EZ222-DN DeviceNet Slave Interface

User Manual

May 2005

Warning!
Dangerous electrical voltage!

Before commencing the installation

- Disconnect the power supply of the device.
- Ensure that devices cannot be accidentally restarted.
- Verify isolation from the supply.
- Short circuit to earth.
- Cover or enclose neighboring units that are live.
- Follow the engineering instructions (AWA) of the device concerned.
- Only suitably qualified personnel in accordance with EN 50110-1/-2 (VDE 0105 Part 100) may work on this device/system.
- Before installation and before touching the device ensure that you are free of electrostatic charge.
- The functional earth (FE) must be connected to the protective earth (PE) or to the potential equalization. The system installer is responsible for implementing this connection.
- Connecting cables and signal lines should be installed so that inductive or capacitive interference does not impair the automation functions.
- Install automation devices and related operating elements in such a way that they are well protected against unintentional operation.
- Suitable safety hardware and software measures should be implemented for the I/O interface so that a line or wire breakage on the signal side does not result in undefined states in the automation devices.
- Ensure a reliable electrical isolation of the low voltage for the 24 volt supply. Only use power supply units complying with IEC 60364-4-41 (VDE 0100 Part 410) or HD 384.4.41 S2.
- Deviations of the mains voltage from the rated value must not exceed the tolerance limits given in the specifications, otherwise this may cause malfunction and dangerous operation.
- Emergency stop devices complying with IEC/EN 60204-1 must be effective in all operating modes of the automation devices.Unlatching the emergency-stop devices must not cause restart.

- Devices that are designed for mounting in housings or control cabinets must only be operated and controlled after they have been installed with the housing closed. Desktop or portable units must only be operated and controlled in enclosed housings.
- Measures should be taken to ensure the proper restart of programs interrupted after a voltage dip or failure. This should not cause dangerous operating states even for a short time. If necessary, emergency-stop devices should be implemented.
- Wherever faults in the automation system may cause damage to persons or property, external measures must be implemented to ensure a safe operating state in the event of a fault or malfunction (for example, by means of separate limit switches, mechanical interlocks etc.).

Contents

About this manual

Target group

This manual is targeted at automation technicians and engineers. Expert knowledge of the DeviceNet fieldbus and programming of a DeviceNet master PLC is assumed. Furthermore, you should be familiar with the handling of the EZ control relay and the EZD HMI control.

Further manuals for this device

The following manuals apply:

- EZ500/700 Series (MN05013003E)
- EZ800 Series (MN05013004E)
- EZD Series (MN05013005E)

All manuals are available on the Internet for download as PDF files. Enter the document number at www.EatonElectrical.com as a search term in order to quickly locate the required manual.

References

[1] DeviceNet Specification Volume One CIP

[2] DeviceNet Specification Volume Two Ethernet/IP

References are available on the Internet at www.odva.org

Device name

The following short names for equipment types are used in this manual, as far as the description applies to all of these types:

- EZ500 for
  - EZ512-AB...
  - EZ512-AC
  - EZ521-DA…
  - EZ512-DC
- EZ700 for
  - EZ719-AB…
  - EZ719-AC…
  - EZ719-DA…
  - EZ719-DC…
  - EZ721-DC…
- EZ800 for
  - EZ819-...
  - EZ820-...
  - EZ821-...
  - EZ822-...
- EZD-CP8.. for
  - EZD-CP8-ME
  - EZD-CP8-NT
- EZ-AB for
  - EZ512-AB...
  - EZ719-AB...
- EZ-AC for
  - EZ512-AC-..
  - EZ719-AC
  - EZ8..-AC-...

- EZ-DC for
  - EZ512-DC-..
  - EZ719-DC-..
  - EZ8..-.DC-...
- EZ-DA for
  - EZ512-DA...
  - EZ719-DA...

Abbreviations and symbols

Meaning of abbreviations and symbols used in this manual:

BCD	Binary Coded Decimal code
CAN	Controller Area Network
DEC	Decimal (number system based on 10s)
HEX	Hexadecimal (Number system based on 16)
MAC ID	Media Access Control Identifier
ODVA	Open DeviceNet Vendor Association
PC	Personal Computer
SELV	Safety Extra Low Voltage“
UCMM	Unconnected Message Manager

Writing conventions

For greater clarity, the name of the current chapter is shown in the header of the left-hand page and the name of the current section in the header of the right-hand page. Pages at the start of a chapter and empty pages at the end of a chapter are exceptions.

▶ indicates actions to be taken.

Note!
Warns of a hazardous situation that could result in damage to the product or components.

Caution!
Warns of the risk of major damage to assets and minor injury.

Warning
Warns of the possibility of a hazardous situation that could result in major damage and serious or fatal injury or even death.

→ Draws your attention to interesting tips and supplementary information

1 The EZ222-DN

The EZ222-DN communication module has been developed for automation tasks with the DeviceNet field bus. EZ222-DN acts as a “gateway” and can only be operated in conjunction with the expanded EZ700, EZ800 or EZD basic units.

The system unit consists of the EZ/EZD control device and the EZ222-DN gateway and operates exclusively as a slave station on the fieldbus system.

System overview

The EZ222-DN slaves are integrated into a DeviceNet fieldbus system.

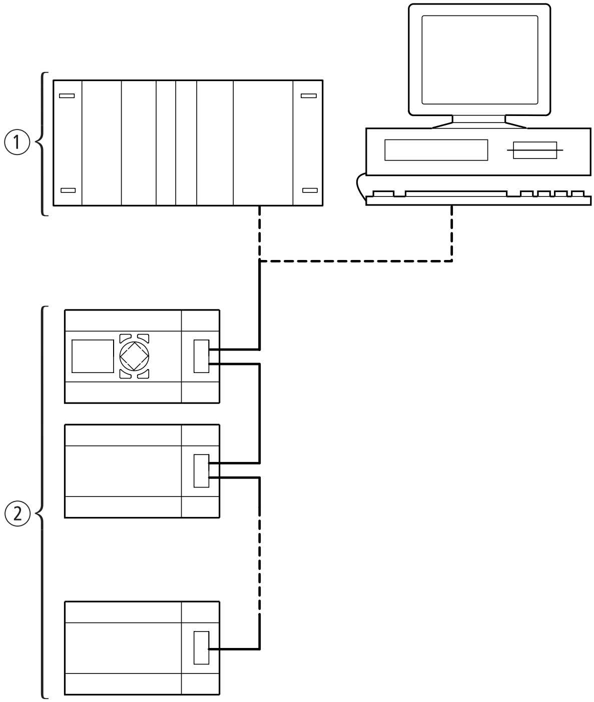


Figure 1: Implementation of EZ222-DN in the DeviceNet

① Master area, PLC or PC with DeviceNet card

② Slave area, e.g.: Control relay EZ/EZD with DeviceNet interface

Structure of the unit

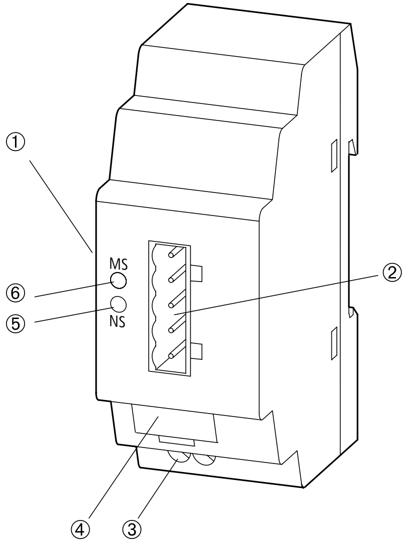


Figure 2: Structure of EZ222-DN

① EZ-LINK socket
② 5-pin DeviceNet connection to ODVA
③ Power supply 24 V
④ Equipment rating plate
⑤ Network Status LED NS
⑥ Module Status LED MS

EZ222-DN Communication profile

- Predefined master/slave communication settings
  - The **I/O polling** connection is used for the transfer of 3 bytes of input data (R1 to R16) and 3 bytes of output data (S1 to S8) between the EZ base unit with gateway interconnection and the DeviceNet PLC.
  - The **I/O Change of State/Cyclic** connection (acknowledged, unacknowledged) is used to transfer 2 bytes of diagnostic data from the EZ control relay to DeviceNet the PLC.
  - The **explicit connection set-up** is used for read/write access to function relay parameters in the EZ control relay. This type of connection set-up also supports the configuration, diagnostics and management services of the control relay.
- DeviceNet Communication adapter profile (device type 12), which has been expanded by EZ requests
- Group 2 server
- UCMM-capable device
- Dynamic set-up of explicit and I/O connections are possible
- Device Heartbeat Message
- Device Shutdown Message
- Offline communication settings

Use other than intended

EZ may not be used to replace safety-relevant control circuits, e.g.:

- Furnace,
- emergency-stop,
- crane or
- Two-hand safety controls.

2 Installation

Applicable are the same guidelines as for EZ/EZD basic units with expansion modules.

EZ222-DN connection to the basic unit

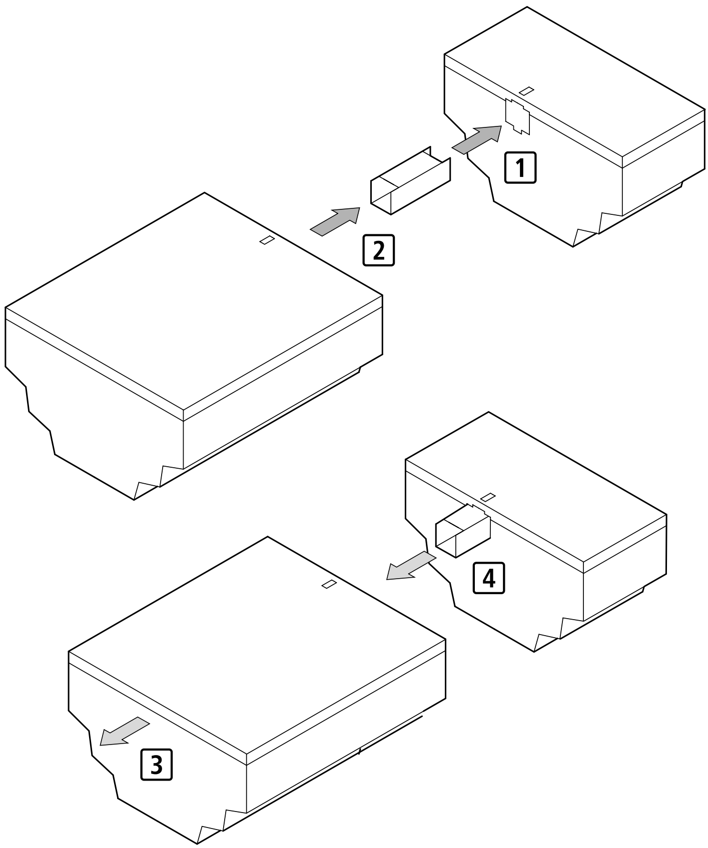


Figure 3: Mounting the EZ222-DN on the basic unit

1 + 2 Installation

3 + 4 Removal

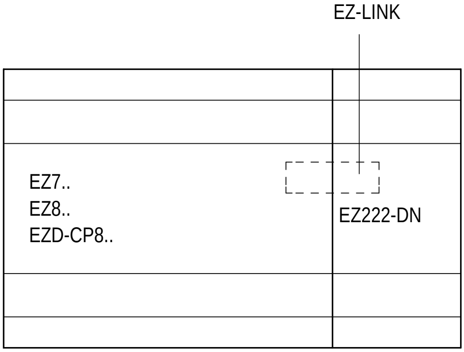
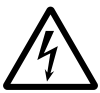


Figure 4: Connection between basic unit and EZ222-DN

Connecting the power supply

EZ222-DN operates with a 24 VDC supply voltage (→ section “Power supply”, page 192).



Warning
Always ensure safe electrical isolation between the extra low voltage (SELV) and the 24-V power supply.

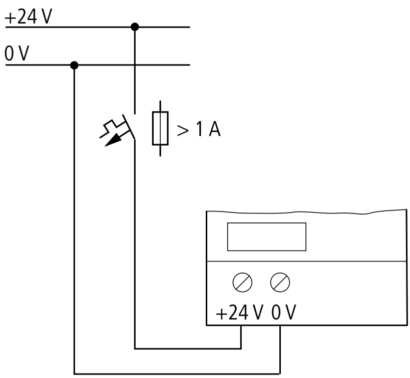


Figure 5: Power supply of EZ222-DN

Connecting DeviceNet

A 5-pole DeviceNet plug connects the DeviceNet interface of thedevice to the DeviceNet fieldbus.

Please use a special DeviceNet plug and DeviceNet cable for this connection. Both are specified in the ODVA. The type of cable has an influence on the maximum available length of the bus line and thus on the data transfer rate.

Pin assignment DeviceNet

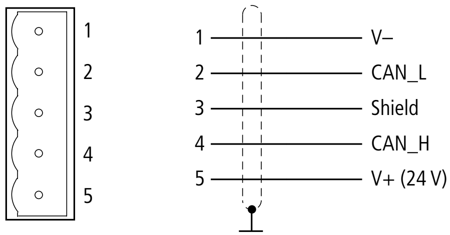


Figure 6: Pin assignment of the equipment socket

1	GND	black
2	CAN_L	blue
3	screen	clear
4	CAN_H	white
5	24 V	red

All pins of the plug must be connected to ensure safe communication of the EZ222-DN on the fieldbus DeviceNet. This also applies to the 24-V bus voltage.

→ The gateway therefore does not participate in communication on the bus if the bus voltage is not available. The Network status LED indicates OFF mode in this situation.

Terminating resistors

The first and last node of a DeviceNet network must be terminated by means of a 120 Ω bus termination resistor. This device is interconnected between the CAN_H and CAN_L terminals.

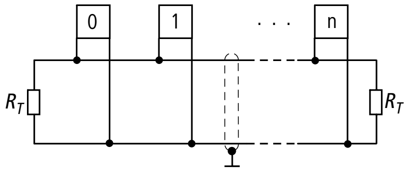


Figure 7: Terminating resistors R_T: CAN_H and CAN_L terminals

R_T = 120 Ω

EMC compatible wiring

Electromagnetic interference may lead to unwanted effects on the communications fieldbus, which can be significantly reduced by using the cable described above, a shielded RJ45 connector and by terminating the screen.

The two figures below show the correct termination of the shielding.

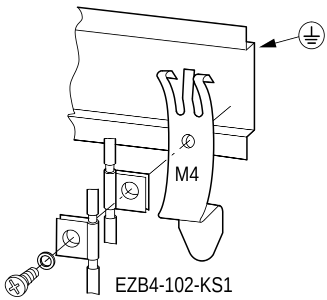


Figure 8: Shielding connection to the mounting rail

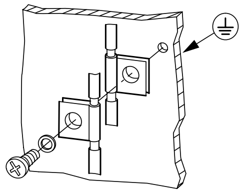


Figure 9: Shielding connection to the mounting plate

Potential isolation

The following potential isolation specifications apply to EZ222-DN interfaces:

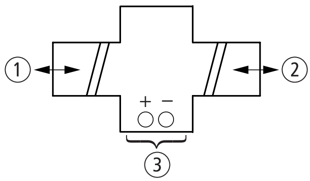


Figure 10: Potential isolation between the supply voltage and outputs

① Safe electrical isolation between EZ-LINK and the 240 VAC mains

② Simple electrical isolation to the DeviceNet communication bus

③ Power supply 24 V DC

Data transfer rates – automatic baud rate recognition

After it is switched on, the EZ222-DN module automatically detects the data transfer rate of the communication network. However, this is possible only if at least one network node transmits valid message frames. The device supports the following data transfer rates according to ODVA:

- 125 kbps
- 250 kbps
- 500 kbps

Maximum distances and bus cable lengths

The max. bus length is not determined by the data transfer rate, but rather by the cable used. The following cables are permitted:

- A so-called "Thin Cable",
- a "Thick Cable"
- or a "Flat Cable".

