integrity the data transfer between data carrier and processor must be monitored using a check procedure. The factory default setting in the processor is for double read with compare. A CRC_16 check may however be selected as an alternative. In CRC_16 checking a check code is written to the data carrier, which enables checking the data for validity at any time. Which procedure is used depends on how you are using the identification system.

Note
Mixed operation of the two check procedures is not possible!

4 Basic Knowledge

The following table provides an overview of the advantages of each respective check procedure.

CRC_16 data check	Double read
Data integrity even during the non-active phase (data carrier outside the read/write head).	No user data bytes are lost for storing a check code.
Shorter read time – page is read just once.	Shorter read time – no check code is written.

4.5 Bus connection

The processor and host system communicate using EtherNet/IP protocol.

EtherNet/IP is an industrial network standard. The IP in EtherNet/IP stands for "Industrial Protocol". EtherNet/IP uses the open communications protocol "Common Industrial Protocol" (CIP) on the application layer (as per the ISO/OSI reference model).
EtherNet/IP is supported by the network organization "Open DeviceNet Vendor Association".

Use of a switch in full-duplex mode is necessary for collision-free data exchange.

5 Technical Data

Dimensions

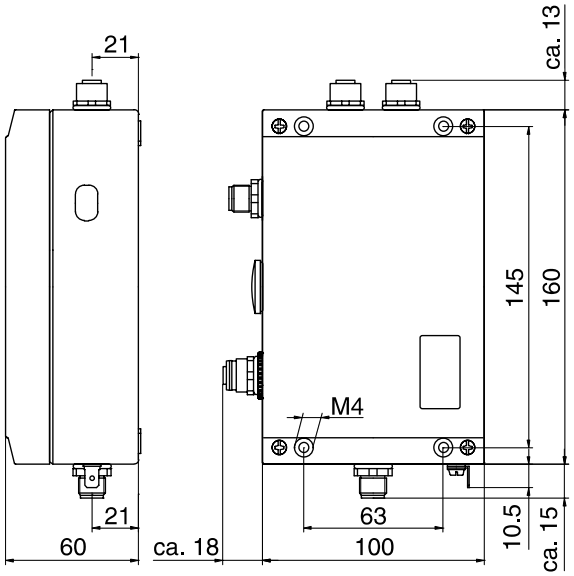


Fig. 3: Dimensions in mm

Mechanical data

Housing material	EN AC-AlSi12 (a), DIN EN 1706
X1 – Input	V_S 24 V DC – 5-pin terminal
X3 – Ethernet	M12 – 4-pin socket, D-coded
X4 – Service port	RS 232 – 4-pin terminal
Head 1, 2 (Write/read head connections)	8-pin socket
Enclosure rating	IP65 (with plugs connected)
Weight	950 g

Electrical data

Operating voltage V_S	24 V DC ±10 %
Ripple	≤ 10 %
Current consumption	≤ 400 mA
Device interface	Ethernet
Service port	RS 232

5 Technical Data

Operating conditions

Ambient temperature	0 °C … 60 °C
EMV – EN 61000-4-2/3/4/5/6 – EN 55011	 – Severity level 4A/3A/4A/1A/3A – Gr. 1, Cl. A
Shock/Vibration	EN 60068 Part 2-6/27/29/64/32

Function indicators

BIS operating states	Ready CT1 Present/Operating CT2 Present/Operating	Green LED Green/yellow LED Green/yellow LED
Status EtherNet/IP	Data Rate (DR) Module Status (MS) Network Status (NS) Link/Activity (L/A)	LED LED LED LED

6 Installation

6.1 Processor installation

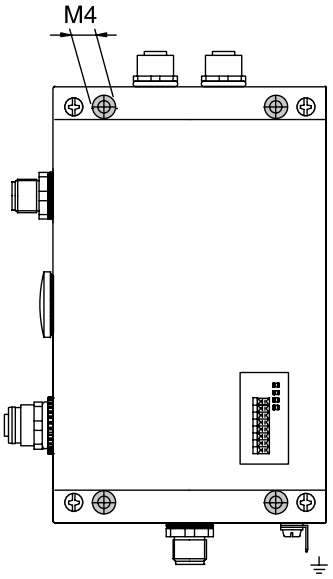


Fig. 4: Installation

► Attach processor using (4) M4 screws.

6.2 Interface information/ Connection diagrams

Note
Route the ground wire to ground either directly or through an RC combination, depending on the system.
When connecting to the Ethernet, be sure that the connector shield is perfectly connected to the connector body.
Do not alter the factory setting for the DIL switches.

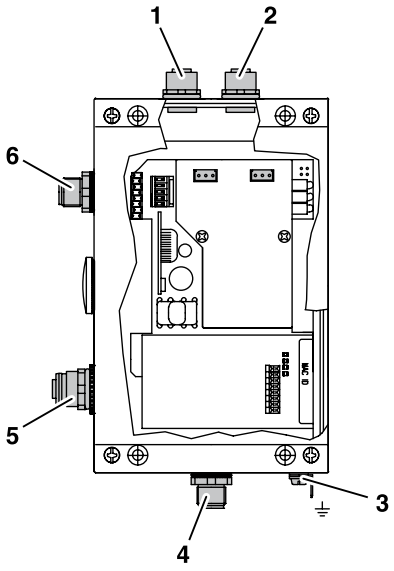
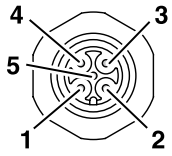


Fig. 5: Processor connections

1 Head 2 - Read/write head 2
2 Head 1 - Read/write head 1
3 Function ground FE
4 X4 - Service port
5 X3 - Ethernet
6 X1 - Power supply

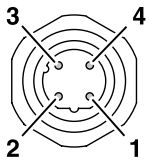
6 Installation

X1 - Power supply



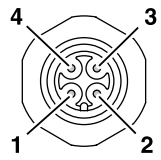
PIN	Function
1	+Vs
2	
3	-Vs
4	
5	

X3 - Ethernet



PIN	Function
1	TD+
2	RD+
3	TD-
4	RD-

X4 - Service port



PIN	Function
1	
2	TxD
3	GND
4	RxD

6.3 Changing the EEPROM

Attention!
Components may be damaged by electrostatic discharge.
► Be sure to turn off power to the device before opening it.

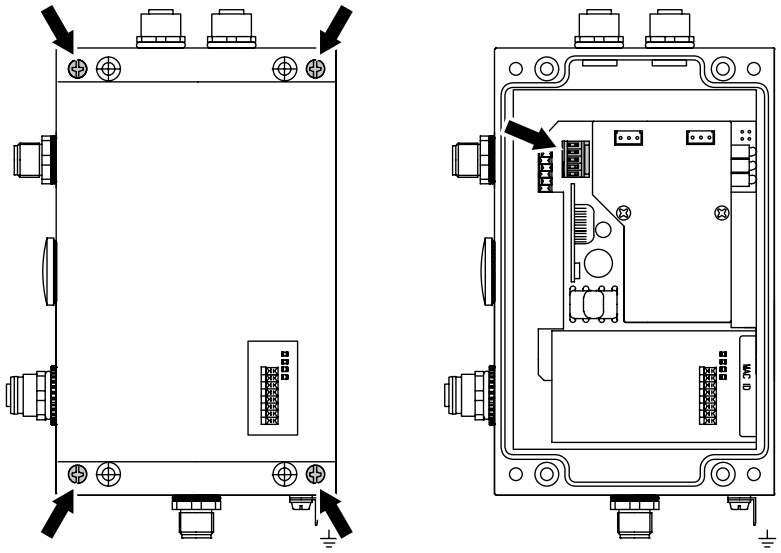


Fig. 6: Changing the EEPROM

► Remove 4 screws from housing cover and remove cover.
► Pull EEPROM from socket.
► Insert new EEPROM into socket.
► Replace cover and tighten 4 screws.

7 Bus Connection

7.1 IP address

The processor and host system communicate using EtherNet/IP protocol. Assigning a unique IP address associates the processor with a network.