The data cable requirements are specified by the ODVA.

Baud rate (kbps)	max. bus length in m		
	"Thick Cable"	"Thin Cable"	"Flat Cable"
125	500	100	420
250	250	100	200
500	100	100	100

3 Operating the device

Initial power on

▶ Before you switch on the device, verify that it is properly connected to the power supply, to the bus connectors and to the basic unit.

▶ Switch on the power supply for the basic unit and the EZ222-DN.

The LEDs of the EZ222-DN flicker.
The device is in the mode for detection of the correct baud rate (→ section "Data transfer rates – automatic baud rate recognition" on page 18).
The GW information (intelligent station connected) is displayed on the basic unit.

Basic unit	GW display
EZ700	flashing
EZ800	flashing
EZD-CP8..	flashing

As soon as the device in the network management is switched to the "Operational" status, the state of the GW changes to static even on the devices with a flashing GW, (→ section "Network status LED" on page 24).

If the EZ222-DN has factory settings (node ID = 127), you need to define the DeviceNet slave address.

DeviceNet setting the slave address

Each DeviceNet slave requires a unique address (MAC ID) in the DeviceNet structure. Within a DeviceNet structure, you can assign a maximum of 64 addresses (0 to 63). Each MAC ID must be unique within the entire bus structure.

There are three ways to set the DeviceNet address of an EZ222-DN:

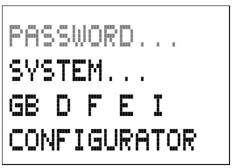
- Using the integrated display and keyboard on the EZ basic unit
- Using EZSoft on the PC
- Using the configuration software of the installed master PLC (possibly by means of an explicit message).

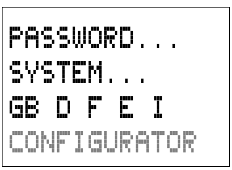
Setting the address at the basic unit with display:

Precondition

- The respective basic units (EZ700, EZ800 or EZD) and EZ222-DN are supplied with voltage.
- The basic unit is accessible (password protection not activated).
- The basic unit has a valid operating system version.
- The basic unit is in STOP mode.

▶ Press the DEL + ALT shortcut to change to the special menu.





▶ Use the cursor keys ︿ or ﹀ to change to the CONFIGURATOR.

▶ Confirm with OK.

```
NET...
LINK...
```

▶Select the LINK.... menu with the EZ800/EZD units

OK

▶Confirm with OK.

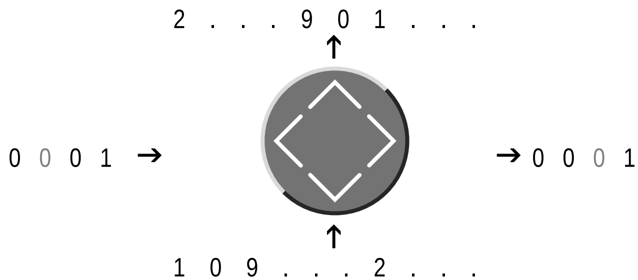
The DEVICENET menu appears.

```
DEVICENET

MAC ID 0026
222-01.20- D
```

▶Set the address by means of the cursor keys:

- Set the current numeric value via the ︿ or ﹀ keys.
- You can change the current numeric value via ‹ or ›.



OK

▶Accept the address with OK.

ESC

▶Cancel address input with ESC.

Information about the 4th display line:

xxx - xx . xx - xx

222 - 02 . 10 - B

- Hardware version, Index: b
- Software version, OS version: 2.1
- Device identity: EZ222-DN

Setting the address by means of EZSoft

‹Menu → Communication → Configuration → Expansion units → EZ222-DN›.

→ The menu is only available in the communication view; therefore please activate the “Communication” tab.

→ The following applies for device version identity 01:

After you have modified the MAC ID via the basic unit, restart the EZ222-DN by switching power off and on. EZ222-DN devices with a version identity > 01 automatically accept the address.

Setting the address via the master PLC

The configuration software supplied with your master PLC offers a further option of setting or modifying the MAC ID of the gateway. For more information, refer to the included PLC documentation.

You can also use various other software packages to modify the MAC ID, e.g. by sending an explicit message. Do so by using the corresponding service of the DeviceNet object (section “DeviceNet object”, page 33).

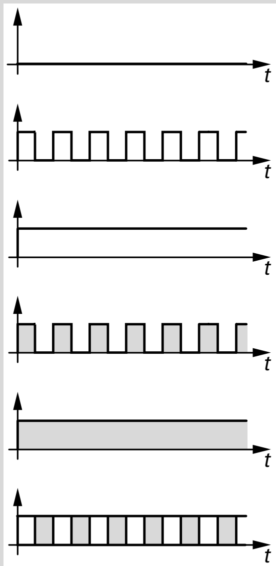
LED status displays

The expansion module EZ222-DN is equipped with two indicator LEDs for quick diagnostics. EZ222-DN monitors itself as well as the DeviceNet communication bus.

Module status LED

The dual-color LED (GREEN/RED) indicates the status of EZ222-DN. It monitors whether the device is fully functional and operates without fault.

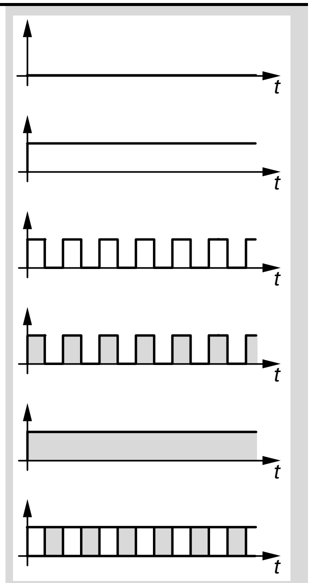
OFF	No power supply at the EZ222-DN.
GREEN	EZ222-DN is in normal operational state.
GREEN flashing	EZ222-DN is in standby mode. The configuration is faulty or incomplete, or a configuration does not exist.
RED flashing	An error has occurred. There is no need to replace the EZ222-DN.
RED	A fatal error has occurred. EZ222-DN must be replaced.
GREEN-RED flashing	EZ222-DN is performing a self-test.



Network status LED

The dual-color LED (GREEN/RED) indicates the status of the DeviceNet communication bus. This function monitors operability and correct operation of the EZ222-DN.

OFF	EZ222-DN is offline. Either it is performing a DUP_MAC_ID test or power is missing at the device or bus.
GREEN flashing	EZ222-DN is online. Communication has not yet been established.
GREEN	EZ222-DN is online and the connection is active.
RED flashing	Time-out of at least one I/O connection (time-out state).
RED	A fatal network error has occurred. EZ222-DN has shut down communication.
GREEN-RED flashing	EZ222-DN has detected a network access error and is now in communication error state.



Cycle time of the EZ basic unit

Network traffic between the EZ/EZD basic unit and the EZ222-DN via EZ-LINK extends the cycle scan time of the basic unit

In the worst case, this time can be extended by 25 ms.

Please take this factor into account when you calculate the response times of the basic unit.

EDS file

You can implement EZ222-DN into the DeviceNet structure by means of a standardized EDS file (Electronic Data Sheet).

This EDS file primarily defines the polled I/O connection, the COS I/O connection and the cyclic I/O connection of the gateway. It does not contain data or parameters (EZ object) for functions of the EZ basic unit. These functions are accessed by means of explicit messages.

The file EZ222DN.eds can be obtained at www.EatonElectrical.com. The file is also available on the EZSoft CD-ROM.

→ Note on the EDS file:

The Identity Object entry - Major Revision defines the current operating system state of the EZ222-DN communication module. As the device with a newer operating system version can deviate from the EDS description in this point, this entry must be modified accordingly, → section “Identity Object” on page 31.

4 DeviceNet functions

Object model

EZ222-DN is based on the Communications Adapter Profile according to ODVA specifications (Release V2.0).

The DeviceNet object model can be used to describe all EZ222-DN functions. The object model reflects the principle of communication at the application layer. This manual deals in the following only with objects relevant for your application. Primary topic is the manufacturer-specific class EZ object.

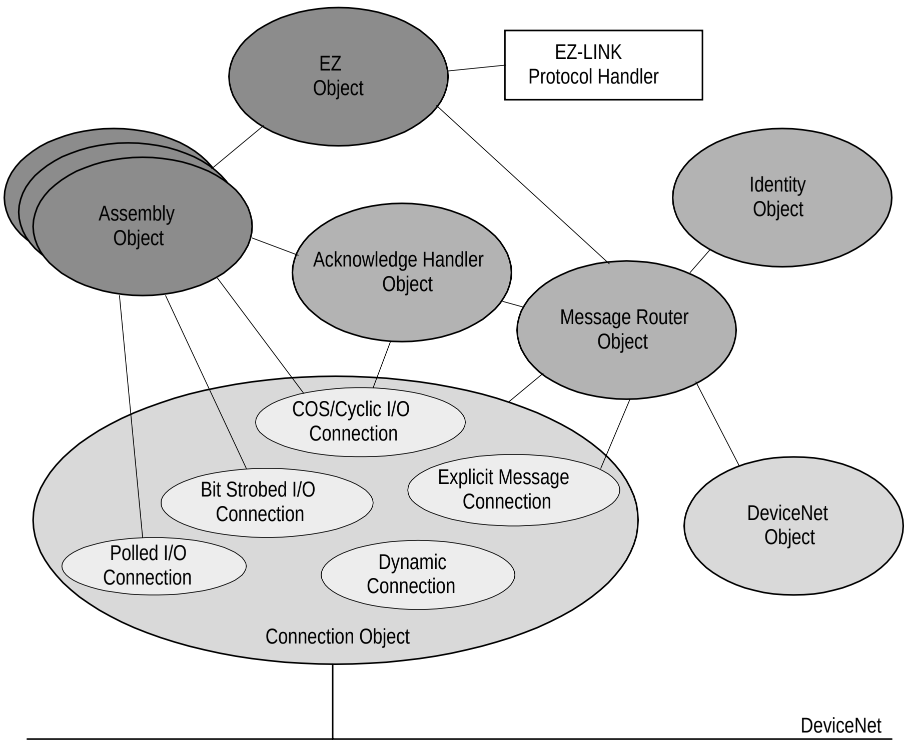


Figure 11: DeviceNet objects

The DeviceNet objects in the illustration can be compiled again as “Management objects”, “Connection objects” and “Manufacturer-specific objects”. Their tasks will be briefly explained after the following.

	Object address		Service address	Function
	Class ID [hex]	Instance ID [hex]	[hex]	Attribute ID [hex]
① Management objects				
Identity Object	01	01		→ page 29
Message Router	02	01		
② Connection objects				
DeviceNet Object	03	01		→ page 29
Connection Object	05	01 – 04, 04 – 0F		
③ Manufacturer-specific objects				
EZ Object	64	01		→ page 34
Direct access: inputs/outputs, mode				
Read			0E	→ chapter 5
Write			10	
Extended access: time, image data, function blocks			32	
EZ700				→ chapter 6
EZ800/EZD				→ chapter 7
Assembly Object	04	64 – 66		

① Management objects
These define DeviceNet-specific data and functions and must be supported by all DeviceNet devices:

- Identity Object

The Identity Object (Class ID 01_{hex}) contains all data for unique identification of a network node, e.g. the Vendor ID, Device Type and Product Code. It also comprises the actual status of a device, the serial number and the product name.

Detailed information → page 31.

- Message Router Object

The Message Router Object (Class ID 02_{hex}) provides access to all classes and instances in the device by means of explicit messages.

② Connection objects
Define messages exchanged via DeviceNet:

- DeviceNet Object

All devices must support the DeviceNet object (Class ID: 03_{hex}). It defines the physical interconnection of a device to the DeviceNet network, meaning it also contains the device address (MAC ID) and the currently set transmission speed, for example.

Detailed information → page 33.

- Connection Object

The Connection Object (Class ID: 05_{hex}) is supported by all DeviceNet devices in at least one instance. It defines the access to data via I/O messages or explicit messages, the path and length of producer/consumer data, the CAN connection identifier, the watchdog and the error response.

③ Manufacturer-specific objects

Define device-specific data and functions (Application Objects, Parameter Object, Assembly Object).

- Application Objects – EZ Object

Application objects (Class ID: 64_{hex}) describe simple applications for automation engineering. They are either predefined in the DeviceNet object library or by the user.

Detailed information → page 34.

- Assembly Objects

The Assembly Object (Class ID: 04_{hex}) provides the user with mapping options, i.e. attribute data of different instances in different classes can be grouped together to form a single attribute of an instance in an assembly object.

Identity Object

Object address		Function	Access
Class ID	Instance ID	Attribute ID	Service code
01_{hex}	01_{hex}	→ table 1	→ table 2

Table 1: Attribute IDs of the Identity Object instance

Attribute ID	Access	Name	Description	Size [byte]
1	Read	Vendor ID	The ODVA specifies the Vendor ID. See the EDS file.[1)]	2
2	Read	Device type	The EZ222-DN belongs to the communication adapters category. Its value is 12_{dec}.	2
3	Read	Product code	The product code is defined by Eaton and describes the model number. See the EDS file.[1)]	2
4	Read	Device version	Two bytes are returned when reading the device version.	
		Hardware version,	The low byte defines the hardware version, the high byte the operating system version.	1
		Operating system version		1
5	Read	Status	This attribute describes the global status of the device.	2
6	Read	Serial number	The serial number of the device can be read with this attribute.	4
7	Read	Product name	The product name EZ222-DN is stored as hex value in ASCII format.	12

Attribute ID	Access	Name	Description	Size [byte]
9	Read	Configuration consistency value	This attribute returns a counter value that monitors the number of modifications in non-volatile memory (E2PROM).	2
10	Read/ Write	Heartbeat Interval	Defines an interval between heartbeat messages in [s].	2

1) The file EZ222DN.eds can be obtained at www. EatonElectrical.com. The file is also available on the EZSoft CD-ROM.

Service code

The Identity Object Instance and also the following instances support the services listed in the table below.

Table 2: Service code

Service code value	Service name	Description
05_{hex}	Reset	Calls the reset function of the communication module EZ222-DN.
$0E_{hex}$	Get_Attribute_Single	This service can be used to fetch the value of a selected attribute from the communication module.
10_{hex}	Set_Attribute_Single	This service can be used to set a selected attribute in the device.

DeviceNet object

Object address		Function	Access
Class ID	Instance ID	Attribute ID	Service code
03_{hex}	01_{hex}	→ table 3	→ table 2

The DeviceNet object instance is used to configure the communication module EZ222-DN and to define the physical environment. The Service Codes used for the Identity Object also apply in this case.

Table 3: Attribute IDs of the DeviceNet Object instance

Attribute ID	Access	Name	Description	Size [byte]
1	Read/Write	MAC ID	The MAC ID represents the network address of a network node. It can be read and set for EZ222-DN via the fieldbus by means of this attribute. Range of values: 0 to 63_{dec}. (→ section “DeviceNet setting the slave address”, page 20)	1
2	Read/Write	Baud rate	This attribute can be used to read/set the data transfer rate for communication functions. Range of values: 0 to 2, 125 to 500 kbps (→ section “Data transfer rates – automatic baud rate recognition”, page 18).	1
3	Read/Write	BOI (Bus-Off interrupt)	This attribute can be used to define the reaction to a Bus-Off event (CAN-specific).	1
4	Read/Write	Bus-Off counter	This value shows how often a Bus-Off event has occurred. Range of values: 0 to 255.	1

EZ Object

Object address		Function	Access
Class ID	Instance ID	Attribute ID	Service code
64_{hex}	01_{hex}	→ table 4	→ table 5

The EZ object can be used to access EZ/EZD functions via the DeviceNet communication bus . The table below shows the attributes supported by this object. The two bytes of attributes 1 and 2 provide the diagnostic data of the device. You can use attribute 3 to access the outputs (S1 to S8) and attribute 4 to access the inputs (R1 of R16) of the basic unit.

By using a DeviceNet configuration software (e.g. RS Networx), you can map these data directly to the corresponding memory areas of a PLC.

Table 4: Attribute IDs of the EZ Object instance

Attribute ID	Access	Name	Description	Size [byte]
1	Read	EZ Status	This attribute can be used to read the status of EZ (RUN or STOP).→ table 6	1
2	Read	Coupling Module Status	This attribute can be used to read the status of EZ-LINK.→ table 6	1
3	Read	Inputs – Send Data	EZ transfers the input data to the DeviceNet bus. The EZ outputs S1 to S8 must be used for this function. The structure of these 3 bytes is described in detail under page 45, section “Input data:Mode, S1 – S8” .	3

Attribute ID	Access	Name	Description	Size [byte]
4	Read/ Write	Outputs – Receive Data	The DeviceNet bus transfers the data to EZ. The EZ inputs R1 to R16 must be used for this function. The structure of these 3 bytes is described in detail under page 47, section “Output data:mode, R1 – R16”, .	3
5	Read/ Write	Predefined Outputs	This attribute can be used to preset the output data ("R" data) at the EZ222-DN during start-up. The structure of these 3 bytes is described in detail under section “Output data:mode, R1 – R16”, page 47.	3

Service code

The EZ object instance supports the following services.

Table 5: Service code

Service code value	Service name	Description
$0E_{hex}$	Get_Attribute_Single	This service can be used to fetch the value of a selected attribute from the communication module.
10_{hex}	Set_Attribute_Single	This service can be used to set a selected attribute in the device.
32_{hex}	Extended access[1)]	This service can be used to address the supplementary parameters[1)] of the control relay:

1) Additional parameters are “Time”, “Image data” and “Function block”. Addressing of the parameters is EZ specific and is described in chapters 5 and 6 in detail.
Extended access is implemented via explicit message transfer. This transfer protocol allows the exchange of control data. Further information about the transfer protocol can be found in section “DeviceNet Communication profile” on page 37.

Change of State I/O connection

Table 6: Diagnostics data: 2 Byte

Byte	Meaning	Value	Meaning
0	EZ status (attribute ID 1)	00_{hex}	Static value.
1	Coupling module status (attribute ID 2)	00_{hex}	The basic unit is connected to the EZ222-DN gateway via EZ-LINK.
		04_{hex}	The basic unit is either switched off or disconnected from the EZ222-DN gateway via EZ-LINK.

→ When communication between the basic unit EZ/EZD and the expansion unit EZ222-DN goes down, a corresponding error code will be generated in the third data byte. Furthermore, the Rx/Tx data of the gateway will be transferred with the value 00_{hex}.

DeviceNet Communication profile

DeviceNet is based on a connection-oriented communications model, i.e. data are exchanged only via the specific connections assigned to the units.

DeviceNet stations communicate either by means of I/O messages or explicit messages.

I/O Messages

I/O messages are used for exchanging high-priority process and application data across the network. Communication between DeviceNet nodes is based on the client/server model, i.e. a "producer" application transfers data to one or several "consumer" applications. It is quite possible in this case that several application objects are addressed in the same unit.

Prerequisite for communication between the units via I/O messages is the implementation of an I/O Messaging Connection Object. You can activate this function in two ways:

- Either by means of a static and in the unit already existing "I/O connection object" or via the “Predefined Master/Slave Connection Set“, or
- via a dynamically configured "I/O connection object", which you can configure using an Explicit Messaging Connection Object that already exists in the unit.

Explicit Messages

Explicit messages are used for exchanging low-priority configuration data, general management data or diagnostics data between two specific units across the PtP connection in a client/server system, in which the server always has to acknowledge client requests.

Same as for I/O messaging, the prerequisite for explicit messaging is the implementation of a “Connection Object“, namely the Explicit Messaging Connection Object“. This can be achieved either by activating an existing static connection object in the unit, or via the Predefined Master/Slave Connection Set“, or dynamically across the so-called UCMM port (Unconnected Message Manager Port) of a device.

All data of the function relay (EZ basic unit) are processed by means of explicit messages. The master PLC can thus read/ write access the parameters of the following functions.

- Time
- Image data
- Function blocks (counters, timers, analog value comparators,...).

General method of operation

The general method of operation with the EZ222-DN should be presented in the following. The acyclic data transfer is realized with the aid of explicit messages. The function blocks of the EZ basic unit can be addressed via the service code = 32_{hex}. The assigned attribute ID is here used to distinguish between different parameters and functions.

Service code	Object address	
	Class ID	Instance ID
32_{hex}	64_{hex}	01_{hex}

Digression:
DeviceNet is based on the standard CAN protocol and therefore uses an 11 bit message identifier. As a result 2^{11} = 2048 messages (000_{hex} - $7FF_{hex}$) are distinguishable. Six bits are sufficient for identification of a device as a DeviceNet network is limited to a maximum of 64 stations. These are referred to as the MAC-ID (device or node address).

Four message groups of differing sizes are available to suit the utilization model.

In DeviceNet language terms the CAN identifier is referred to as the Connection ID. This is comprised of the identifier for the message group (Message ID) and the MAC ID of the device:

- The source and target addresses are possible as the MAC ID; the definition is dependant on the message group and message ID.
- The significance of the message is defined in the system with the message ID.

Four message groups are available in the DeviceNet world. The EZ222-DN uses message group 2. This group uses 512 CAN identifiers (400_{hex} - $5FF_{hex}$). Most of the message IDs defined for this group are optional and defined for use of the "Predefined Master/Slave Connection Sets". A message ID is used for network management. The priority is primarily determined by the device address and then by the message ID. If the bit position is examined in detail, you will find that a CAN controller with an 8 bit mask is capable of filtering out its group 2 messages.

Connection ID = CAN identifier											Meaning
10	9	8	7	6	5	4	3	2	1	0	
1	0	MAC ID						Message ID			Message group 2
1	0	Source MAC ID						0	0	0	Master's I/O Bit–Strobe Command Message
1	0	Source MAC ID						0	0	1	Reserved for Master's Use – Use is TBD
1	0	Destination MAC ID						0	1	0	Master's Change of State or Cyclic Acknowledge Message
1	0	Source MAC ID						0	1	1	Slave's Explicit/ Unconnected Response Messages
1	0	Destination MAC ID						1	0	0	Master's Explicit Request Messages
1	0	Destination MAC ID						1	0	1	Master's I/O Poll Command/Change of State/Cyclic Message
1	0	Destination MAC ID						1	1	0	Group 2 Only Unconnected Explicit Request Messages
1	0	Destination MAC ID						1	1	1	Duplicate MAC ID Check Messages

Source: ODVA- DeviceNet Specification Release 2.0, Chapter 7-2

The data transfer on the DeviceNet communication bus is indicated in the following table. The data flow indicates the telegram for reading the date and time in the EZ700 (→ section "Read/write date and time" on page 53).

The EZ222-DN communication module has MAC ID = 3. It must be noted with the data stream that access is implemented in fragmented form. More information can be found in the ODVA specification.

Description		ID (hex)	Length	DeviceNet – Byte (hex) 0	1	2	3	4	5	6	7
Master sends a request (hex) with:		41C	8	80	00	32	64	01	93	05	00
Byte 2 - service code = 32 Byte 3 - CLASS ID = 64 Byte 4 - Instance ID = 01	DeviceNet specific										
Byte 5 - Attribute ID = 93 Byte 6 - Len = 05 Byte 7 - Index = 0	EZ-LINK specific										
Confirmation of the slave (Fragmentation protocol)		41B	3	80	C0	00					
Master sends remaining EZ-LINK byte		41C	6	80	01	00	00	00	00		
Byte 2 - Data 1 = 00 Byte 3 - Data 2 = 00 Byte 4 - Data 3 = 00 Byte 5 - Data 4 = 00											
Acknowledgement of the slave (Fragmentation protocol)		41B	3	80	C1	00					
Slave sends a response to the request		41B	8	80	00	B2	C2	05	00	05	09
Byte 3 – response = C2 (read successful) Byte 4 – Len = 05 Byte 5 – Index = 00 Byte 6 – Data 1 = 05											
Acknowledgement from master (Fragmentation Protocol)		41C	3	80	C0	00					

Description	ID (hex)	Length	DeviceNet – Byte (hex) 0	1	2	3	4	5	6	7
Slave sends remaining EZ-LINK data: Data 2 = 0D Data 3 = 05 Data 4 = 04	41B	5	80	81	0D	05	04			
Acknowledgement from master (Fragmentation protocol)	41C	3	80	C1	00					

5 Direct data exchange with EZ/EZD (Polled I/O Connection)

The DeviceNet master can exchange the following data with the EZ/EZD via the direct cyclic data exchange:

- Write operation
  - Setting or /resetting of the EZ/EZD inputs
  - Determination of the RUN/STOP mode.
- Read operation
  - Scanning the output states of the EZ/EZD
  - Scanning the mode of the EZ/EZD.

In order to transfer data between the slave EZ222-DN and a DeviceNet master control, you must map the respective cyclic data to the respective slave configuration.

→ The interconnection to the DeviceNet controls from Allen-Bradley is implemented using an assignment table in the RSNetWorx software tool.

→ The terms “input data” and “output data” are used relative to the point of view of the DeviceNet master.

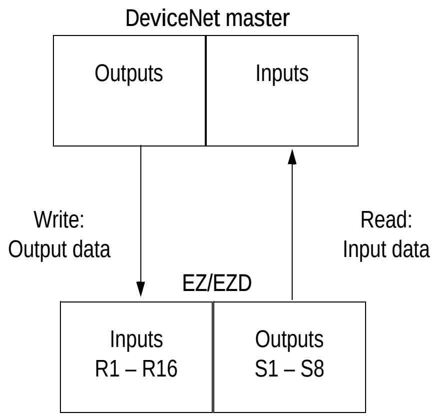


Figure 12: Input and output data relative to the DeviceNet master

Input data:Mode, S1 – S8

Attribute ID: 3
The cyclic data transfer between DeviceNet master and the EZ222-DN slave is provided by the input data byte 0, 1 and 2.

Table 7: Byte 0 to 2: input data, mode

Byte	Meaning	Value
0	Operating mode scan	→ table 8
1	Scan status of the EZ outputs S1 to S8	→ table 9
2	Not used	00_{hex}

The master reads the following data from bytes 0, 1 and 2:

Table 8: Byte 0: Operating mode

EZ/EZD identification	Bit							
	7	6	5	4	3	2	1	0 STOP/RUN
without input delay	0	0	0	1	0	0	0	0/1
with input delay	0	0	1	0	0	0	0	0/1

0 = status "0“ 1 = status "1"

Example:
Value 21_{hex} = $0010\,0001_{bin}$:
EZ/EZD is in RUN mode and operates with input delay

Table 9: Byte 1: Status of the EZ/EZD outputs S1 to S8

EZ/EZD	Bit							
	7	6	5	4	3	2	1	0
S1								0/1
S2							0/1	
S3						0/1		
S4					0/1			
S5				0/1				
S6			0/1					
S7		0/1						
S8	0/1							

0 = status "0“ 1 = status "1"

Example:
Value 19_{hex} = 00011001_{bin}:
S5, S4 and S1 are active

Byte 2: not used

→ If control commands and I/O data are used at the same time:

- The inputs will retain their previous state until this control command has been executed.
- The input bytes will be updated again after the data exchange control command has been terminated.

If the status value of the coupling module is invalid (= 04_{hex}), then byte 1 (data byte) is transferred with the value 00_{hex} to the communication bus.

Output data:mode, R1 – R16

Attribute ID: 4

The cyclic data transfer between DeviceNet master and the EZ222-DN slave is provided by the output data byte 0, 1 and 2.

Table 10: Byte 0 to 2: output data, mode

Byte	Meaning	Value
0	Determine mode	→ table 11
1	Setting/resetting of the EZ/EZD inputs R9 to R16	→ table 12
2	Setting/resetting of the EZ/EZD inputs R1 to R8	→ table 13

The master writes the following data to the bytes 0, 1 and 2:

Table 11: Byte 0: Operating mode

EZ/EZD operating mode	Bit							
	7	6	5	4	3	2	1	0
Index for setting the basic unit to safety state	0	0	0	0	0	0	0	0
Index for transferring valid data	0	0	0	1	0	1	0	0
RUN command	0	0	1	1	0	1	0	0
STOP command	0	1	0	0	0	1	0	0

0 = status "0“ 1 = status "1"

Explanation:

Value 14_{hex} = 0001 0100_{bin}:
Byte 0 must always contain this value if data are to be written to the EZ/EZD basic unit via the EZ222-DN gateway.

Value 34_{hex} = 0011 0100_{bin}:
This value sets the EZ/EZD status from STOP to RUN. It is only interpreted as command and therefore does not permit an additional transfer of data. The index value 14_{hex} must be used in this situation.

Value 44_{hex} = 01000100_{bin}:
This value sets the EZ/EZD status from RUN to STOP. It is also used only as command and is therefore based on the same operating principle as the RUN command.

Value 00_{hex} = 00000000_{bin}:
If this value is written to the control byte, the gateway overwrites the R data with zero. This function is of interest only if a master is to be set to STOP mode and as resultant measure transfers zero values to all I/O in order to ensure safety state.

→ Even if the I/O of a control relay can be assigned directly to a specific memory area of the master PLC, it is nonetheless important to conform with the correct data structure format (e.g.: input data byte 0 = 14_{hex}).

Table 12: Byte 1: Setting/resetting of the EZ/EZD inputs R9 to R16

EZ/EZD	Bit							
	7	6	5	4	3	2	1	0
R9								0/1
R10							0/1	
R11						0/1		
R12					0/1			
R13				0/1				
R14			0/1					
R15		0/1						
R16	0/1							

0 = status "0“ 1 = status "1"

Example:
Value 19_{hex} = $0001\,1001_{bin}$:
Enable R13, R12 and R9.

Table 13: Byte 2: Setting/resetting of the EZ/EZD inputs R1 to R8

EZ/EZD input	Bit							
	7	6	5	4	3	2	1	0
R1								0/1
R2							0/1	
R3						0/1		
R4					0/1			
R5				0/1				
R6			0/1					
R7		0/1						
R8	0/1							

0 = status "0“ 1 = status "1"

Example:
Value $2B_{hex}$ = 0010 1011_{bin}:
Enables R6, R4, R2 and R1.

→ If control commands and I/O data are used at the same time:

- The inputs will retain their previous state until this control command has been executed.
- The input bytes will be updated after the data exchange control command has been executed.

6 Control commands for EZ700

Control commands can be used to initiate data exchange for special services:

- Read/write date and time (page 53)
- Read image data (page 57)
- Read/write function block data (page 79).

The master PLC in this case falls back upon the message transfer protocol of the explicit messages. All parameters are addressed via the Service Code 32_{hex}. The assigned attribute ID is here used to distinguish between different parameters.

Service code	Object address	
	Class ID	Instance ID
32_{hex}	64_{hex}	01_{hex}

Note!
The I/O data retain their previously defined state while a control command is being executed. The I/O data will not be updated until data exchange for the control command has been terminated.