The processor can be incorporated into a network in various ways (DHCP, ARP). The MAC address is used as the basis for incorporating into the network. This hardware address is used only one time and uniquely identifies network devices such as the processor.

DHCP

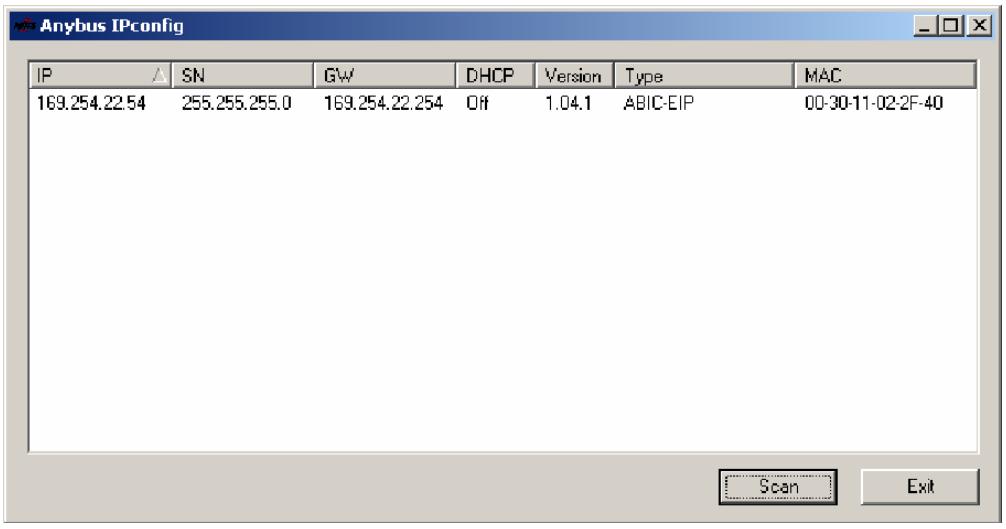
Dynamic Host Configuration Protocol (DHCP) allows a server to be used for dynamic assigning of an IP address. The hardware can be inserted into the network without having to perform any additional configuration. Only automatic obtaining (MAC address) of the IP address needs to be set.

7.2 AnyBus IPconfig

"AnyBus IPconfig" is software which allows the hardware to be addressed for the corresponding subnet prior to installation. In addition, assigning of the IP address through a DHCP server or BOOTP program can be activated (DHCP on) or deactivated (DHCP off).

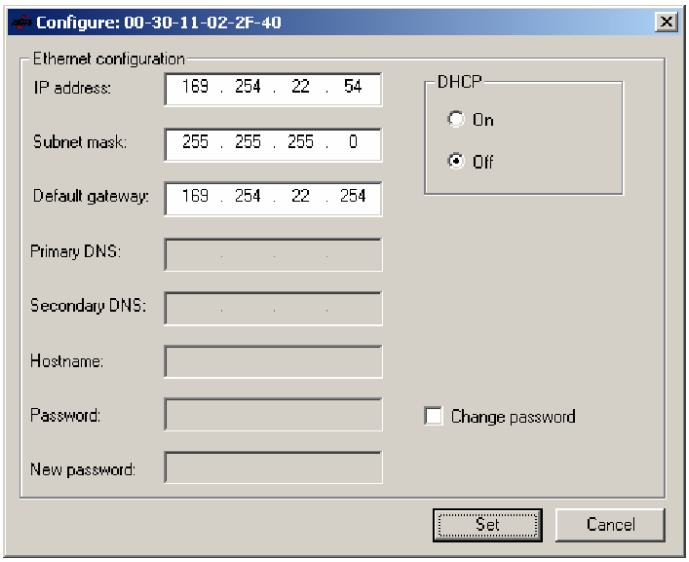
The "Anybus IPconfig" application is included on the BIS SD which comes with the processor.

Start "Anybus IPconfig".
The subnet is scanned for a connected BIS _-6026. The result of the scan is displayed in the "Anybus IPconfig" window.



► Select the device from the scan list and double-click on it.
⇒ The "Configure" window is opened.

7 Bus Connection



- ► Assign the IP address, subnet mask and gateway address.
- ► Turn DHCP on/off.
- ► Confirm your settings by clicking on Set.

8 Parameterizing the Processor

8.1 Basic knowledge

Data carrier types

Two data carrier models are available for the BIS L-6026 processor. Depending on your selection either all or only one particular data carrier can be processed.

Data carrier / Parameter	BIS L-10_-01/L	BIS L-20_-03/L
Memory capacity	192 bytes of user data (read/write) + 4 bytes of fixed serial number (read-only).	5 bytes of fixed serial number (read-only), corresponding to the user data.
CT Present	The first user data are read from the data carrier and loaded into the input buffer.	5 bytes of the serial number are read from the data carrier and loaded into the input buffer.
	If "*Output function type and serial number when CT present*" is configured:	
	Output type 01_{hex} in byte 1 of the input buffer and then the 4 bytes of unique serial number.	Output type 03_{hex} in byte 1 of the input buffer and then the 5 bytes of unique serial number.
Functions	The full command set of the BIS L-6026 processor is available.	No commands from the BIS L-6026 processor are required (all data are output as soon as CT Present is active).
Device parameters	Depends on the number of bytes to be read/written for each read/write head.	*DTTyp* to 'All Tag Types' or 'BIS L-20_' *TypSN* to 'Enable'.

Note
Type BIS L-10_-01/L data carriers are shipped configured with $FF_{hex}37_{hex}$. Only data carriers having this configuration are processed.

The BIS L-10_-01/L carrier contains additional memory ranges for configuration and protected data. These ranges cannot be processed using the BIS L-6026 processor.

CRC check

The CRC check is a procedure for determining a test value for data so as to detect errors in transferring data. If CRC check is activated, an error message is output when a CRC error is detected.

Initializing
To be able to use the CRC check, the data carriers must be initialized. The data carriers are initialized in the output buffer using the command 12_{hex} If the data carrier does not contain the correct CRC, then the processor sets an error message in the input buffer (see Example 10 on page 44).
As shipped from the factory, data carriers may be immediately written a checksum, since all the data are set to 0.

8 Parameterizing the Processor

Error message
- If an error message is the result of a failed write job, then the data carrier must be reinitialized before it can be used again.
- If an error message is not the result of a failed write job, then one or more of the memory cells in the data carrier are defective. This means the data carrier must be replaced.

Checksum
The checksum is written to the data carrier as a 2-byte information. 2 bytes per block are lost. This leaves 14 bytes remaining per block. The usable number of bytes can be determined from the following table.

Data carrier type	Memory capacity	Usable bytes
BIS L-10_-01/L	192 bytes	168 bytes
BIS L-20_-03/L	5 bytes	CRC_16 is not supported

Simultaneous data transmission

Reading
The processor reads the data from the data carrier directly into the input buffer. As soon as the buffer is filled, the Toggle-Bit Out (TO-Bit) is inverted to indicate data ready to the host system. The system inverts the Toggle-Bit In (TI-Bit) to indicate that it is ready to receive, and data read in the meantime are transmitted to the input buffer. This repeats itself until the desired data have been read from the data carrier. After the read procedure is finished, the processor sets the Job End bit (AE-bit) and transmits the remaining data to the input buffer (see example 2 on page 36).

Writing
The processor begins to write data to the data carrier as soon as it has received the first data from the host system. Once all the data have been written to the data carrier, the AE-bit is set.

Dynamic mode

As soon as the Dynamic function is activated, the processor accepts the read/write job from the host system and stores it regardless of whether there is a data carrier in the active zone of the read/write head. When a data carrier enters the active zone of the read/write head, the stored job is carried out

Auto-Read (Standard)

When a data carrier enters the active zone of the read/write head 14 bytes starting at address 00_{hex} are automatically read into the input buffer. No additional read command is required. This allows small data amounts which are stored starting at address 00_{hex} to be read faster.
If a BIS L-20_-03/L data carrier is in front of the read/write head, a maximum of 5 bytes are sent to the input buffer.