Caution!
You may use only the values specified for the instruction code.
Verify data to be transferred in order to avoid unnecessary errors.

A data exchange procedure is required in order to ensure the safe exchange of data via DeviceNet from master to slave and vice versa.

→ The operating mode of the basic unit must correspond with the status indicated at the LEDs when the various parameters are being set.

The master transmits a control command to initiate data exchange between the communication partners. The slave always returns an answer to this request, which indicates whether data has been exchanged or not. An error code will be returned if data exchange has failed. This code is precisely defined in the ODVA specifications → section “References”, page 5.

Read/write date and time

→ Please also note the relevant description of the real-time clock provided in the EZ500/700 manual (MN05013003E).

Telegram structure

Byte m	Byte s	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	93	–
		Write	B3	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	05	05
1	2	Index	0 – 2[1]	0 – 2[1]
2 – 6	3 – 7	Data 1 – 5	depending on index, → table 14	depending on index, → table 14

1) 0 = Time/date, → table 14
1 = Summer time, → table 15
2 = Winter time, → table 16

m = Master
s = slave

Table 14: Index 0 – date and time of real-time clock

Byte		Content	Operand		Value (hex)
Master	Slave				
2	3	Data 1	Hour	0 to 23	0x00 to 0x17h
3	4	Data 2	Minute	0 to 59	0x00 to 0x3Bh
4	5	Data 3	Day	Day (1 to 28; 29, 30, 31; depending on month and year)	0x01 to 0x1Fh
5	6	Data 4	Month	1 to 12	0x01 to 0x0Ch
6	7	Data 5	Year	0 to 99 (corresponds to 2000-2099)	0x00 to 0x63h

Table 15: Index 1 – Summer time

Byte		Content		Value (hex)
Master	Slave			
2	3	Data 1	Area	
			None	00
			Rule	01
			Automatic EU	02
			Automatic GB	03
			Automatic US	04
for “Area” = “Rule”:				
3	4	Data 2	Summer time switching rule	→ table 17
4	5	Data 3		
5	6	Data 4		
6	7	Data 5		

Table 16: Index 2 – Winter time (only valid if Area = “Rule” selected)

Byte		Content		Value (hex)
Master	Slave			
2	3	Data 1	Area = Rule	01
3 – 6	4 – 7	Data 2 – 5	Winter time switching rule	→ table 17

Switching rule bit array

→ Please also read the detailed description in the EZ500/700 manual (MN05013003E).

The following table shows the composition of the corresponding data bytes.

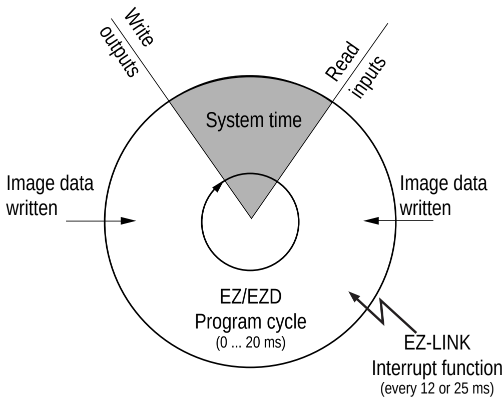
Table 17: Switching rule bit array

<table>
<tr><td></td><td colspan="8">Data 5</td><td colspan="8">Data 4</td><td colspan="8">Data 3</td><td colspan="8">Data 2</td></tr>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td></td><td colspan="3">Difference</td><td colspan="12">Time of time change</td><td colspan="4">Month</td><td colspan="5">Day</td><td colspan="2">Rule_2</td><td colspan="3">Day</td><td colspan="3">Rule_1</td></tr>
<tr><td></td><td colspan="3">0: 0:30h</td><td colspan="7">Minute: 0 to 59</td><td colspan="5">Hour: 0 to 23</td><td colspan="4">0 to 11</td><td colspan="5">0 to 30</td><td colspan="2">0: month</td><td colspan="3">0: Su</td><td colspan="3">0: on</td></tr>
<tr><td></td><td colspan="3">1: 1:00h</td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2">1: after</td><td colspan="3">1: Mo</td><td colspan="3">1: on the first</td></tr>
<tr><td></td><td colspan="3">2: 1:30h</td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2">2: before</td><td colspan="3">2: Tu</td><td colspan="3">2: on the second</td></tr>
<tr><td></td><td colspan="3">3: 2:00h</td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2"></td><td colspan="3">3: We</td><td colspan="3">3: on the third</td></tr>
<tr><td></td><td colspan="3">4: 2:30h</td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2"></td><td colspan="3">4: Thu</td><td colspan="3">4: on the fourth</td></tr>
<tr><td></td><td colspan="3">5: 3:00h</td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2"></td><td colspan="3">5: Fr</td><td colspan="3">5: on the last</td></tr>
<tr><td></td><td colspan="3"></td><td colspan="7"></td><td colspan="5"></td><td colspan="4"></td><td colspan="5"></td><td colspan="2"></td><td colspan="3">6: Sa</td><td colspan="3"></td></tr>
</table>

Read image data

→ Please also observe the relevant description of possible image data provided in the EZ500/700 manual (MN05013003E) or in the EZSoft Help.

General notes on working with image data



When writing to image data, it must be remembered that an image (e.g. inputs, outputs,...) used in the EZ/EZD program is also written cyclically by the actual program. The only image data that is unchanged is the data that is not used in the program and is therefore not overwritten in the program cycle. This operating principle also means that an image written via EZ-LINK, such as output data is only then output at the physical outputs of the EZ/EZD when the control relay is in Run mode.

Overview

Operands	Meaning	Read/Write	Type (hex)	Page
A1 – A16	Analog value comparators/threshold comparators: A1 – A16	read	8B	59
C1 – C16	Counters: C1 – C16	read	EE	60
D1 – D16	Text function blocks: D1 – D16	read	94	61
I1 – I16	Local inputs: I1 – I16	read	84	62
IA1 – IA4	Local analog inputs: IA1 – IA4	read	8C	63
M1 – M16, N1 – N16	Write marker: M1 – M16/N1 – N16	write	86/87	65
M1 – M16, N1 – N16	Read marker: M1 – M16/N1 – N16	read	86/87	67
O1 – O4	Operating hours counters: O1 – O4	read	EF	69
P1 – P4	Local P buttons: P1 – P4	read	8A	70
Q1 – Q8	Local outputs: Q1 – Q8	read	85	72
R1 – R16/ S1 – S8	Inputs/outputs of EZ-LINK: R1 – R16/S1 – S8	read	88/89	73
T1 – T16	Timers: T1 – T16	read	ED	75
Y1 – Y4	Year time switch: Y1 – Y8	read	91	76
Z1 – Z3	Master reset: Z1 – Z3	read	93	77
H1 – H4	7-day time switch: 🕒1 – 🕒8	read	90	78

Analog value comparators/threshold comparators: A1 – A16

The following commands are used to read the logic state of the individual analog value comparators A1 to A16.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	8B	8B
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 18
4	5	Data 2 (Low Byte)	00	→ table 18
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 18: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1	Bit 7	6	5	4	3	2	1	0
A1								0/1
A2							0/1	
...				...				
A8	0/1							
Data 2	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
A9								0/1
A10							0/1	
...				...				
A16	0/1							

Counters: C1 – C16

The following commands are used to read the logic state of the individual counters C1 – C16.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	EE	EE
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 19
4	5	Data 2 (Low Byte)	00	→ table 19
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 19: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1	Bit	7	6	5	4	3	2	1	0
C1									0/1
C2								0/1	
...					...				
C8		0/1							
Data 2	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
C9									0/1
C10								0/1	
...					...				
C16		0/1							

Text function blocks: D1 – D16

The following commands are used to read the logic state of the individual text function blocks (D markers).

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	94	94
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 20
4	5	Data 2 (High Byte)	00	→ table 20
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 20: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1	Bit 7	6	5	4	3	2	1	0
D1								0/1
D2							0/1	
...				...				
D8	0/1							
Data 2	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
D9								0/1
D10							0/1	
...				...				
D16	0/1							

Local inputs: I1 – I16

This command string enables you to read the local inputs of the EZ700 basic unit. The relevant input word is stored in Intel format.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1)]
0	1	Len	02	02
1	2	Type	84	84
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 21
4	5	Data 2 (High Byte)	00	→ table 21
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 21: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1	Bit 7	6	5	4	3	2	1	0
I1								0/1
I2							0/1	
..				..				
I8	0/1							
Data 2	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I9								0/1
I10							0/1	
..				..				
I16	0/1							

Local analog inputs: IA1 – IA4

The analog inputs on the EZ700 basic unit (I7, I8, I11, I12) can be read directly via DeviceNet. The 16-bit value is transferred in Intel format (Low Byte first).

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	02	02
1	2	Type	8C	8C
2	3	Index	00 – 03[2]	00 – 03[2]
3	4	Data 1 (Low Byte)	00	→ table 22
4	5	Data 2 (High Byte)	00	→ table 22
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

2) 00 = Analog input I7
01 = Analog input I8
02 = Analog input I11
03 = Analog input I12

Example:
A voltage signal is present at analog input 1. The required telegrams for reading the analog value are as follows:

Table 22: Example telegram for reading the value at the analog input

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response: read successful	–	C2
0	1	Len	02	02
1	2	Type	8C	8C
2	3	Index	02[1]	02[1]
3	4	Data 1	00	4B
4	5	Data 2	00	03
5	6	Data 3	00	00
6	7	Data 4	00	00

1) 02 = Analog input I11

Byte 4 – Data 1 (Low Byte): $4B_{hex}$
Byte 5 – Data 2 (High Byte): 03_{hex}
➔ corresponding 16-bit value: $034B_{hex}$ = 843

The value 843 corresponds to the 10 bit value of the analog converter. The following conversion is required for the actual analog value:

$$\frac{10\ \text{V}}{1023} \times 10\ \text{bit} \quad \Rightarrow \quad \frac{10\ \text{V}}{1023} \times 843 = 8.24\ \text{V}$$

Write marker: M1 – M16/N1 – N16

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Write	8C	–
	0	Response:		
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type[2]		
		With M marker	86	86
		With N marker	87	87
2	3	Index[2]	00 – 0F	00 – 0F
3	4	Data 1 (Low Byte)[3]	00/01	00/01
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

2) There are 16 M markers and 16 N markers. The markers are addressed by Type and Index:
Use Type to select the M or N marker.
Use Index to select the marker number.

3) The marker is set if a value is written to the data byte that does not equal zero. The marker is reset accordingly if the value 0 is written to data byte Data 1.

Example:
Marker M13 is set.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Write	8C	–
	0	Response:		
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type		
		M marker	86	86
2	3	Index	0C	0C
3	4	Data 1	01	00
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Read marker: M1 – M16/N1 – N16

Unlike the write operation, the marker read operation reads the entire marker area of a particular marker type (M or N) is read.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type		
		M marker	86	86
		N marker	87	87
2	3	Index[2]	00	00
3	4	Data 1 (Low Byte)	00	→ table 23
4	5	Data 2 (Low Byte)	00	→ table 23
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

2) There are 16 M markers and 16 N markers. The markers are addressed by Type and Index:
Use Type to select the M or N marker.
Use Index to select the marker number.

Table 23: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1		Bit	7	6	5	4	3	2	1	0
m	N									
M1	N1									0/1
M2	N2								0/1	
...	...					...				
M8	N8		0/1							
Data 2		Bit	7	6	5	4	3	2	1	0
M9	N9									0/1
M10	N10								0/1	
...	–					...				
M16	N16		0/1							

Example: The N markers are read:

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type		
		N marker	87	87
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	04
4	5	Data 2 (Low Byte)	00	84
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

The markers N3, N11 and N16 are set.

Operating hours counters: O1 – O4

The following commands are used to read the logic state of the operating hours counters O1 – O4.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	EF	EF
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 24
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 24: Byte 3 (master) or byte 4 (slave): Data 1

Data 1	Bit 7	6	5	4	3	2	1	0
O1								0/1
O2							0/1	
O3						0/1		
O4					0/1			
...	...	...	...	...				

Local P buttons: P1 – P4

The local P buttons are the display cursor buttons of the EZ700 basic unit. You can scan the buttons in both RUN and STOP mode.

→ Ensure that the P buttons are also activated via the System menu (in the basic unit).

Only one byte has to be transferred for the P buttons.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	8A	8A
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 25
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 25: Byte 3 (master) or byte 4 (slave): Data 1

Data 1	Bit 7	6	5	4	3	2	1	0
P1								0/1
P2							0/1	
P3						0/1		
P4					0/1			
–				0				
–			0					
–		0						
–	0							

Example:
Data 1 = 2_{hex} → P3 is active.

Local outputs: Q1 – Q8

The local outputs can be read directly via the DeviceNet fieldbus.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1)]
0	1	Len	01	01
1	2	Type	85	85
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 26
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 26: Byte 4: Data 1

Data 1	Bit	7	6	5	4	3	2	1	0
Q1									0/1
Q2								0/1	
..					..				
Q8		0/1							

Example:
Data 1 = 52_{hex} → Q2, Q5 and Q7 are active.

Inputs/outputs of EZ-LINK: R1 – R16/S1 – S8

This service allows you to read the local R and S data and the data of the NET stations (1 – 8) transferred via EZ-LINK, again from the relevant EZ700 image.

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1)]
0	1	Len	01	01
1	2	Type		
		for R data	88	88
		for S data	89	89
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 27
4	5	Data 2 (Low Byte)	00	→ table 27
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 27: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1		Bit	7	6	5	4	3	2	1	0
RW	SW									
R1	S1									0/1
R2	S2								0/1	
...	...					...				
R8	S8		0/1							
Data 2		**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R9	–									0/1
R10	–								0/1	
...	–					...				
R16	–		0/1							

Timers: T1 – T16

The following commands are used to read the logic state of the individual timers T1 - T16.

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	ED	ED
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 28
4	5	Data 2 (Low Byte)	00	→ table 28
5 – 6	6 – 7	Data 3 – 4	00	00

1) Possible causes → page 97

Table 28: Byte 3 to 4 (master) or Byte 4 to 5 (slave): Data 1 to 2

Data 1	Bit 7	6	5	4	3	2	1	0
T1								0/1
T2							0/1	
...				...				
T8	0/1							
Data 2	**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
T9								0/1
T10							0/1	
...				...				
T16	0/1							

Year time switch: Y1 – Y8

The following commands are used to read the logic state of the individual year time switches.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	91	91
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 29
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 29: Byte 3 (master) or byte 4 (slave): Data 1

Data 1	Bit 7	6	5	4	3	2	1	0
HY1								0/1
HY2							0/1	
HY3						0/1		
HY4					0/1			
HY5				0				
HY6			0					
HY7		0						
HY8	0							

Example:
Data 1 = 1_{hex} → HY2 is active

Master reset: Z1 – Z3

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	93	93
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 30
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 30: Byte 3 (master) or byte 4 (slave): Data 1

Data 1	Bit 7	6	5	4	3	2	1	0
Z1 for Q outputs								0/1
Z2 for M markers							0/1	
Z3 for outputs and markers						0/1		
...	0	0	0	0	0			

7-day time switch: Ö1 – Ö8

The following commands are used to read the logic state of the individual 7-day time switches.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	88	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0[1]
0	1	Len	01	01
1	2	Type	90	90
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 31
4 – 6	5 – 7	Data 2 – 4	00	00

1) Possible causes → page 97

Table 31: Byte 3 (master) or byte 4 (slave): Data 1

Data 1	Bit 7	6	5	4	3	2	1	0
HW1								0/1
HW2							0/1	
HW3						0/1		
HW4					0/1			
HW5				0				
HW6			0					
HW7		0						
HW8	0							

Example:
Data 1 = 2_{hex} → Ö3 is active.

Read/write function block data

→ Please also observe the relevant description of the function blocks provided in the EZ500/700 manual (MN05013003E) or in the EZSoft Help.

General notes

Always note the following when working with function blocks:

- The relevant data is transferred in Intel format. In other words, the first byte is the low byte (Byte 5) and the last byte (byte 8) the high byte.
- The maximum data length is 4 bytes. All values must be transferred in hexadecimal format.

Overview

Operands	Meaning	Read/Write	Type (hex)	Page
A1 – A16	Analog value comparator/threshold comparator: A1 – A16	Read/Write	8D	80
C1 – C16	Counter relays: C1 – C16	Read/Write	8F	83
O1 – O4	Operating hours counters: O1 – O4	Read/Write	92	86
T1 – T16	Timing relays: T1 – T16	Read/Write	8E	88
Y1 – Y8	Year time switch: Y1 – Y8	Read/Write	A2	91
⌚1 – ⌚8	7-day time switch: ⌚1 – ⌚8	Read/Write	A1	94

Analog value comparator/threshold comparator: A1 – A16

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	8D	8D
1	2	Instance[2]	00 – 0F	00 – 0F
2	3	Index	→ table 32	→ table 32
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 33	depending on index, → table 33

1) Possible causes → page 97

2) EZ provides 16 analog comparators A1 to A16 for use as required. These can be addressed using the instance (0 – F).

Table 32: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 33		×	
01	Control byte → table 34		×	
02	Comparison value 1	I1[2]	×	c[1]
03	Comparison value 2	I2[2]	×	c[1]
04	Gain factor for I1 (I1 = F1 × I1)	F1[2]	×	c[1]
05	Gain factor for I2 (I2 = F2 × I2)	F2[2]	×	c[1]
06	Offset for value I1 (I1 = OS + actual value at I1)	OS[2]	×	c[1]
07	Switching hysteresis for value I2	HY[2]	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

2) A 16-bit value is transferred in data bytes Data 1 – Data 2. It should be remembered that the low byte 1 is in Data 1 (Byte 5) and the high byte 2 (byte 8) in Data 2.
Example: 5327_{dec} = $14CF_{hex}$ → Data 1 = 0xCF, Data 2 = 0x14

Table 33: Index 00 – Parameters

Meaning	Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Appears in the parameter menu																	
Yes/no																	0/1
Compare																	
FB not used														0	0	0	
EQ (=)														0	0	1	
GE (≧)														0	1	0	
LE (≦)														0	1	1	
GT (>)														1	0	0	
LT (<)														1	0	1	
Use as constant and therefore can be written to																	
I1= Constant													0/1				
F1= Constant												0/1					
I2= Constant											0/1						
F2 = Constant										0/1							
OS = Constant									0/1								
HY = Constant								0/1									
Not used		0	0	0	0	0	0										

Example:
Data 1 (Byte 4) = 0xA3, Data 2 (Byte 5) = 0x03
→ Resulting 16-bit value = $03A3_{hex}$

Meaning: HY, OS, F2, F1 are assigned a constant; I1, I2 are assigned to a variable such as I7, I8 C2...etc., appears in the Parameter menu;

The output of the analog value comparator is active for as long as the comparison (I1 × F1) + OS = (I2 × F2) + HY is fulfilled.

Table 34: Index 01 – Control byte

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	Q1[1]

1) Status 1 if comparison condition is fulfilled.

Counter relays: C1 – C16

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	8F	8F
1	2	Instance[2]	00 – 0F	00 – 0F
2	3	Index	→ table 35	→ table 35
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 36	depending on index, → table 36

1) Possible causes → page 97

2) EZ provides 16 counters C1 to C16 for use as required. These can be addressed using the instance (0 – F).

Table 35: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 36		×	
01	Control byte → table 37		×	
02	Process variable	S1[2]	×	c[1]
03	Counter setpoint 2	S2[2]	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

2) A 16-bit value is transferred in data bytes Data 1 – Data 2. It should be remembered that Data 1 is the low byte and Data 2 the high byte.

Table 36: Index 00 – Parameters

Meaning	Bit	7	6	5	4	3	2	1	0
Appears in the parameter menu									
Yes/no									0/1
Counter mode									
FB not used							0	0	
Up/down counter (N)							0	1	
High-speed up/down counter (H)							1	0	
Frequency counter (F)							1	1	
Use as constant and therefore can be written to									
Counter setpoint S1						0/1			
Unused bits		–	–	–	–				

Example:
Data 1 (Byte 4) = 0x07

Meaning:
The values appear in the Parameter menu. The counter is used in the mode of the frequency meter. The counter setpoint 1 is not assigned to a constant and cannot therefore be written to.

Table 37: Index 01 – Control byte

Data 1	Bit	7	6	5	4	3	2	1	0
FB output		–	–	–	–	C[4]	RE[3]	D[2]	Q1[1]

1) Switch contact
2) Count direction: 0 = up counting, 1 = down counting
3) Reset, the timing relay is reset (reset coil)
4) Count coil, counts on every rising edge

Example:
the actual value of C3 is to be read:

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Command: Read	89	–
	0	Response: read successful	–	C2
0	1	Type	8F	8F
1	2	Instance	02	02
2	3	Index	02	02
3	4	Data1	00	12
4	5	Data 2	00	03
5	6	Data 3	00	00
6	7	Data 4	00	00

Explanation:

Data 1 = 12
Data 2 = 03
→ resulting 16-bit value = 0312_{hex} = 786_{dec}

Counter status = 786

Operating hours counters: O1 – O4

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	92	92
1	2	Instance[2]	00 – 03	00 – 03
2	3	Index	→ table 38	→ table 38
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 39	depending on index, → table 39

1) Possible causes → page 97

2) EZ provides 4 operating hours counters O1 to O4. These can be addressed using the instance (0 – 3).

Table 38: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 39		×	
01	Control byte → table 40		×	
02	Process variable	S1[2]	×	c[1]
03	Counter setpoint 2	S2[2]	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

2) A 32-bit value is transferred in data bytes Data 1 – Data 4. It should be remembered that the Data 1 is the low byte and Data 4 the high byte.

Table 39: Index 00 – Parameters

Meaning	Bit	7	6	5	4	3	2	1	0
Appears in the parameter menu									
Yes/no									0/1
Use in the program									
Setpoint S1								0/1	
Unused bits		–	–	–	–	–	–		

Example:
Data 1 (Byte 4) = 0x01

Meaning:
The values appear in the Parameter menu.

Table 40: Index 01 – Control byte

Data 1	Bit	7	6	5	4	3	2	1	0
FB output		–	–	–	–	–	RE3)	EN2)	Q11)

1) Switch contact
2) Enable, the timing relay is started (trigger coil)
3) Reset, the timing relay is reset (reset coil)

Example:
Index 02/03

Transferred values: Data 1 0x21
Data 2 0x23
Data 3 0x40
Data 4 0x00

Resulting value: $00402321_{hex} = 4203297_{dec}$

Timing relays: T1 – T16

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	8E	8E
1	2	Instance[2]	00 – 0F	00 – 0F
2	3	Index	→ table 41	→ table 41
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 42	depending on index, → table 42

1) Possible causes → page 97
2) EZ provides 16 timing relays T1 to T16 for use as required. These can be addressed using the instance (0 – F).

Table 41: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 42		×	
01	Control byte → table 43		×	
02	Actual value 1	T	×	c[1]
03	Time setpoint 1	S1[2]	×	c[1]
04	Time setpoint 2	S2[2]	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.
2) A 16-bit value is transferred in data bytes Data 1 – Data 2. It should be remembered that Data 1 is the low byte and Data 2 the high byte.

Table 42: Index 00 – Parameters

Meaning	Bit	7	6	5	4	3	2	1	0
Appears in the parameter menu									
Yes/no									0/1
Timer mode									
On-delayed,						0	0	0	
Off-delayed.						0	0	1	
On-delayed with random setpoint						0	1	0	
Off-delayed with random setpoint						0	1	1	
On and off delayed (two time setpoints)						1	0	0	
On and off delayed each with random setpoint (two time setpoints)						1	0	1	
Impulse transmitter						1	1	0	
Flashing relay (two time setpoints)						1	1	1	
Timebase									
FB not used				0	0				
Millisecond: S				0	1				
Second: M:S				1	0				
Minute: H:M				1	1				
Use as constant and therefore can be written to									
Time setpoint S1			0/1						
Time setpoint S2		0/1							

Example:
Data 1 (Byte 4) = 0xAC

Meaning:
The values appear in the Parameter menu. The time is used in the impulse transmitter mode with the Second time base. The time setpoint S1 is assigned a constant and the time setpoint S2 is assigned a variable such as I7, I8 C2...etc.

Table 43: Index 01 – Control byte

	Bit	7	6	5	4	3	2	1	0
FB input/output Data 3		–	–	–	–	ST[4]	RE[3]	EN[2]	Q1[1]

1) Switch contact
2) Enable, the timing relay is started (trigger coil)
3) Reset, the timing relay is reset (reset coil)
4) Stop, the timing relay is stopped (Stop coil)

Example:
The time setpoint 1 is to be read:

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Command: Read	89	–
	Response: read successful	–	C2
1	Type	8E	8E
2	Instance	00	00
3	Index	03	03
4	Data1	00	4C
5	Data 2	00	06
6	Data 3	00	00
7	Data 4	00	00

Explanation:

Data 1 = 4C
Data 2 = 06
→ resulting 16-bit value = $064C_{hex}$ = 1612_{dec}

Meaning depending on set time base:

millisecond	s	16120 ms	16.120 s
Seconds	M:S	1620 s	26:52 Minutes
Minute	H:M	1612 min	67:04 Hours

Year time switch: Y1 – Y8

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	A2	A2
1	2	Instance[2]	00 – 07	00 – 07
2	3	Index	→ table 44	→ table 44
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 45	depending on index, → table 45

1) Possible causes → page 97

2) EZ provides 8 year time switches Y1 to Y8 for use as required. These can be addressed using the instance (0 – 7).

Table 44: Operand overview

Index (hex)	Operand	Read	Write
00	Parameters → table 45	×	
01	Control byte → table 46	×	
	Channel A	×	c[1]
11	Time point ON	×	c[1]
12	Time point OFF	×	c[1]
	Channel B	×	c[1]
21	Time point ON	×	c[1]
22	Time point OFF	×	c[1]
	Channel C	×	c[1]
31	Time point ON	×	c[1]
32	Time point OFF	×	c[1]
	Channel D	×	c[1]
41	Time point ON	×	c[1]
42	Time point OFF	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

Note: In the data bytes Data 1 – Data 3 the switching points are transferred.

Table 45: Index 00 – Parameters

Meaning	Bit	7	6	5	4	3	2	1	0
Appears in the parameter menu									
Channel A									0/1
Channel B								0/1	
Channel C							0/1		
Channel D						0/1			
Unused bits		–	–	–	–				

Example:
Data 1 (Byte 4) = 0x03 l The values for the year time switch of channels A and B appear in the parameter menu.

Table 46: Index 01 – Control byte

Data 1	Bit	7	6	5	4	3	2	1	0
FB output		–	–	–	–	–	–	–	Q1[1]

1) Status 1 if count condition is fulfilled.

Channel A, index 11/12
Index 0x11 channel A timepoint of switch on
Index 0x12 channel A timepoint of switch off
Data 1 (Byte 4) – day
Data 2 (Byte 5) – month
Data 3 (Byte 6) – year

Example:
The year time switch channel A should be switched on at the 21.04.2004.

Index = 0x11
Data 1 = 0x15
Data 2 = 0x04
Data 3 = 0x04

The year time switch channel B should be switched off on the 05.11.2012.

Index = 0x22
Data 1 = 0x05
Data 2 = 0x0B
Data 3 = 0x0C

7-day time switch: ⏲1 – ⏲8

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	89	–
		Write	8D	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0[1]
0	1	Type	A1	A1
1	2	Instance[2]	00 – 07	00 – 07
2	3	Index	→ table 47	→ table 47
3 – 6	4 – 7	Data 1 – 4	depending on index, → table 48	depending on index, → table 48

1) Possible causes → page 97

2) EZ provides 8 week time switches ⏲1 to ⏲8 for use as required. These can be addressed using the instance (0 – 7).

Table 47: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 48		×	
01	Control byte → table 49		×	
11	Channel A	Day on/off	×	c[1]
12		Time on	×	c[1]
13		Time off	×	c[1]
21	Channel B	Day on/off	×	c[1]
22		Time on	×	c[1]
23		Time off	×	c[1]
31	Channel C	Day on/off	×	c[1]
32		Time on	×	c[1]
33		Time off	×	c[1]
41	Channel D	Day on/off	×	c[1]
42		Time on	×	c[1]
43		Time off	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

Note: A 16-bit value is transferred in data bytes Data 1 – Data 4. It should be remembered that Data 1 is the low byte and Data 2 the high byte.

Table 48: Index 00 – Parameters

Meaning	Bit	7	6	5	4	3	2	1	0
Appears in the parameter menu									
Channel A									0/1
Channel B								0/1	
Channel C							0/1		
Channel D						0/1			
Unused bits		–	–	–	–				

Example:
Data 1 (Byte 4) = 0x03

Meaning:
The values of the 7-day timer switch WH.. of channel A and B appear in the parameter menu.

Table 49: Index 01 – Control byte

Data 1	Bit	7	6	5	4	3	2	1	0
FB output		–	–	–	–	–	–	–	Q1[1]

1) Status 1 if count condition is fulfilled.

Channel A, index 11/12/13
Index 0x11 channel A day on/off
Data 1 (Byte 4) – day on
Data 2 (Byte 5) – day off
0x01 = Sunday ... 0x07 = Saturday

If the channel is not used the 16 bit value is equal to 0x00.

Index 0x12 – time on (2 bytes)
Index 0x13 – time off (2 bytes)
Data 1 (Byte 4) – hour
Data 2 (Byte 5) – minute

Example: time on at 13:43
Data 1 = 0x0D
Data 2 = 0x2B

Analysis – error codes via EZ-LINK

The EZ700 basic unit will return a defined error code in the event of an incorrectly selected operating mode or an invalid telegram. The error code transferred has the following structure:

Telegram structure

Byte	Meaning	Slave transmits (value hex)
0	Answer	
	Command rejected	C0
1	Type	00
2	Instance	00
3	Index	00
4	Error code	→ table 50

Table 50: Error codes

Error code	Description
0x01	An unknown telegram has been sent.
0x02	An unknown object has been sent.
0x03	An unknown command has been sent.
0x04	An invalid instance has been sent.
0x05	An invalid parameter set has been used.
0x06	An attempt has been made to write a variable which is not a constant.
0x0C	The device is in an invalid device mode. STOP → RUN or RUN → STOP
0x0D	An invalid display access occurs. Please exit the menu level to allow the status display to be shown on the display. Writing to the clock is not possible.
0xF0	An attempt has been made to control an unknown parameter.
0xF1	Invalid value

7 EZ800/EZD Control Commands

Control commands can be used to initiate data exchange for special services:

- Read/write date and time (page 101)
- Read/write image data (page 107)
- Read/write function block data (page 127)

The master PLC in this case falls back upon the message transfer protocol of the explicit messages. All parameters are addressed via the Service Code 32_{hex}. The assigned attribute ID is here used to distinguish between different parameters.