If the parameter *TypSN* (Type and serial number when CT Present) is set, then instead of the user data the data carrier type and the unique serial number of the data carrier are sent. For type BIS L-20_-03/L this is always the serial number.

8 Parameterizing the Processor

Auto-Read *Extra*

If Auto-Read *Extra* is activated, then the 14 bytes starting at a specified address are read from the data carrier to the input buffer and then the Codetag Present bit (CP bit) is set. The start address is specified using the parameter *Extra_Adr*.

> **Note**
> For read/write heads hardware version V2.0 and higher the number of bytes to read can be set from 4 to 14 bytes using the parameter *CP_Number*.

Type and serial number

If this function is activated, with CT Present the data carrier type and serial number (UID = unique ID) of the data carrier are issued. For the data carrier type BIS M-1_ _-02/L this is a useful application in order also to record the UID.

8.2 Parameterizing

There are two different ways to set parameters. Parameterizing from an application program or using the EDS file.

Basics

The parameters for operating the processor are stored in the BIS Sonfig Object (class 64hex). Explicit messages are used to access the parameters.

Parameterizing from an application program
One widely used application for EtherNet/IP device parameterizing is the Windows program RSLogix 5000 written for the Logix 5000 controller of Rockwell Automation. An example for device programming is included on the BIS-CD. For additional information see section "Example for parameterizing with application program" on page 21.

EDS file
The EDS file contains all the device parameters for the processor. The file is included on the BIS-CD.

Parameters

CRC_16	class:	64_{hex}
	instance:	01_{hex}
	attribute:	01_{hex}
Factory setting:	Disable	(= 0)
Data validity is checked using double read.		
Other settings:	Enable	(= 1)
Data validity is ensured using CRC check.		

Simultaneous	class:	64_{hex}
	instance:	01_{hex}
	attribute:	02_{hex}
Factory setting:	Disable	(= 0)
Read/write jobs and data transmission are run in sequence.		
Other settings:	Enable	(= 1)
Read/write jobs and data transmission are run simultaneously.		

8 Parameterizing the Processor

Dynamic1 — class: 64hex; instance: 01hex; attribute: 03hex

Factory setting: Disable (= 0)

Read/write head 1 is in static mode. Read/write command from the controller is carried out only if there is a data carrier in the active zone of Read/Write Head 1.

Other settings: Enable (= 1)

Read/Write Head 1 is in dynamic mode.

Dynamic2 — class: 64hex; instance: 01hex; attribute: 04hex

Factory setting: Disable (= 0)

Read/write head 2 is in static mode. Read/write command from the controller is carried out only if there is a data carrier in the active zone of Read/Write Head 2.

Other settings: Enable (= 1)

Read/Write Head 2 is in dynamic mode.

Extra1 — class: 64hex; instance: 01hex; attribute: 05hex

Factory setting: Disable (= 0)

CT Present data if there is a data carrier in the active zone of Read/Write Head 1.

Other settings: Enable (= 1)

The Auto-Read function is active.

Extra2 — class: 64hex; instance: 01hex; attribute: 06hex

Factory setting: Disable (= 0)

CT Present data if there is a data carrier in the active zone of Read/Write Head 2.

Other settings: Enable (= 1)

The Auto-Read function is active.

Extra_Adr1 — class: 64hex; instance: 01hex; attribute: 07hex

Factory setting: 0

Other settings: 1…191

Specifies the start address (Auto-Read) beginning at which the data carrier is read when a data carrier enters the active zone of Read/Write Head 1.

8 Parameterizing the Processor

Extra_Adr2

class: 64hex
instance: 01hex
attribute: 08hex

Factory setting: 0

Other settings: 1…191

Specifies the start address (Auto-Read) beginning at which the data carrier is read when a data carrier enters the active zone of Read/Write Head 2.

TypSN

class: 64hex
instance: 01hex
attribute: 09hex

Factory setting: Disable (= 0)

At CT Present the data carrier type and the serial number of the data carrier are output.

Other settings: Enable (= 1)

DTTyp

class: 64hex
instance: 01hex
attribute: 0Ahex

Factory setting: All data carrier types (= 0)

Depending on your selection either all or only one particular data carrier can be processed.

Other settings: BIS L-10_-01/L (= 1)

Not used (= 2)

BIS L-20_-03/L (= 3)

CP_Number1

class: 64hex
instance: 01hex
attribute: 0Bhex

Factory setting: 4

Other settings: 1…14

Specify the number of bytes to be read when a data carrier enters the active zone of read/write head 1.
Only active for hardware version V2.0 and higher of the read/write head. For hardware versions <V2.0 14 bytes are always read.

CP_Number2

class: 64hex
instance: 01hex
attribute: 0Chex

Factory setting: 4

Other settings: 1…14

Specify the number of bytes to be read when a data carrier enters the active zone of read/write head 2.
Only active for hardware version V2.0 and higher of the read/write head. For hardware versions <V2.0 14 bytes are always read.

8 Parameterizing the Processor

Example of parameterizing using the application program

This example shows how the example project included on the BIS-CD can be used with the RSLogix 5000 software for a user project.

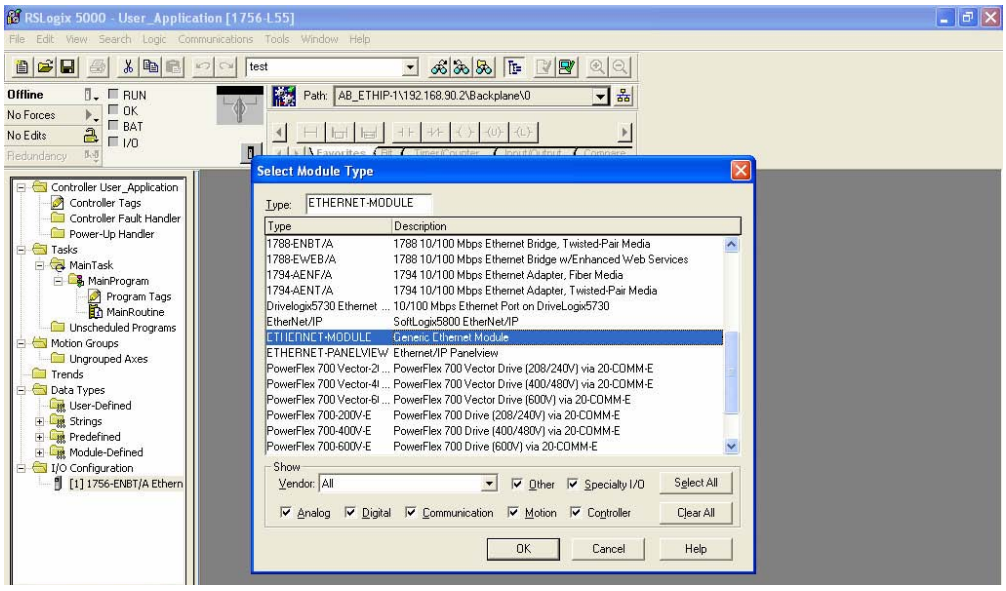
Note the following procedure:
1. Add the BIS L-6026 to a user project.
2. Import the example project into a new project.
3. Copy user-defined data type from the example project to the user project.
4. Create a sub-routine in the user project.
5. Set invoking of the sub-routine in the main program of the user program.

To run the example the files stored on the BIS-CD must by copied to a local directory.

Note
Information about the software, installation, creating projects and working in projects can be found in the manual for the RSLogix 5000 manual.

1. Add the processor

- ► Open user project.
- ► Under I/O-CONFIGURATION\[1] 1756-ENBT/A ETHERNETIP create a new module. (type: Generic Ethernet Module).