Service code	Object address	
	Class ID	Instance ID
32_{hex}	64_{hex}	01_{hex}

Note!
The I/O data retain their previously defined state while a control command is being executed. The I/O data will not be updated until data exchange for the control command has been terminated.

Caution!
You may use only the values specified for the instruction code.
Verify data to be transferred in order to avoid unnecessary errors.

A data exchange procedure is required in order to ensure the safe exchange of data via DeviceNet from master to slave and vice versa.

→ The operating mode of the basic unit must correspond with the status indicated at the LEDs when the various parameters are being set.

The master transmits a control command to initiate data exchange between the communication partners. The slave always returns an answer to this request, which indicates whether data has been exchanged or not. An error code will be returned if data exchange has failed. This code is precisely defined in the ODVA specifications.

Read/write date and time

→ Please also note the relevant description of the real-time clock provided in the EZ800 manual (MN05013004E) and the EZD manual (MN05013005E).

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	93	–
		Write	B3	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	05	05
1	2	Index	00	00
2 – 6	3 – 7	Data 1 – 5		
		Read operation	00	→ table 51
		Write operation	→ table 51	00

Table 51: Byte 2 to 6 (master) or Byte 3 to 7 (slave): Data 1 to 5

Byte		Content	Operand		Value (hex)
Master	Slave				
2	3	Data 1	Hour	0 to 23	00 – 17
3	4	Data 2	Minute	0 to 59	00 – 3B
4	5	Data 3	Day	Day (1 to 28; 29, 30, 31; depending on month and year)	01 – 1F
5	6	Data 4	Month	1 to 12	01 – 0C
6	7	Data 5	Year	0 to 99 (corresponds to 2000-2099)	00 – 63

Winter/summer time, DST

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	93	–
		Write	B3	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	05	05
1	2	Index		
		01: Summer/Winter time	→ table 52	→ table 52
		02: Winter time (to the “Area” = rule”)[1]	→ table 53	→ table 53
2 – 6	3 – 7	Data 1 – 5		
		Read operation	00	depending on index, → table 52, 53
		Write operation	depending on index, → table 52, 53	00

1) Detailed setting possibilities for EZ800/EZD

Table 52: Index 01 – Summer/Winter time switchover

Byte		Content		Value (hex)
Master	Slave			
2	3	Data 1	Area	
			None	00
			Manual	01
			Automatic EU	02
			Automatic GB	03
			Automatic US	04
			Rule[1]	05
for "Area" = "manual":				
3	4	Data 2	Set summer time day (1 to 28, 29, 30, 31 depending on month and year)	00 – 3B
4	5	Data 3	Set Summer time month (1 to 12)	01 – 1F
5	6	Data 4	Set winter time day (1 to 28, 29, 30, 31 depending on month and year)	01 – 0C
6	7	Data 5	Set Winter time month (1 to 12)	00 – 63
for "Area" = "Rule"[1]:				
3 – 6	4 – 7	Data 2 – 5	Summer time switching rule	→ table 54

1) Detailed setting possibilities for EZ800/EZD

Table 53: Index 02 – Winter time (only valid if Area = "Rule" selected)

Byte		Content		Value (hex)
Master	Slave			
2	3	Data 1	Area = Rule	01
3 – 6	4 – 7	Data 2 – 5	Winter time switching rule	→ table 54

Switching rule bit array

→ Please also read the detailed description in the EZ800 manual (MN05013004E). The following table shows the composition of the corresponding data bytes.

Table 54: Switching rule bit array

	Data 5				Data 4			Data 3		Data 2	
Bit	31 30 29	28 27 26	25 24	23 22 21 20 19	18 17 16 15	14 13 12 11 10	9 8 7 6 5 4	3 2 1 0
	Rule_1	Day	Rule_2	Day	Month	Time of time change		Difference
	0: on	0: Su	0: month	0 to 30	0 to 11	Hour: 0 to 23	Minute: 0 to 59	0: 0:30h
	1: on the first	1: Mo	1: after					1: 1:00h
	2: on the second	2: Tu	2: before					2: 1:30h
	3: on the third	3: We						3: 2:00h
	4: on the fourth	4: Thu						4: 2:30h
	5: on the last	5: Fr						5: 3:00h
		6: Sa						

Example

The real-time clock of the EZ800 is to be set to Friday 23.05.2003, 14:36.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Write	B3	–
	0	Response: Write successful	–	C1
0	1	Len	05	05
1	2	Index	00	00
2	3	Data 1 (hex)	0E	00
3	4	Data 2 (minute)	24	00
4	5	Data 3 (day)	17	00
5	6	Data 4 (month)	05	00
6	7	Data 5 (year)	03	00

→ All values must be transferred as hexadecimal values.

Read/write image data

→ Please also observe the relevant description of possible image data provided in the EZ800 manual (MN05013004E) or in the EZSoft Help. The information provided in section “General notes on working with image data” on page 57 also applies to EZ800.

Overview

Operands	Meaning	Read/Write	Command (hex)	Page
IA1 – IA4	Local analog inputs: IA1 – IA4	read	02	108
ID1 – ID16	Local diagnostics: ID1 – ID16	read	03	110
IW0	Read local inputs: IW0	read	01	112
IW1 – IW8	Inputs of the network station: IW1 – IW8	read	01	114
M...	Marker: M..	read/write	0B – 0E	115
P1 – P4	Local P buttons: P1 – P4	read	06	118
QA1	Local analog output: QA1	read/write	05	120
QW0, QW1 – QW8	Local outputs: QW0/ outputs of the network station: QW1 – QW8	read/write	04	121
R1 – R16 S1 – S8	Inputs/outputs of EZ-LINK: RW/SW	read	07/09	123
RN1 – RN32 SN1 – SN32	Receive data network: RN1 – RN32/ Send data network: SN1 – SN32	read	08/0A	125

Local analog inputs: IA1 – IA4

The analog inputs on the EZ800 and EZD basic units can be read directly via DeviceNet. The 16-bit value is transferred in Intel format (Low Byte first).

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	02	02
2	3	Index	01 – 04[1]	01 – 04[1]
3	4	Data 1 (Low Byte)	00	→ example on page 109
4	5	Data 2 (High Byte)	00	
5 – 6	6 – 7	Data 3 – 4	00	00

1) 01 = Analog input I7
02 = Analog input I8
03 = Analog input I11
04 = Analog input I12

Example

A voltage signal is present at analog input 1. The required telegrams for reading the analog value are as follows:

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response: Read successful	–	C2
0	1	Len	02	02
1	2	Type	02	02
2	3	Index	01[1]	01[1]
3	4	Data 1	00	D9
4	5	Data 2	00	02
5	6	Data 3	00	00
6	7	Data 4	00	00

1) 01 = Analog input 1

Byte 4 – Data 1 (Low Byte): $D9_{hex}$

Byte 5 – Data 2 (High Byte): 02_{hex}

→ corresponding 16-bit value: $02D9_{hex}$ = 729 (7.29 V)

Local diagnostics: ID1 – ID16

The local diagnostics (ID1 – ID8) bytes indicate the status of the individual NET stations. The connection to the remote station (only EZD) is indicated via ID9.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	03	03
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 55
4	5	Data 2 (High Byte)	00	→ table 55
5 – 6	6 – 7	Data 3 – 4	00	00

Table 55: Byte 4 to 5: Data 1 to 2

Data 1	Bit	7	6	5	4	3	2	1	0
ID1									0/1
ID2								0/1	
..					..				
ID8		0/1							
Data 2	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID9									0/1
–								1	
...					...				
–		1							

0/1= active/inactive NET station, –= not assigned

Example

Data 1 = F8, Data 2 = FF → In the EZ-NET network, the three stations are present with the NET IDs 1, 2, 3

Read local inputs: IW0

This command string enables you to read the local inputs of the EZ800/EZD. The relevant input word is stored in Intel format.

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	01	01
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 56
4	5	Data 2 (High Byte)	00	→ table 56
5 – 6	6 – 7	Data 3 – 4	00	00

Table 56: Byte 4 to 5: Data 1 to 2

Data 1	Bit	7	6	5	4	3	2	1	0
I1									0/1
I2								0/1	
..					..				
I8		0/1							
Data 2	**Bit**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
I9									0/1
I10								0/1	
..					..				
I16		0/1							

Example: Read local inputs IW0

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response: Read successful	–	C2
0	1	Len	02	02
1	2	Type	01	01
2	3	Index	00	00
3	4	Data 1	00	C4
4	5	Data 2	00	02
5	6	Data 3	00	00
6	7	Data 4	00	00

→ All values must be transferred as hexadecimal values.

The values Data 1 = C4 and Data 2 = 02 indicate that the inputs I8, I7, I3 and I10 have been set to 1.

Inputs of the network station: IW1 – IW8

The EZ800 and EZD devices can be remotely expanded very simply using the EZ-NET. The service offered here makes it possible to implement read access to the inputs of individual NET stations.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	01	01
2	3	Index	01 – 08[1]	01 – 08[1]
3	4	Data 1 (Low Byte)	00	→ table 56 on page 112.
4	5	Data 2 (High Byte)	00	
5 – 6	6 – 7	Data 3 – 4	00	00

1) Corresponds to address of network station

Marker: M..

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	91	–
		Write	B1	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	→ table 57	→ table 57
1	2	Type		
2	3	Index		
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	→ Example 1: Set/reset market bit on page 117
		Write operation	→ Example 2: Write marker word on page 117	00

Table 57: Byte 0 to 2 (master) or: Byte 1 to 3 slave: Len, Type, Index

Operand				Len	Type	Index
Marker bit	M1	..	M96	01_{hex}	$0B_{hex}$	01 to 60_{hex}
Marker byte	MB1	..	MB96	01_{hex}	$0C_{hex}$	01 to 60_{hex}
Marker word	MW1	..	MW96	02_{hex}	$0D_{hex}$	01 to 60_{hex}
Marker double word	MD1	..	MD96	04_{hex}	$0E_{hex}$	01 to 60_{hex}

If required, refer to the more detailed description of the marker allocation in the EZ800 manual. Only a small extract of this manual is shown at this point in order to illustrate the allocation principle.

Note!
The function blocks and DW markers (32-bit values) of EZ800/EZD operate with signed values.

Applies to MD, MW, MB, M	Left = Most significant bit, byte, word			Right = Least significant bit, byte, word
32 bit	MD1			
16 bit	MW2		MW1	
8 bit	MB4	MB3	MB2	MB1
1 bit	M32 to M25	M24 to M17	M16 to M9	M8 to M1
32 bit	MD2			
16 bit	MW4		MW3	
8 bit	MB8	MB7	MB6	MB5
1 bit	M64 to M57	M56 to M49	M48 to M41	M40 to M33

→ The relevant marker values are transferred in Intel format. In other words, the first byte is the low byte (Byte 4) and the last byte the high byte.

Example 1: Set/reset market bit

Marker bit 62 should be set or reset. Write a “1” to set or a “0” to reset the marker bit in the least significant bit of data byte “Data 1”.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Write	B1	–
	0	Response: Write successful	–	C1
0	1	Len	01	01
1	2	Type	0B	0B
2	3	Index	3E	3E
3	4	Data 1	010[1)]	00
4 – 6	5 – 7	Data 2 – 4	00	00

1) 01 = set, 00 = reset

Example 2: Write marker word

The value 823 should be written into the marker word MW32: $823_{dec} = 337_{hex}$ → Data 1 = 37_{hex}, Data 2 = 03_{hex}

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Write	B1	–
	0	Response: Write successful	–	C1
0	1	Len	01	01
1	2	Type	0D	0D
2	3	Index	20	20
3	4	Data 1	37	00
4	5	Data 2	03	00
5	6	Data 3	00	00
6	7	Data 4	00	00

Local P buttons: P1 – P4

The local P buttons are the display cursor buttons of the EZ800/EZD basic unit. You can scan the buttons in both RUN and STOP mode.

→ Ensure that the P buttons are also activated via the SYSTEM menu (in the basic unit).

Only one byte has to be transferred for the P buttons.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	06	06
2	3	Index	00	00
3	4	Data 1 (Low Byte)	00	→ table 58
4 – 6	5 – 7	Data 2 – 4	00	00

Table 58: Byte 4: Data

Data 1	Bit	7	6	5	4	3	2	1	0
P1									0/1
P2								0/1	
P3							0/1		
P4						0/1			
–					0				
–				0					
–			0						
–		0							

Local analog output: QA1

The commands provided can be used to access the local analog output of the EZ800 or EZD basic unit. When writing to the analog output the value will only be output if the respective device is in RUN mode and if the respective image is not written by the actual program, → section “Read/write image data”on page 107.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	91	–
		Write	B1	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	05	05
2	3	Index	00	00
3 – 4	4 – 5	Data 1 – 2		
		Read operation	00	→ example
		Write operation	→ example	00
5 – 6	6 – 7	Data 3 – 4	00	00

Example:
The analog output should output a value of approx. 5 V.

500 = $01F4_{hex}$ Byte 4 – Data 1 (LowByte) : $F4_{hex}$
Byte 5 – Data 2 (HighByte): 01_{hex}

Local outputs: QW0/ outputs of the network station: QW1 – QW8

The local outputs can be read directly via DeviceNet. However, the outputs are only switched externally if the device is in Run mode and the addressed output is not being used in the circuit diagram. → section "Read/write image data" on page 107.

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	91	–
		Write	B1	–
	0	Answer		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	04	04
2	3	Index[1)]	00/01 – 08	00/01 – 08
3	4	Data 1		
		Read operation	00	→ table 55
		Write operation	→ table 59	00
4 – 6	5 – 7	Data 2 – 4	00	00

1) 00 = Local output
01 – 08 = Outputs of network stations 1 – 8

Table 59: Byte 4: Data

Data 1	Bit	7	6	5	4	3	2	1	0
Q1									0/1
Q2								0/1	
Q3							0/1		
Q4						0/1			
Q5					0				
Q6				0					
Q7			0						
Q8		0							

Inputs/outputs of EZ-LINK: RW/SW

This service allows you to read the local R and S data and the data of the NET stations (1 – 8) transferred via EZ-LINK, again from the relevant EZ800/EZD image.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
		Response:		
	0	Read successful	–	C2
		Command rejected	–	C0
0	1	Len	02	02
1	2	Type	For RW: 07	For RW: 07
2			For SW: 09	For SW: 09
	3	Index	00/01 – 08[1]	00/01 – 08[1]
3	4	Data 1 (Low Byte)	00	→ table 60
4	5	Data 2 (High Byte)	00	→ table 60
5 – 6	6 – 7	Data 3 – 4	00	00

1) 00 = Local input/output
01 – 08 = Address of network station (NET-ID 1 – 8)

Table 60: Byte 4 to 5: Data 1 to 2

Data 1		Bit 7	6	5	4	3	2	1	0
RW	SW								
R1	S1								0/1
R2	S2							0/1	
R3	S3						0/1		
R4	S4					0/1			
R5	S5				0/1				
R6	S6			0/1					
R7	S7		0/1						
R8	S8	0/1							
Data 2		**Bit 7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R9	–								0/1
R10	–							0/1	
R11	–						0/1		
R12	–					0/1			
R13	–				0/1				
R14	–			0/1					
R15	–		0/1						
R16	–	0/1							

Receive data network: RN1 – RN32/ Send data network: SN1 – SN32

EZ-NET allows a point-to-point connection to be implemented between the individual NET stations. The RN and SN data are used for the data exchange (see the EZ800 manual).

→ The RN SN data of the local device (Index = 0) to which the EZ222-DN is fitted cannot be scanned. In this case the command would be denied with the $0C_{hex}$ signal.