8 Parameterizing the Processor

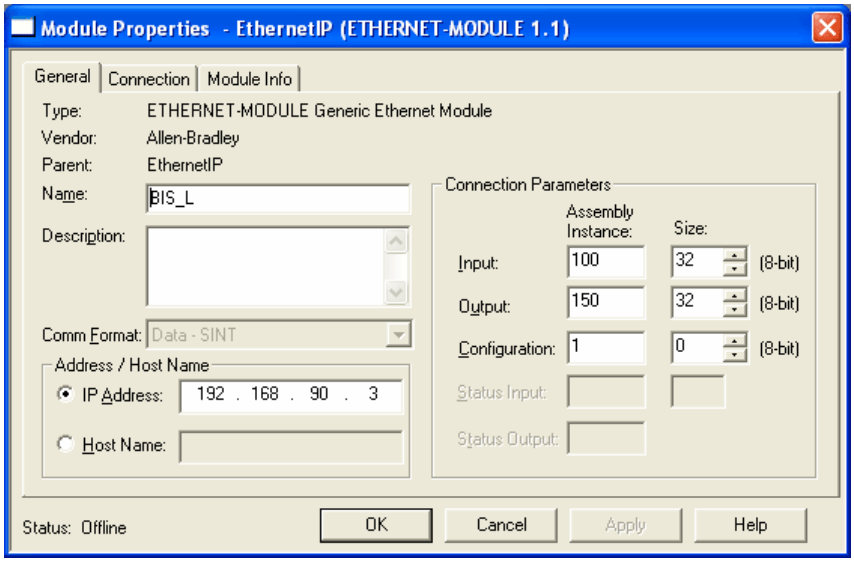
► Set module properties:

Name:	e.g.	BIS_L
Communication format:		Data SINT
IP address:	e.g.	192.168.90.3

► Set the connection parameters as follows:

		Instance	Size
Connection parameters	Input:	100	32 bytes
	Output:	150	32 bytse
	Configuration:	1	0

Note
"Configuration" is not supported. Therefore the values are set to 1 and 0.



► Save settings by clicking on "OK" and confirm the remaining dialog fields until the module has been successfully created.

Note
When confirming the dialog fields, be sure that the Requested Packed Interval (RPI) ≥ 10 ms is set.

2. Import example project

Note
Only one project per window can be opened in RSLogix 5000.

► Open a new project.
► Import the example project "Example_Project_Param_BIS_L.L5K" from the local directory to the project (File\Open).
► Save example project in *.ACD format (File\Save as) – File name is freely selectable.

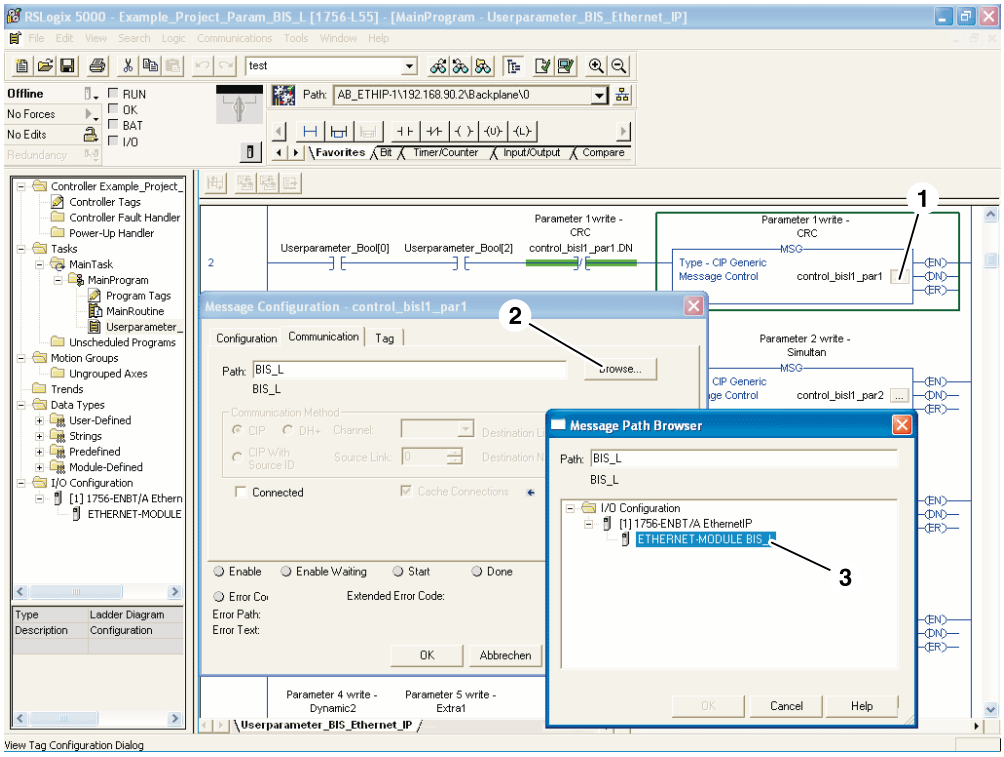
8 Parameterizing the Processor

3. Copy user-defined data type

- Under Data Types\User defined in the example project copy "BIS_L_Userparameters".
- Under Data Types\User defined in the example project paste "BIS_L_Userparameters".

4. Create a sub-routine in the user project

- Under Tasks\MainTask\MainProgram in the user project create a new routing with the name "Userparameters_BIS_Ethernet_IP".
- Double-click to open the new routine.
- Right-click on "Import flow path" from the context menu.
- Import the file "Example_Project_Rung_BIS_L.L5X" from the local directory to the user project.
- Reconfigure the communication paths for all messages – see screenshot for sequence.



5. Set invoking of the sub-routine

- Under Tasks\MainTask\MainProgram select MainRoutine.
- In the MainRoutine set "Userparameter_Bool (0)" to high.
  ⇒ Sub-routine is activated.

9 Device Function

9.1 Function principle BIS L-6026

Two buffers are required to exchange data and commands between the processor and the host system. Cyclical polling is used for exchanging the buffer contents. The buffer content depends on the cycle in which it is written (e.g. control commands at the start of the job).
When writing the buffer, the transmitted data from the previous cycle are overwritten. Unwritten bytes are not deleted and retain their data content.

The buffer size of the overall buffer is 32 bytes. 16 bytes are available for each read/write head.

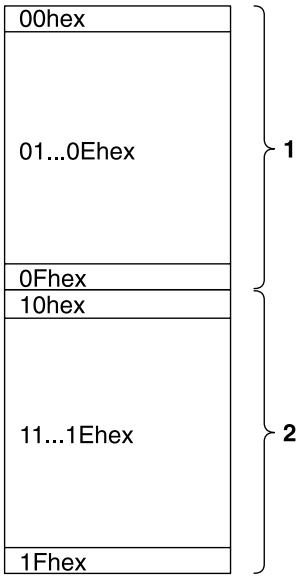


Fig. 7: Total buffer Read/Write Heads 1 and 2

1 Read/Write Head 1 **2** Read/Write Head 2

Only 14 bytes per read/write head are available for data exchange, since the first and last bytes in the respective data buffer are used for control and for status messages.

Output buffer

The output buffer is used to transmit the identification system commands and the data to be written to the data carrier.

Bit no. / Subaddress	7	6	5	4	3	2	1	0
00hex = Bit header		TI	KA			GR		AV
01hex	Command					or	Data	
02hex	Stared address (Low Byte) or program no.					or	Data	
03hex	Start address (High Byte)					or	Data	
04hex	No. of bytes (Low Byte)					or	Data	
05hex	No. of bytes (High Byte)					or	Data	
06hex	Data							
…	Data							
0Fhex = Bit header		TI	KA			GR		AV

9 Device Function

Allocation and explanation

Subaddress	Bit name	Meaning	Function description
00hex/0Fhex	TI	Toggle-Bit In	Controller is ready to receive additional data (read job).
	KA	Head deselect	Turns the read/write head off.
	GR	Base state	Identification goes into base state for the respective read/write head. Any pending job is cancelled.
	AV	Job	A job is pending for the respective read/write head.

Subaddress	Meaning	Function description
01hex	Command 00hex 01hex 02hex 06hex 12hex 21hex 22hex	 No command. Read data carrier. Write to data carrier. Save program for "Mixed data access" in EEPROM. Initialize CRC_16 data check. Read data carrier as per a program for "Mixed data access". Write to data carrier as per a program for "Mixed data access".
	or Data	Transmitting data written to the data carrier.
	or Program data	Transmitting program data written to the EEPROM.

02hex	Start address (Low Byte)	Address starting at which reading or writing should commence (address range from 0 to 191 is covered).
	or Program No.	Program No. to be stored for "Mixed Data Access" in conjunction with command 06hex (value range 01hex to 0Ahex).
	or Program No.	Program No. to be run for "Mixed Data Access" in conjunction with command 21hex or 22hex (value range 01hex to 0Ahex).
	or Data	Transmitting data written to the data carrier.
	or Program data	Transmitting program data written to the EEPROM.

03hex	Start address (High Byte)	Address starting at which reading or writing should commence (at current data carrier capacity always 0).
	or Data	Transmitting data written to the data carrier.
	or Program data	Transmitting program data written to the EEPROM.

9 Device Function

Subaddress	Meaning	Function description
04hex	No. of bytes (Low Byte)	No. of bytes (1 to 192 bytes) to read or write beginning at the start address (Low Byte).
	or Data	Transmit the data which are written to the data carrier.
	or Program data	Transmitting program data written to the EEPROM.
05hex	No. of bytes (High Byte)	No. of bytes to read or write commencing with the start address (at current data carrier capacity always 0).
	or Data	Transmitting data written to the data carrier.
	or Program data	Transmitting program data written to the EEPROM.
06hex	Data	Transmitting the data written to the data carrier.
	or Program data	Transmit the program data which are written to the EEPROM.
...	Data	Transmitting the data written to the data carrier.
	or Program data	Transmit the program data which are written to the EEPROM.

Input buffer

The input buffer is used to transmit the data read by the identification system, the identifiers and error codes to the host system.

Bit no. / Subaddress	7	6	5	4	3	2	1	0
00hex = Bit header	BB	HF	TO		AF	AE	AA	CP
01hex	Error code or Data							
02hex	Data							
…	Data							
0Fhex = Bit header	BB	HF	TO		AF	AE	AA	CP

9 Device Function

Allocation and explanation

Subaddress	Bit name	Meaning	Function description
00hex/0Fhex	BB	Ready	Identification system is ready.
	HF	Head error	Cable break on read/write head or no read/write head connected.
	TO	Toggle-Bit Out	**Read procedure:** Identification system has additional data ready. **Write procedure:** Identification system can accept additional data.
	AF	Job error	Error in processing the job, or job cancelled.
	AE	Job end	Confirmation – Job ended without error.
	AA	Job start	Confirmation – Job was recognized and started.
	CP	Codetag Present	There is a data carrier in the active zone of the read/write head.

Subaddress	Meaning	Function description
01hex	Error code	Error number valid only with AF-bit!
	01hex	Job cannot be carried out becoffe there is no data carrier in the active zone of the read/write head.
	02hex	Read error.
	03hex	Data carrier was removed from the active zone of the read/write head during reading.
	04hex	Write error.
	05hex	Data carrier was removed from the active zone of the read/write head during writing.
	06hex	Memory access error.
	07hex	Invalid or no command for set AV-bit or number of bytes is 00hex.
	09hex	Cable break on read/write head or no read/write head connected.
	0Chex	EEPROM cannot be read or programmed.
	0Dhex	Communication fault with data carrier.
	0Ehex	CRC for read data and CRC for data carrier do not agree.
	0Fhex	1st and 2nd bit header are not identical. The 2nd bit header must be operated.
	20hex	Addressing of the read/write job is outside the memory range of the data carrier.
	21hex	This function is not possible for this data carrier.
	or Data	Transmit data which were read from the data carrier.
02hex	Data	Transmit data which were read from the data carrier.
...	Data	Transmit data which were read from the data carrier.

9 Device Function

Communication

Communication between the host system and the processor is defined by a sequence protocol. A control bit in the output and input buffer is used to implement communication between the host system and the processor.

Basic sequence

1. Controller sends command in output buffer to processor with set AV bit. The AV bit tells the processor that a job is beginning and the transmitted data are valid.
2. Processor takes the job and confirms the job by setting the AA bit in the input buffer.
3. If additional data need to be exchanged for the job, readiness for additional exchange is indicated by inverting the Toggle-Bits TI and TO.
4. The processor has correctly executed the job and set the AE bit in the input buffer.
5. The controller has received all data. The AV bit in the output buffer is reset.
6. The processor resets all the control bits (AA, AE) in the input buffer which were set during the job. The processor is ready for the next job.

Mixed data access

By carrying out read/write programs it is possible to write data to various address ranges on the data carrier or read data which are contained in various address ranges on the data carrier. This function is referred to as "Mixed data access". The read/write programs are stored in the processor's EEPROM. 10 programs with up to 25 instructions can be stored. Each program instruction contains the information about the start address and number of bytes. The maximum allowable amount of data that can be transmitted is 2 kB.

Saving programs:

Command 06_{hex} in the output buffer sends the program to the processor. Saving a program is considered a job. All 25 instructions and two additional bytes with $FF_{hex}FF_{hex}$ as an end delimiter must always be transmitted. This means that 104 bytes per program, including the command and program number, are transmitted (see example 7 on page 41).

9 Device Function

Program structure example:

Program structure	Subaddress	Value	Value range
Command	01_{hex}	06_{hex}	
1. Program set			
Program number	02_{hex}	01_{hex}	01_{hex} to $0A_{hex}$
1. 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}		
2. Data record			
…			
25. 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}		
End delimiter	$FF_{hex}FF_{hex}$		

Running programs:
The programs stores in the EEPROM may be used for reading data records from data carriers as well as for writing data records to a data carrier. The command 21_{hex} (read) or 22_{hex} (write) in the output buffer is used to specify reading or writing (see example 8 on page 42 and example 9 on page 43).

9 Device Function

Read/write times

Note
All specifications are typical values. Deviations are possible depending on the application and combination of read/write head and data carrier.
The specifications apply to static operation, no CRC_16 data checking.

Read times:

Data carrier BIS L-1_ _ with 4-byte blocks	
Data carrier recognition	~ 370 ms
Read bytes 0 to 3	~ 180 ms
for each additional start of 4 bytes	+ ~ 90 ms

Data carrier BIS L-2_ _	
Data carrier recognition + Read data carrier	~ 270 ms

Write times:

Data carrier BIS L-1_ _ with 4-byte blocks	
Data carrier recognition	~ 370 ms
Write bytes 0 to 3	~ 305 ms
for each additional start of 4 bytes	+ ~ 215 ms

Data carrier BIS L-2_ _
Writing not possible

9 Device Function

9.2 Function indicators

The operating states of the identification system, the Ethernet connection and the EtherNet/IP connection are indicated by means of LED's.

Overview of indicators

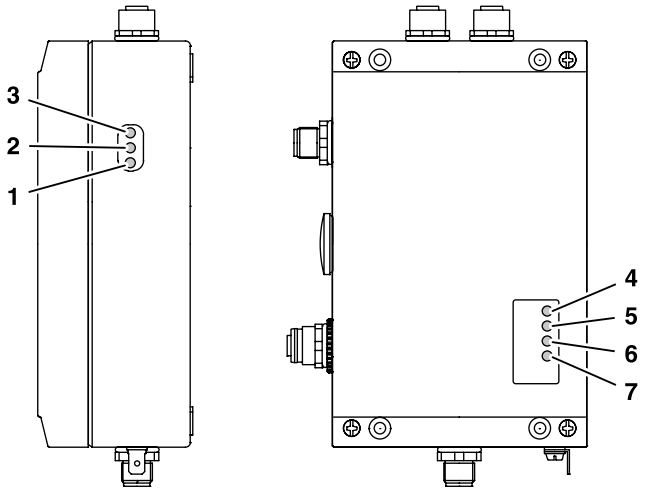


Fig. 8: Function indicators

Identification system

1 CT2 Present/Operating
2 CT1 Present/Operating
3 Ready

Ethernet and EtherNet/IP

4 Data Rate (DR)
5 Module Status (MS)
6 Network Status (NS)
7 Link/Activity (L/A)

Power-up

During power-up all LED's for the Ethernet and EtherNet/IP connection are tested as described in the following table.