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	91	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Len	04	04
1	2	Type	For RN1 – RN32: 08	For RN1 – RN32: 08
			For SN1 – SN32: 0A	For SN1 – SN32: 0A
2	3	Index	01 – 08[1]	01 – 08[1]
3 – 6	4 – 7	Data 1 – 4	00	→ table 61

1) Corresponds to NET-ID

Table 61: Byte 4 to 7: Data 1 to 4

Data 1		Bit	7	6	5	4	3	2	1	0
RN1	SN1					...				0/1
...									0/1	
RN8	SN8		0/1							
Data 2		Bit	7	6	5	4	3	2	1	0
RN9	SN9									0/1
....						...				
RN16	SN16		0/1							
Data 3		Bit	7	6	5	4	3	2	1	0
RN17	SN17									0/1
...						...				
RN24	SN24		0/1							
Data 4		Bit	7	6	5	4	3	2	1	0
RN25	SN25									0/1
...						...				
RN32	SN32		0/1							

Read/write function block data

→ Please also note the relevant description of the function blocks provided in the EZ800 manual.

General notes

Always note the following when working with function blocks:

- The relevant data is transferred in Intel format. In other words, the first byte is the low byte (Byte 4) and the last byte (byte 7) the high byte.
- The maximum data length is 4 bytes. All values must be transferred in hexadecimal format.
- All 32-bit values are treated as signed values. When transferring 32-bit values, ensure that the appropriate value range is suitable for long integers, i.e. signed.
  32-bit value: –2147483648 .. 0 .. +2147483647

Overview

Operands	Meaning	Read/Write	Type (hex)	Page
A01 – A32	Analog value comparator: A01 – A32	Read/Write	11	129
AR01 – AR32	Arithmetic function block: AR01 – AR32	Read/Write	12	131
BC01 – BC32	Block Compare: BC01 – BC32	Read/Write	25	133
BT01 – BT32	Block Transfer: BT01 – BT32	Read/Write	26	135
BV01 – BV32	Boolean operation: BV01 – BV32	Read/Write	13	137
C01 – C32	Counter: C01 – C32	Read/Write	14	139
CF01 – CF04	Frequency counters: CF01 – CF04	Read/Write	15	141
CH01 – CH04	High-speed counter: CH01 – CH04	Read/Write	16	143
CI01 – CI02	Incremental encoder counters: CI01 – CI02	Read/Write	17	145
CP01 – CP32	Comparator: CP01 – CP32	Read/Write	18	147
D01 – D32	Text output function block: D01 – D32	Read/Write	19	149
DB01 – DB32	Data block: DB01 – DB32	Read/Write	1A	152
DC01 – DC32	PID controller: DC01 – DC32	Read/Write	27	154
FT01 – FT32	Signal smoothing filter: FT01 – FT32	Read/Write	28	157
GT01 – GT32	Receipt of network data: GT01 – GT32	Read	1B	159
HW01 – HW32	7-day time switch: HW01 – HW32	Read	1C	161
HY01 – HY32	Year time switch: HY01 – HY32	Read	1D	164
LS01 – LS32	Value scaling: LS01 – LS32	Read/Write	29	167
MR01 – MR32	Master reset: MR01 – MR32	Read	0F	169
NC01 – NC32	Numerical converter: NC01 – NC32	Read/Write	A 2	171
OT01 – OT04	Hours-run meters: OT01 – OT04	Read/Write	1E	173
PT01 – PT32	Sending of network data: PT01 – PT32	Read	1F	175
PW01 – PW02	Pulse width modulation: PW01 – PW02	Read/Write	2B	177
SC01	Synchronize clock function block: SC01	Read	20	179
ST01	Set cycle time function block: ST01	Read/Write	2C	180
T01 – T32	Timing relays: T01 – T32	Read/Write	21	182
VC01 – VC32	Value limitation: VC01 – VC32	Read/Write	2D	185

Analog value comparator: A01 – A32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	11	11
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 62	→ table 62
3 – 6	4 – 7	Data 1 – 4	00	depending on index, → table 63, 64

Table 62: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 63		×	
01	Mode, → table 64		×	
02	Comparison value 1	I1	×	c[1]
03	Gain factor for I1 (I1 = F1 × value)	F1	×	c[1]
04	Comparison value 2	I2	×	c[1]
05	Gain factor for I2 (I2 = F2 × value)	F2	×	c[1]
06	Offset for value I1	O S	×	c[1]
07	Switching hysteresis for value I2 (the value of HY is for both positive and negative hysteresis.)	HY	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 7 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 63: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	CY[1]	Q1[2]

1) Status 1 if the value range is exceeded
2) Status 1 if the condition is fulfilled (e.g. I1 < I2 with LT mode)

Table 64: Index 1 - Mode

Data 1 (hex)		
00	LT	Less than (I1 < I2)
01	EQ	Equal to (I1 = I2)
02	GT	Greater than (I1 > I2)

Arithmetic function block: AR01 – AR32

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	12	12
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 65	→ table 65
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 66, 67
		Write operation	depending on index, → table 66, 67	00

Table 65: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 66		×	
01	Mode, → table 67		×	
02	First operand	I1	×	c[1]
03	Second operand	I2	×	c[1]
04	Result	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 4 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 66: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1
FB output Data 3		–	–	–	–	–	ZE[1]	CY[1]

1) Status 1 if the value of the function block output QV (the calculation result) equals zero
2) Status 1 if the value range is exceeded

Table 67: Index 1 - Mode

Data 1 (hex)		
00	ADD	Add (I1 + I2 = QV)
01	SUB	Subtract (I1 – I2 = QV)
02	MUL	Multiply (I1 × I2 = QV)
03	DIV	Divide (I1 : I2 = QV)

Block Compare: BC01 – BC32

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	25	25
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 68	→ table 68
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 69, 70
		Write operation	depending on index, → table 69, 70	00

Table 68: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 69		×	
01	Mode, → table 70		×	
02	Source range 1	I1	×	c[1]
03	Target range 2	I2	×	c[1]
04	Number of elements to compare: 8 (max. 192 bytes)	NO	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 4 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 69: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	EN[1]
FB output Data 3		–	–	–	–	EQ[2]	E3[3]	E2[4]	E1[5]

1) Activates the function block on status 1.
2) Status 1 if the data ranges are equal; status 0 if not equal

Error outputs

3) Status 1 if the number of elements exceeds the source or target range.
4) Status 1 if the source and target range overlap.
5) Status 1 if the source or target range are outside of the available marker range (offset error)

Table 70: Index 1 - Mode

Mode	Data 1 (hex)	Mode
	02	Compare (internal EZ status signal for Block Compare mode)

Block Transfer: BT01 – BT32

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	26	26
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 71	→ table 71
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 72, 73
		Write operation	depending on index, → table 72, 73	00

Table 71: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 72		×	
01	Mode, → table 73		×	
02	Source range 1	I1	×	c[1]
03	Target range 2	I2	×	c[1]
04	Number of elements to compare: max. 192 bytes	NO	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 and 3 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte .. Data 2 - High Byte).

Table 72: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	T1)
FB output Data 3		–	–	–	–	–	E32)	E23)	E14)

1) Transfer of the source address specified at I1 to the target address specified at I2 on rising edge.

Error outputs

2) Status 1 if the number of elements exceeds the source or target range.
3) Status 1 if the source and target range overlap.
4) Status 1 if the source or target range are outside of the available marker range (offset error)

Table 73: Index 1 - Mode

Data 1 (hex)	Mode
00	INI: Initializes the target range with a byte value stored at the source address.
01	CPY: Copies a data block from a source to a target range. Data block size is specified at NO.

Boolean operation: BV01 – BV32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	13	13
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 74	→ table 74
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 75, 76
		Write operation	depending on index, → table 75, 76	00

Table 74: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 75		×	
01	Mode, → table 76		×	
02	First operand	I1	×	c[1]
03	Second operand	I2	×	c[1]
04	Operation result	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 4 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 75: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1
FB output Data 3		–	–	–	–	–	–	ZE[1]

1) Status 1 if the value of the function block output QV (the operation result) equals zero

Table 76: Index 1 - Mode

Data 1 (hex)		
00	AND	And sequence
01	OR	Or sequence
02	XOR	Exclusive Or sequence
03	NET	Inverts the individual bits of the value at I1. The inverted value is represented as a signed decimal value.

Counter: C01 – C32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	14	14
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 77	→ table 77
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 78
		Write operation	depending on index,→ table 78	00

Table 77: Operand overview

Index (hex)	Operand		Value	read	write
00	Bit IO		→ table 78	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 5 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 78: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	SE[1]	D[2]	C[3]	RE[4]
FB output Data 3		–	–	–	–	ZE[5]	CY[6]	FB[7]	OF[8]

1) Transfer preset actual value on rising edge
2) Count direction: 0 = up counting, 1 = down counting
3) Count coil, counts on every rising edge
4) Reset actual value to zero
5) Zero: Status 1 if the value of the function block output QV (the counter status) equals zero
6) Carry: Status 1 if the value range is exceeded
7) Fall below: Status 1 if the actual value ≦ lower setpoint
8) Overflow: Status 1 if the actual value ≧ upper setpoint

Frequency counters: CF01 – CF04

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	15	15
1	2	Instance	01 – 04	01 – 04
2	3	Index	→ table 79	→ table 79
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 80
		Write operation	depending on index,→ table 80	00

Table 79: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 80		×	
01	Mode/Parameter		–	–
02	Upper setpoint	SH	×	c[1]
03	Lower setpoint	SL	×	c[1]
04	Actual value in Run mode	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 4 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 80: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	EN[1]
FB output Data 3		–	–	–	–	–	ZE[2]	FB[3]	OF[4]

1) Counter enable
2) Zero: Status 1 if the value of the function block output QV (the counter status) equals zero
3) Fall below: Status 1 if the actual value ≦ lower setpoint
4) Overflow: Status 1 if the actual value ≧ upper setpoint

High-speed counter: CH01 – CH04

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	16	16
1	2	Instance	01 – 04	01 – 04
2	3	Index	→ table 81	→ table 81
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 82
		Write operation	depending on index,→ table 82	00

Table 81: Operand overview

Index (hex)	Operand		Value	read	write
00	Bit IO		→ table 82	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 5 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 82: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	EN[1]	SE[2]	D[3]	RE[4]
FB output Data 3		–	–	–	–	ZE[5]	CY[6]	FB[7]	OF[8]

1) Counter enable
2) Transfer preset actual value on rising edge
3) Count direction: 0 = up counting, 1 = down counting
4) Reset actual value to zero
5) Zero: Status 1 if the value of the function block output QV (the counter status) equals zero
6) Carry: Status 1 if the value range is exceeded
7) Fall below: Status 1 if the actual value ≦ lower setpoint
8) Overflow: Status 1 if the actual value ≧ lower setpoint

Incremental encoder counters: CI01 – CI02

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	17	17
1	2	Instance	01 – 02	01 – 02
2	3	Index	→ table 83	→ table 83
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 84
		Write operation	depending on index,→ table 84	00

Table 83: Operand overview

Index (hex)	Operand		Value	read	write
00	Bit IO		→ table 84	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 5 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 84: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	EN[1]	SE[2]	RE[3]
FB output Data 3		–	–	–	–	ZE[4]	CY[5]	FB[6]	OF[7]

1) Counter enable
2) Transfer preset actual value on rising edge
3) Reset actual value to zero
4) Zero: Status 1 if the value of the function block output QV (the counter status) equals zero
5) Carry: Status 1 if the value range is exceeded
6) Fall below: Status 1 if the actual value ≦ lower setpoint
7) Overflow: Status 1 if the actual value ≧ lower setpoint

Comparator: CP01 – CP32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	18	18
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 85	→ table 85
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 86
		Write operation	depending on index, → table 86	00

Table 85: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 86		×	
01	Mode/Parameter		–	–
02	Comparison value	I1	×	c[1]
03	Comparison value	I2	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 and 3 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 86: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1
FB output Data 3		–	–	–	–	GT[1]	EQ[2]	LT[3]

1) greater than: Status 1 if the value at I1 is greater than value at I2 (I1 > I2)
2) equal: Status 1 if the value at I1 is equal to value at I2 (I1 = I2)
3) less than: Status 1 if the value at I1 is less than value at I2 (I1 < I2)

Text output function block: D01 – D32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	19	19
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 87	→ table 87
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 88
		Write operation	depending on index, → table 88	00

Table 87: Operand overview

Index (hex)	Operand	read	write
00	Bit IO, → table 88	×	
01	Mode/Parameter	–	–
02	Text line 1, column 1 - 4	×	
03	Text line 1, column 5 - 8	×	
04	Text line 1, column 9 - 12	×	
05	Text line 1, column 13 - 16	×	
06	Text line 2, column 1 - 4	×	
07	Text line 2, column 5 - 8	×	
08	Text line 2, column 9 - 12	×	
09	Text line 2, column 13 - 16	×	
10	Text line 3, column 1 - 4	×	
11	Text line 3, column 5 - 8	×	
12	Text line 3, column 9 - 12	×	
13	Text line 3, column 13 - 16	×	
14	Text line 4, column 1 - 4	×	
15	Text line 4, column 5 - 8	×	
16	Text line 4, column 9 - 12	×	
17	Text line 4, column 13 - 16	×	
18	Variable 1	×	c[1]
19	Variable 2	×	c[1]
20	Variable 3	×	c[1]
21	Variable 4	×	c[1]
22	Scaling minimum value 1	×	
23	Scaling minimum value 2	×	
24	Scaling minimum value 3	×	
25	Scaling minimum value 4	×	
26	Scaling maximum value 1	×	

Index (hex)	Operand	read	write
27	Scaling maximum value 2	×	
28	Scaling maximum value 3	×	
29	Scaling maximum value 4	×	
30	Control information line 1	×	
31	Control information line 2	×	
32	Control information line 3	×	
33	Control information line 4	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The variables 1 to 4 (index 18 to 21) are transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 88: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	EN[1]
FB output Data 3		–	–	–	–	–	–	–	Q1[2]

1) Text function block enable

2) Status 1, text function block is active

Data block: DB01 – DB32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	1A	1A
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 89	→ table 89
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 90
		Write operation	depending on index, → table 90	00

Table 89: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 90		×	
01	Mode/Parameter		–	–
02	Input value: value that is transferred to the QV output when the FB is triggered.	I1	×	c[1]
03	Output value	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 and 3 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 90: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	T[1]
FB output Data 3		–	–	–	–	–	–	–	Q1[2]

1) Transfer of the value present at I1 on rising edge.
2) Status 1 if the trigger signal is 1.

PID controller: DC01 – DC32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	27	27
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 91	→ table 91
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 92, 93
		Write operation	depending on index, → table 92, 93	

Table 91: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 92		×	
01	Mode, → table 93		×	
02	Setpoint: –32768 to +32767	I1	×	c[1]
03	Actual value: –32768 to +32767	I2	×	c[1]
04	Proportional gain [%], Value range: 0 to 65535	KP	×	c[1]
05	Reset time [0.1 s], Value range: 0 to 65535	TN	×	c[1]
06	Rate time [0.1 s], Value range: 0 to 65535	TV	×	c[1]
07	Scan time = Time between function block calls. Value range: 0.1s to 6553.5s. If 0 is entered as the value, the scan time will be determined by the program cycle time.	TC	×	c[1]
08	Manual manipulated variable, value range: –4096 to +4095	MV	×	c[1]
09	Manipulated variable • Mode: UNI, value range: 0 to +4095 (12 bit) • Mode: BIP, value range: –4096 to +4095 (13 bit)	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 9 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte .. Data 2 - High Byte).