LED name	LED sequence									
Data Rate (DR)	off								green	red
Module Status (MS)	green	red	green							
Network Status (NS)	off					green	red	off		
Link/Activity (L/A)	off							green	red	off

Diagnostics

Identification system

Status LED	Meaning
Ready	
green	Operating voltage present; no hardware error

CT1 Present/Operating	
green	Data carrier ready to read/write at Read/Write Head 1
yellow	Read/write job being processed at Read/Write Head 1
yellow flashing	Cable break on Read/Write Head 1 or Read/Write Head 1 not connected.
off	No data carrier in the active zone of the Read/Write Head 1.

9 Device Function

Status LED	Meaning
CT2 Present/Operating	
green	Data carrier ready to read/write at Read/Write Head 2
yellow	Read/write job being processed at Read/Write Head 2
yellow flashing	Cable break on Read/Write Head 2 or Read/Write Head 2 not connected.
off	No data carrier in the active zone of the Read/Write Head 2

Ethernet- and EtherNet/IP connection

Status LED	Meaning
Data Rate	
off	Transmission rate 10 Mbit
green	Transmission rate 100 Mbit
red	-

Module Status	
off	No power to module
green	Device ready
green flashing	Module configuration missing or incorrect
red	Non-clearable error
red flashing	Clearable error

Network Status	
off	No voltage or no IP address
green	Device has at least one EtherNet/IP connection
green flashing	Device has no EtherNet/IP connection
red	An IP address is duplicated
red flashing	One or more EtherNet/IP connections has timed out

Link/Activity	
off	No power
green	Device is connected to Ethernet
green flashing	RX/TX activity
red	-

9 Device Function

9.3 Examples

1. Read 30 bytes on Head 1, start address 10

Controller

1. Process output buffer (note sequence):

01hex	Command 01hex
02hex	Start address 0Ahex
03hex	Start address 00hex
04hex	No. of bytes 1Ehex
05hex	No. of bytes 00hex
00hex /0Fhex	Set AV bit

3. Process input buffer:

01…0Ehex	Copy first 14 bytes

Process output buffer:

00hex /0Fhex	Invert TI-Bit

5. Process input buffer:

01…0Ehex	Copy second 14 bytes

Process output buffer:

00hex /0Fhex	Invert TI-Bit

7. Process input buffer:

01…02hex	Copy last bytes

Process output buffer:

00hex /0Fhex	Reset AV-Bit

Identification system

2. Process input buffer (note sequence):

00hex /0Fhex	Set AA bit
01…0Ehex	Enter first 14 bytes
00hex /0Fhex	Set AE-Bit

4. Process input buffer:

01…0Ehex	Enter second 14 bytes
00hex /0Fhex	Invert TO-Bit

6. Process input buffer:

01…02hex	Enter last bytes
00hex /0Fhex	Invert TO-Bit

8. Process input buffer:

00hex /0Fhex	Reset AA and AE-Bit

9 Device Function

2. Read 30 bytes on Head 1, start address 10, simultaneous data transmission

While the read job is being carried out and as soon as sufficient data have been read for filling the input buffer of Read/Write Head 1, these data are sent to the input buffer. The AE bit is not set until the "Read" operation has been finished by the processor.

The reply "Job End" (AE-Bit) is reliably set no later than when the last data are sent. The actual time depends on the requested amount of data and the time response of the controller. In the example the italicized "*Set AE-Bit*" draws your attention to this fact.

Controller

1. Process output buffer (note sequence):

01_{hex}	Command 01_{hex}
02_{hex}	Start address $0A_{hex}$
03_{hex}	Start address 00_{hex}
04_{hex}	No. of bytes $1E_{hex}$
05_{hex}	No. of bytes 00_{hex}
00_{hex} /$0F_{hex}$	Set AV bit

3. Process input buffer:

$01…0E_{hex}$	Copy first 14 bytes

Process output buffer:

00_{hex} /$0F_{hex}$	Invert TI-Bit

5. Process input buffer:

$01…0E_{hex}$	Copy second 14 bytes

Process output buffer:

00_{hex} /$0F_{hex}$	Invert TI-Bit

7. Process input buffer:

$01…02_{hex}$	Copy last bytes

Process output buffer:

00_{hex} /$0F_{hex}$	Reset AV-Bit

Identification system

2. Process input buffer (note sequence):

00_{hex} /$0F_{hex}$	Set AA bit
$01…0E_{hex}$	Enter first 14 bytes
00_{hex} /$0F_{hex}$	Invert TO-Bit
00_{hex} /$0F_{hex}$	*Set AE-Bit*

4. Process input buffer:

$01…0E_{hex}$	Enter second 14 bytes
00_{hex} /$0F_{hex}$	Invert TO-Bit
00_{hex} /$0F_{hex}$	*Set AE-Bit*

6. Process input buffer:

$01…02_{hex}$	Enter last bytes
00_{hex} /$0F_{hex}$	Invert TO-Bit
00_{hex} /$0F_{hex}$	*Set AE-Bit*

8. Process input buffer:

00_{hex} /$0F_{hex}$	Reset AA and AE-Bit

9 Device Function

3. Read 30 bytes on Head 1, start address 10, with read error

Note
If an error occurs, the AF-Bit with corresponding error number is provided instead of the AE-Bit. Setting the AF-Bit cancels the job and declares it as ended.

Controller

1. Process output buffer (note sequence):

01_{hex}	Command 01_{hex}
02_{hex}	Start address $0A_{hex}$
03_{hex}	Start address 00_{hex}
04_{hex}	No. of bytes $1E_{hex}$
05_{hex}	No. of bytes 00_{hex}
00_{hex} /$0F_{hex}$	Set AV bit

3. Process input buffer:

01_{hex}	Copy error number

Process output buffer:

00_{hex} /$0F_{hex}$	Reset AV-Bit

Identification system

2. Process input buffer (note sequence):
If error occurs immediately!

00_{hex} /$0F_{hex}$	Set AA bit
01_{hex}	Enter error number
00_{hex} /$0F_{hex}$	Set AF-Bit

4. Process input buffer:

00_{hex} /$0F_{hex}$	Reset AA and AF-Bit

9 Device Function

4. Read 30 bytes on Head 1, start address 10, simultaneous data transmission, with read error

Controller

1. Process output buffer (note sequence):

01hex	Command 01hex
02hex	Start address 0Ahex
03hex	Start address 00hex
04hex	No. of bytes 1Ehex
05hex	No. of bytes 00hex
00hex /0Fhex	Set AV bit

Identification system

2. Process input buffer (note sequence):
If error occurs immediately!

00hex /0Fhex	Set AA bit
01hex	Enter error number
00hex /0Fhex	Set AF-Bit

3. Process input buffer:

01hex	Copy error number

Process output buffer:

00hex /0Fhex	Reset AV-Bit

4. Process input buffer:

00hex /0Fhex	Reset AA and AF-Bit

9 Device Function

5. Read 30 bytes on Head 1, simultaneous data transmission, start address 10, with read error

> **Note**
> If an error occurs after data have begun to be sent, the AF-Bit with corresponding error number is provided instead of the AE-Bit. The error message AF is dominant. Which data are faulty cannot be specified. Setting the AF-Bit cancels the job and declares it as ended.

Controller **Identification system**

1. Process output buffer (note sequence):

01_{hex}	Command 01_{hex}
02_{hex}	Start address $0A_{hex}$
03_{hex}	Start address 00_{hex}
04_{hex}	No. of bytes $1E_{hex}$
05_{hex}	No. of bytes 00_{hex}
00_{hex} /$0F_{hex}$	Set AV bit

2. Process input buffer (note sequence):

00_{hex} /$0F_{hex}$	Set AA bit
$01…0E_{hex}$	Enter first 14 bytes
00_{hex} /$0F_{hex}$	Invert TO-Bit

3. Process input buffer:

$01…0E_{hex}$	Copy first 14 bytes

Process output buffer:

00_{hex} /$0F_{hex}$	Invert TI-Bit

4. Process input buffer:
If error has occurred!

01_{hex}	Enter error number
00_{hex} /$0F_{hex}$	Set AF-Bit

5. Process input buffer:

$01…0E_{hex}$	Copy error number

Process output buffer:

00_{hex} /$0F_{hex}$	Reset AV-Bit

6. Process input buffer:

00_{hex} /$0F_{hex}$	Reset AA and AF-Bit

9 Device Function

6. Write 30 bytes on Head 1, start address 20

Controller

1. Process output buffer (note sequence):

01hex	Command 02hex
02hex	Start address 14hex
03hex	Start address 00hex
04hex	No. of bytes 1Ehex
05hex	No. of bytes 00hex
00hex /0Fhex	Set AV bit

Identification system

2. Process input buffer (note sequence):

00hex /0Fhex	Set AA bit, invert TO-Bit

3. Process output buffer:

01…0Ehex	Enter first 14 bytes
00hex /0Fhex	Invert TI-Bit

4. Process output buffer:

01…0Ehex	Copy first 14 bytes

Process input buffer:

00hex /0Fhex	Invert TO-Bit

5. Process output buffer:

01…0Ehex	Enter second 14 bytes
00hex /0Fhex	Invert TI-Bit

6. Process output buffer:

01…0Ehex	Copy second 14 bytes

Process input buffer:

00hex /0Fhex	Invert TO-Bit

7. Process output buffer:

01…02hex	Enter last 2 bytes
00hex /0Fhex	Invert TI-Bit

8. Process output buffer:

01…02hex	Copy last 2 bytes

Process input buffer:

00hex /0Fhex	Set AE-Bit

9. Process output buffer:

00hex /0Fhex	Reset AV-Bit

10. Process input buffer:

00hex /0Fhex	Reset AA and AE-Bit

9 Device Function

7. Mixed data access – Save program (3 data records)

1st data record	Start address	5	No. of bytes	7
2nd data record	Start address	75	No. of bytes	3
3rd data record	Start address	112	No. of bytes	17
Total number of bytes exchanged in the operation:				27 bytes

All 104 bytes are written for this programming.

Controller **Identification system**

1. Process output buffer (note sequence):

01_{hex}	Command 06_{hex}
02_{hex}	Program number 01_{hex}
00_{hex} /$0F_{hex}$	Set AV bit

2. Process input buffer (note sequence):

00_{hex} /$0F_{hex}$	Set AA bit, invert TO-Bit

3. Process output buffer:

01_{hex}	1st start address	05_{hex}
02_{hex}		00_{hex}
03_{hex}	1st no. of bytes	07_{hex}
04_{hex}		00_{hex}
05_{hex}	2nd start address	$4B_{hex}$
06_{hex}		00_{hex}
07_{hex}	2nd no. of bytes	03_{hex}
08_{hex}		00_{hex}
09_{hex}	3rd start address	70_{hex}
$0A_{hex}$		00_{hex}
$0B_{hex}$	3rd no. of bytes	11_{hex}
$0C_{hex}$		00_{hex}
$0D_{hex}$ /$0E_{hex}$	End delimiter $FF_{hex}FF_{hex}$	
00_{hex} /$0F_{hex}$	Invert TI-Bit	

4. Process input buffer:

00_{hex} /$0F_{hex}$	Invert TO-Bit

5. Process output buffer:

01_{hex}…$0E_{hex}$	(not used) $FF_{hex}FF_{hex}$
00_{hex} /$0F_{hex}$	Invert TI-Bit

6. Process input buffer:

00_{hex} /$0F_{hex}$	Invert TO-Bit

17. Process output buffer:

01_{hex}…$0E_{hex}$	(not used) $FF_{hex}FF_{hex}$
00_{hex} /$0F_{hex}$	Invert TI-Bit

18. Process input buffer:

00_{hex} /$0F_{hex}$	Set AE-Bit

19. Process output buffer:

00_{hex} /$0F_{hex}$	Reset AV-Bit

20. Process input buffer:

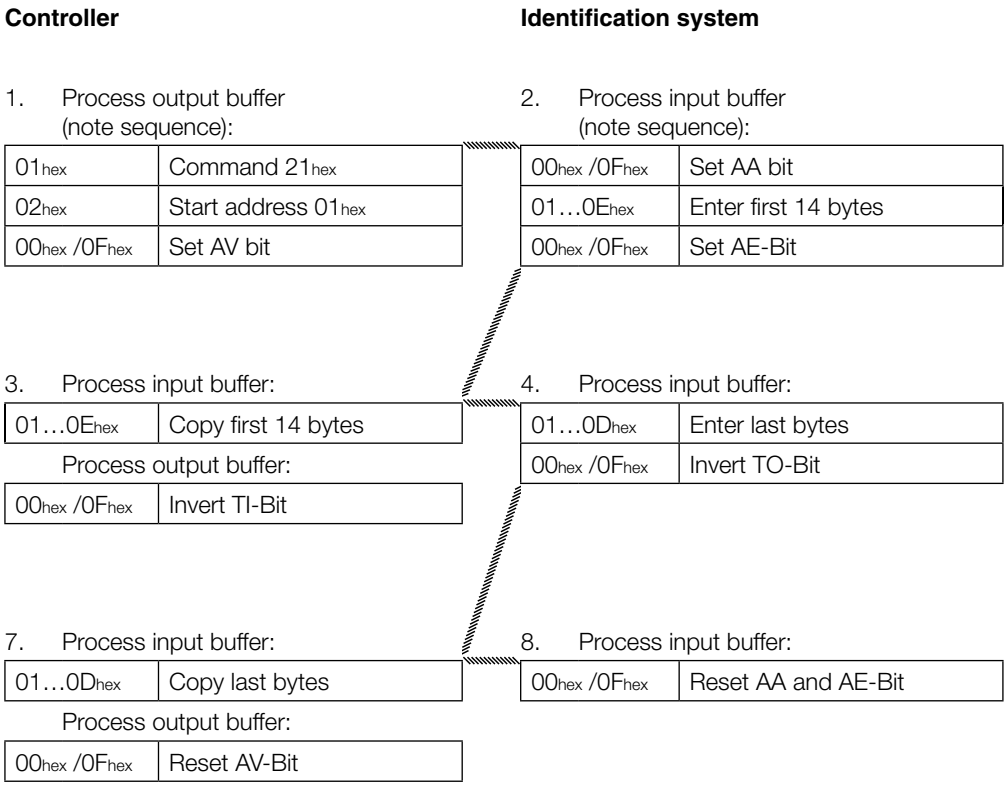
00_{hex} /$0F_{hex}$	Reset AA and AE-Bit

9 Device Function

8. Mixed data access – Reading the data carrier with Program No. 1

Note
Dynamic mode is turned off while the program is running.

A total of 27 bytes are exchanged.



9 Device Function

9. Mixed data access – Writing the data carrier with Program No. 1

Note
Dynamic mode is turned off while the program is running.

A total of 27 bytes are exchanged.

Controller **Identification system**

1. Process output buffer (note sequence):

01_{hex}	Command 22_{hex}
02_{hex}	Program number 01_{hex}
00_{hex} / $0F_{hex}$	Set AV-Bit

2. Process input buffer (note sequence):

00_{hex} / $0F_{hex}$	Set AA-Bit Invert TO-Bit

3. Process output buffer:

$01…0E_{hex}$	Enter first 14 bytes
00_{hex} / $0F_{hex}$	Invert TI-Bit

4. Process output buffer:

$01…0E_{hex}$	Enter first 14 bytes

Process input buffer:

00_{hex} / $0F_{hex}$	Invert TO-Bit

5. Process output buffer:

$01…0D_{hex}$	Enter last bytes
00_{hex} / $0F_{hex}$	Invert TI-Bit

6. Process output buffer:

00_{hex} / $0D_{hex}$	Copy last bytes

Process input buffer:

00_{hex} / $0F_{hex}$	Set AE-Bit

7. Process output buffer:

00_{hex} / $0F_{hex}$	Reset AV-Bit

8. Process input buffer:

00_{hex} / $0F_{hex}$	Reset AA-Bit Reset AE-Bit

9 Device Function

10. Initialize data carrier for CRC

CRC initializing is handled like a write command. The start address and number of bytes must correspond to the maximum used data quantity.
In the example the complete memory range of a 192-byte data carrier is used. 168 bytes of the data carrier are available as user bytes, since 24 bytes are required for the CRC.

Controller

1. Process output buffer (note sequence):

01hex	Command 12hex
02hex	Start address 00hex
03hex	Start address 00hex
04hex	No. of bytes A8hex
05hex	No. of bytes 00hex
00hex /0Fhex	Set AV bit

3. Process output buffer:

01…0Ehex	Enter first 14 bytes
00hex /0Fhex	Invert TI-Bit

5. Process output buffer:

01…0Ehex	Enter second 14 bytes
00hex /0Fhex	Invert TI-Bit

25. Process output buffer:

01…08hex	Enter last bytes
00hex /0Fhex	Invert TI-Bit

27. Process output buffer:

00hex /0Fhex	Reset AV-Bit

Identification system

2. Process input buffer (note sequence):

00hex /0Fhex	Set AA bit, invert TO-Bit

4. Process output buffer:

01…0Ehex	Copy first 14 bytes

Process input buffer:

00hex /0Fhex	Invert TO-Bit

6. Process output buffer:

01…0Ehex	Copy second 14 bytes

Process input buffer:

00hex /0Fhex	Invert TO-Bit

26. Process output buffer:

01…08hex	Copy last bytes

Process input buffer:

00hex /0Fhex	Set AE-Bit

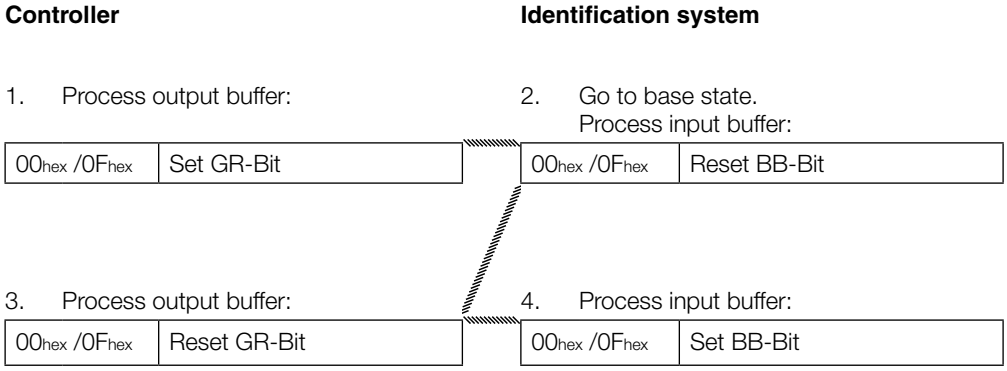
28. Process input buffer:

00hex /0Fhex	Reset AA and AE-Bit

9 Device Function

11. Set Read/Write Head 1 to base state

Both read/write heads in the identification system can be independently set to the base state.



12. Deselecting the Read/Write Head

In normal operation both read-write heads are selected. Setting the KA bit allows one or both read-write heads to be deselected (turned off).

Controller

1. Process output buffer:

00hex /0Fhex	Set KA-Bit

Resetting the KA bit selects the read/write head again.

Note
Selecting a read/write head may take up to a second. Deselecting it requires much less time.

Appendix

Ordering code

BIS L – 6026 – 034 – 050 – 06 – ST19

Balluff Identification system

Series L Read/Write System

Hardware Type
6026 = Metal enclosure, EtherNet/IP

Software Type
034 = EtherNet/IP

Version
050 = with two ports for external read/write heads BIS L-3_ _

Interface
06 = Ethernet

Customer connection
ST19 = Plug variant
X1 = Round connector for supply voltage (5-pin male)
X3 = Round connector for Ethernet (4-pole female)
X4 = Round connector for RS 232 interface (4-pin male)

Accessories (optional, not included in scope of delivery)

Type		Ordering code
Connector no cable:	for Head 1, Head 2	BKS-S117-00
Connection cable	for Head 1, Head 2; 5 m	BIS L-500-PU-05
	for Head 1, Head 2; 10 m	BIS L-500-PU-10
Connection cable: one end with a straight, molded-in connector (female), one end for user-assembled connector, lenght as desired.	for Head 1, Head 2; 25 m	BIS L-501-PU1-25
Connection cable: one end with a right-angle format, molded-in connector (female), one end for user-assembled connector, lenght as desired.	for Head 1, Head 2; 25 m	BIS L-502-PU1-25
Connector	for X1	BKS-S 79-00
	for X3	BKS-S 182-00
Cover cap	for X4	BES 12-SM-2
	for Head 1, Head 2	Cover cap M12 female (121 671)
Adapter cable M12 D coded to RJ45		BIS C-526-PVC-00,5

Appendix

ASCII Table

Decimal	Hex	Control Code	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	00	Ctrl @	NUL	43	2B	+	86	56	V
1	01	Ctrl A	SOH	44	2C	,	87	57	W
2	02	Ctrl B	STX	45	2D	-	88	58	X
3	03	Ctrl C	ETX	46	2E	.	89	59	Y
4	04	Ctrl D	EOT	47	2F	/	90	5A	Z
5	05	Ctrl E	ENQ	48	30	0	91	5B	[
6	06	Ctrl F	ACK	49	31	1	92	5C	\
7	07	Ctrl G	BEL	50	32	2	93	5D	[
8	08	Ctrl H	BS	51	33	3	94	5E	^
9	09	Ctrl I	HT	52	34	4	95	5F	_
10	0A	Ctrl J	LF	53	35	5	96	60	`
11	0B	Ctrl K	VT	54	36	6	97	61	a
12	0C	Ctrl L	FF	55	37	7	98	62	b
13	0D	Ctrl M	CR	56	38	8	99	63	c
14	0E	Ctrl N	SO	57	39	9	100	64	d
15	0F	Ctrl O	SI	58	3A	:	101	65	e
16	10	Ctrl P	DLE	59	3B	;	102	66	f
17	11	Ctrl Q	DC1	60	3C	<	103	67	g
18	12	Ctrl R	DC2	61	3D	=	104	68	h
19	13	Ctrl S	DC3	62	3E	>	105	69	i
20	14	Ctrl T	DC4	63	3F	?	106	6A	j
21	15	Ctrl U	NAK	64	40	@	107	6B	k
22	16	Ctrl V	SYN	65	41	A	108	6C	l
23	17	Ctrl W	ETB	66	42	B	109	6D	m
24	18	Ctrl X	CAN	67	43	C	110	6E	n
25	19	Ctrl Y	EM	68	44	D	111	6F	o
26	1A	Ctrl Z	SUB	69	45	E	112	70	p
27	1B	Ctrl [	ESC	70	46	F	113	71	q
28	1C	Ctrl \	FS	71	47	G	114	72	r
29	1D	Ctrl]	GS	72	48	H	115	73	s
30	1E	Ctrl ^	RS	73	49	I	116	74	t
31	1F	Ctrl _	US	74	4A	J	117	75	u
32	20		SP	75	4B	K	118	76	v
33	21		!	76	4C	L	119	77	w
34	22		"	77	4D	M	120	78	x
35	23		#	78	4E	N	121	79	y
36	24		$	79	4F	O	122	7A	z
37	25		%	80	50	P	123	7B	{
38	26		&	81	51	Q	124	7C	|
39	27		‘	82	52	R	125	7D	}
40	28		(	83	53	S	126	7E	~
41	29		)	84	54	T	127	7F	DEL
42	2A		*	85	55	U			

Index

www.balluff.com

Balluff GmbH
Schurwaldstrasse 9
73765 Neuhausen a.d.F.
Germany
Phone +49 7158 173-0
Fax +49 7158 5010
balluff@balluff.de
www.balluff.com

No. 848415 E | 01.107406 | Edition 0703 | Replaces edition 0701 | Subjects to modification.