Table 92: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	SE[1]	ED[2]	EI[3]	EP[4]	EN[5]
FB output Data 3		–	–	–	–	–	–	–	LI[6]

1) Transfer of manual manipulated variable on status 1
2) Activation of D component on status 1
3) Activation of I component on status 1
4) Activation of P component on status 1
5) Activates the function block on status 1.
6) Status 1 if the value range of the medium-voltage was exceeded

Table 93: Index 1 - Mode

Data 1	Mode
UNP unipolar	The manipulated variable is output as a unipolar 12-bit value. Corresponding value range for QV 0 to 4095.
BIP bipolar	The manipulated variable is output as a bipolar 13-bit value. Corresponding value range for QV –4096 to 4095

Signal smoothing filter: FT01 – FT32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	28	28
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 94	→ table 94
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 95
		Write operation	depending on index,→ table 95	00

Table 94: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 95		×	
01	Mode/Parameter		–	–
02	Input value, value range: –32768 to +32767	I1	×	c[1]
03	Recovery time [0.1 s], Value range: 0 to 65535	TG	×	c[1]
04	Proportional gain [%], Value range: 0 to 65535	KP	×	c[1]
05	Delayed output value, value range: –32768 to +32767	Q V	×	

1) The value can only be written if it is assigned to a constant in the program.

Table 95: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	EN[1]

1) Activates the function block on status 1.

Receipt of network data: GT01 – GT32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	1B	1B
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 96	→ table 96
3 – 6	4 – 7	Data 1 – 4	00	depending on index, → table 97, 98

Table 96: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 97		×	
01	Mode/Parameters, → table 98		×	–
02	Output value: actual value from the network	QV	×	

→ The data for index 2 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 97: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	Q[1]

1) Status 1 if a new value is present that is transferred from the NET network.

Table 98: Index 1 – Mode/Parameters (designation of PUT FB with data to be received)

Mode	Data 1	NET-ID[1]	
		0	NET-ID 1
		..	..
		7	NET-ID 8
Parameters	Data 3	Instance[2]	
		0	PT01
		..	..
		31	PT32

1) Number of station sending the value. Possible station numbers: 01 to 08
2) Send FB (e.g. PT 20) of the sending NET station. Possible station numbers: 01 – 32

7-day time switch: HW01 – HW32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	1C	1C
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 99	→ table 99
3 – 6	4 – 7	Data 1 – 4	00	depending on index, → table 100

Table 99: Operand overview

Index (hex)	Operand		read	write
00	Bit IO	→ table 100	×	
01	Mode/Parameter		–	–
02	Parameters	→ table 101	×	
	Channel A			
03	Channel B			
04	Channel C			
05	Channel D			

Table 100: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	Q[1]

1) Status 1 if the switch-on condition is fulfilled.

The data in the following table is shown in the Motorola format although it is actually transferred in Intel format.

Table 101: Index 2 – 5, Parameter channels A – D

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 2								Date 1							
ON	d4	d3	d2	d1	d0	h4	h3	h2	h1	h0	m5	m4	m3	m2	m1	m0
	Weekday					Hour					Minute					

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 4								Date 3							
OFF	d4	d3	d2	d1	d0	h4	h3	h2	h1	h0	m5	m4	m3	m2	m1	m0
	Weekday					Hour					Minute					

m5 to m0: Minute (0 to 59)
h4 to h0: Hour (0 to 23)
d5 to d0: Weekday (0 = Sunday to 6 = Saturday)

Example

The channel A parameters of 7-day time switch HW19 are to be read.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Attribute ID: Read	92	–
	Response: Read successful	–	C2
1	Type	1C	1C
2	Instance	13	13
3	Index	02	02
4	Data 1	00	62
5	Data 2	00	0B
6	Data 3	00	7B
7	Data 4	00	25

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 2 = $0B_{hex}$								Date 1 = 62_{hex}							
ON	0	0	0	0	1	0	1	1	0	1	1	0	0	0	1	0
	Weekday					Hour					Minute					

Switch-on time:
Weekday = 01_{hex} .. Monday
Hour = $0D_{hex}$ .. 1300 hours
Minute = 22_{hex} .. 34 minutes

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 4 = 25_{hex}								Date 3 = $7B_{hex}$							
OFF	0	0	1	0	0	1	0	1	0	1	1	1	1	0	1	1
	Weekday					Hour					Minute					

Switch-off time:
Weekday = 04_{hex} .. Thursday
Hour = 15_{hex} .. 2100 hours
Minute = 59_{hex} .. 34 minutes

Year time switch: HY01 – HY32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	1D	1D
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 102	→ table 102
3 – 6	4 – 7	Data 1 – 4	00	depending on index,→ table 103

Table 102: Operand overview

Index (hex)	Operand		read	write
00	Bit IO	→ table 103	×	
01	Mode/Parameter		–	–
02	Parameters Channel A	→ table 104	×	
03	Channel B			
04	Channel C			
05	Channel D			

Table 103: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	Q[1]

1) Status 1 if the switch-on condition is fulfilled.

The data in the following table is shown in the Motorola format although it is actually transferred in Intel format.

Table 104: Index 2 – 5, Parameter channels A – D

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 2								Date 1							
ON	y6	y5	y4	y3	y2	y1	y0	m3	m2	m1	m0	d4	d3	d2	d1	d0
	Year							Month				Day				

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 4								Date 3							
OFF	y6	y5	y4	y3	y2	y1	y0	m3	m2	m1	m0	d4	d3	d2	d1	d0
	Year							Month				Day				

d4 ... d0: Day (1 .. 31), m3 ... m0: Month (1 .. 12), y6 ... y0: Year (0: 2000 .. 99: 2099)

Example

The channel A parameters of year time switch HY14 are to be written.

Index 2 – 5, Parameter channels A – D

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 2								Date 1							
ON	0	0	0	0	0	1	1	0	1	1	0	0	1	1	1	0
	Year							Month				Day				

Switch-on time:
Day = 14 = $0E_{hex}$ = 0000 1110_{bin}
Month = 6 (June) = 06_{hex} = 0000 0110_{bin}
Year = 2003 = 03_{hex} = 0000 0011_{bin}

Index 2 – 5, Parameter channels A – D

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 2								Date 1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Date 4								Date 3							
OFF	y6	y5	y4	y3	y2	y1	y0	m3	m2	m1	m0	d4	d3	d2	d1	d0
	Year							Month				Day				

Switch-off time:
Day = 3 = 03_{hex} = $0000\ 0011_{bin}$
Month = 10 (October) = $0A_{hex}$ = $0000\ 1010_{bin}$
Year = 2012 = $0C_{hex}$ = $0000\ 1100_{bin}$

Resulting telegram:

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Attribute ID: Write	B2	–
	Response: Write successful	–	C1
1	Type	1D	1D
2	Instance	0E	0E
3	Index	02	02
4	Data 1	8E	00
5	Data 2	06	00
6	Data 3	43	00
7	Data 4	19	00

Value scaling: LS01 – LS32

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	29	29
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 105	→ table 105
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 106
		Write operation	depending on index,→ table 106	

Table 105: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 106		×	
01	Mode/Parameter		–	–
02	Input value, value range: 32 bit	I1	×	c[1]
03	Interpolation point 1, X co-ordinate, value range: 32 bit	X1	×	c[1]
04	Interpolation point 1, Y co-ordinate, value range: 32 bit	Y1	×	c[1]
05	Interpolation point 2, X co-ordinate, value range: 32 bit	X2	×	c[1]
06	Interpolation point 2, Y co-ordinate, value range: 32 bit	Y2	×	c[1]
07	Output value: contains the scaled input value	QV	χ	

1) The value can only be written if it is assigned to a constant in the program.

Table 106: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	EN[1]

1) Activates the function block on status 1.

Master reset: MR01 – MR32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	0F	0F
1	2	Instance	01 – 20	01 – 20
2	3	Index		
		Bit IO	00	00
		Mode	01	01
3 – 6	4 – 7	Data 1 – 4	00	depending on index, → table 107, 108

Table 107: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	T[1]
FB output Data 3		–	–	–	–	–	–	–	Q1[2]

1) Trigger coil. The appropriate Reset is executed if the coil is triggered (with a rising edge).
2) Status 1 if the trigger coil MR..T is 1.

Table 108: Index 1 - Mode

Data 1 (hex)		
00	Q	Outputs Q.., *Q.., S.., *S.., *SN.., QA01 are reset to 0. * depending on the NET-ID
01	m	The marker range MD01 to MD48 is reset to 0.
02	ALL	Has an effect on Q and M.

Numerical converter: NC01 – NC32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	A 2	A 2
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 109	→ table 109
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 110, 111
		Write operation	depending on index, → table 110, 111	00

Table 109: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 110		×	
01	Mode, → table 111		×	
02	Input value: operand to be converted	I1	×	c[1]
03	Output value: contains the conversion result	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 and 3 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte .. Data 2 - High Byte).

Table 110: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	EN[1]

1) Activates the function block on status 1.

Table 111: Index 1 - Mode

Data 1 (hex)		
00	BCD	Converts a BCD coded decimal value to an integer value.
01	BIN	Converts an integer value to a BCD coded decimal value.

Hours-run meters: OT01 – OT04

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	1E	1E
1	2	Instance	01 – 04	01 – 04
2	3	Index	→ table 112	→ table 112
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 113
		Write operation	depending on index, → table 113	00

Table 112: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 113		×	
01	Mode/Parameter		–	–
02	Upper threshold value	I1	×	c[1]
03	Actual value of operating hours counter	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

Table 113: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	RE[1]	EN[2]
FB output Data 3		–	–	–	–	–	–	–	Q1[3]

1) Reset coil: Status 1 resets the counter actual value to zero.
2) Enable coil
3) Status 1 if the setpoint was reached (greater than/equal to)

→ The data for index 2 and 3 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Sending of network data: PT01 – PT32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	1F	1F
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 114	→ table 114
3 – 6	4 – 7	Data 1 – 4	00	depending on index,→ table 115

Table 114: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 115		×	
01	Mode/Parameter		–	–
02	Input value: Setpoint that it transmitted to the NET network	I1	×	

→ The data for index 2 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 115: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	T[1]
FB output Data 3		–	–	–	–	–	–	–	Q1[2]

1) Trigger coil. The value is provided on the NET if the coil is triggered (with a rising edge).
2) Status 1 if the trigger coil PT..T_ is also 1.

Pulse width modulation: PW01 – PW02

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	2B	2B
1	2	Instance	01 – 02	01 – 02
2	3	Index	→ table 116	→ table 116
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 117
		Write operation	depending on index,→ table 117	00

Table 116: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 117		×	
01	Mode/Parameter		–	–
02	Manipulated variable, value range: 0 to 4095 (12 bit)	SV	×	c[1]
03	Period duration [ms], Value range: 0 to 65535	PD	×	c[1]
04	Minimum on duration [ms], Value range: 0 to 65535	ME	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

Table 117: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	EN[1]
FB output Data 3		–	–	–	–	–	–	–	E1[2]

1) Activates the function block on status 1.

2) Status 1 if below the minimum on duration or minimum off duration

Synchronize clock function block: SC01

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID: Read	92	–
	0	Response:		
		Read successful	–	C2
		Command rejected	–	C0
0	1	Type	20	20
1	2	Instance	01	01
2	3	Index	→ table 118	→ table 118
3 – 6	4 – 7	Data 1 – 4	00	depending on index, → table 119

Table 118: Operand overview

Index (hex)	Operand	read	write
00	Bit IO, → table 119	×	
01	Mode/Parameter	–	–

Table 119: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	–	–	T[1]
FB output Data 3		–	–	–	–	–	–	–	Q1[2]

1) Trigger coil. If the coil is triggered (rising edge), the current date, weekday and time of the sending station are automatically sent to the NET network.
2) Status 1 if the trigger coil SC01T_ is also 1.

Set cycle time function block: ST01

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	2C	2C
1	2	Instance	01	01
2	3	Index	→ table 120	→ table 120
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 121
		Write operation	depending on index,→ table 121	00

Table 120: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 121		×	
01	Mode/Parameter		–	–
02	Cycle time in ms, value range: 0 – 1000	I1	×	c[1]

1) The value can only be written if it is assigned to a constant in the program.

Table 121: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	EN[1]

1) Activates the function block on status 1.

Timing relays: T01 – T32

Telegram structure

Byte		Meaning	Value (hex), sent by	
Master	Slave		Master	Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	21	21
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 122	→ table 122
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index, → table 123, 124
		Write operation	depending on index, → table 123, 124	

Table 122: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 123		×	
01	Mode/Parameters, → table 124		×	
02	Setpoint 1: Time setpoint 1	I1	×	c[1]
03	Setpoint 2: Time setpoint 2 (with timing relay with 2 setpoints)	I2	×	c[1]
04	Actual value: Time elapsed in Run mode	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

→ The data for index 2 to 4 is transferred as a 32-bit value in Intel format (Data 1 – Low Byte to Data 4 – High Byte).

Table 123: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB input Data 1		–	–	–	–	–	ST[1]	EN[2]	RE[3]
FB output Data 3		–	–	–	–	–	–	–	Q1[4]

1) Stop, the timing relay is stopped (Stop coil)
2) Enable, the timing relay is started (trigger coil)
3) Reset, the timing relay is reset (reset coil)
4) Switch contact

Table 124: Index 1 - Mode/Parameters

Mode	Data 1	Mode
	0	On-delayed,
	1	On-delayed with random setpoint
	2	off-delayed.
	3	Off-delayed with random setpoint
	4	On and off delayed (two time setpoints)
	5	On and off delayed each with random setpoint (two time setpoints)
	6	Impulse transmitter
	7	Flashing relay (two time setpoints)
	8	Off-delayed, retriggerable
	9	Off-delayed with random setpoint, retriggerable
Param eters	Data 3	Mode
	0	S (milliseconds)
	1	M:S (seconds)
	2	H:M (minutes)

Value limitation: VC01 – VC32

Telegram structure

Byte Master	Byte Slave	Meaning	Value (hex), sent by Master	Value (hex), sent by Slave
		Attribute ID		
		Read	92	–
		Write	B2	–
	0	Response:		
		Read successful	–	C2
		Write successful	–	C1
		Command rejected	–	C0
0	1	Type	2D	2D
1	2	Instance	01 – 20	01 – 20
2	3	Index	→ table 125	→ table 125
3 – 6	4 – 7	Data 1 – 4		
		Read operation	00	depending on index,→ table 126
		Write operation	depending on index,→ table 126	00

Table 125: Operand overview

Index (hex)	Operand		read	write
00	Bit IO, → table 126		×	
01	Mode/Parameter		–	–
02	Input value	I1	×	c[1]
03	Upper limit value	SH	×	c[1]
04	Lower limit value	SL	×	c[1]
05	Output value: outputs the value present at input I1 within the set limits.	QV	×	

1) The value can only be written if it is assigned to a constant in the program.

Table 126: Index 0 – Bit IO

	Bit	7	6	5	4	3	2	1	0
FB output Data 3		–	–	–	–	–	–	–	EN[1]

1) Activates the function block on status 1.

Analysis – error codes via EZ-LINK

The EZ800/EZD basic unit will return a defined error code in the event of an incorrectly selected operating mode or an invalid telegram. The error code transferred has the following structure:

Telegram structure

Byte	Meaning	Slave transmits (value hex)
0	Answer	
	Command rejected	C0
1	Type	
2	Instance	
3	Index	
4	Error code	→ table 127
5 – 7	Data 2 – 4	

Table 127: Error codes

Error code	Description
0x00	no error
0x03	formal fault in the response relating to type, instance or index
0x04	no communication possible (timeout)
0x05	DP module has only sent 0xC0 (EZ800/EZD).
0x45	the value selected by the type and index may not be written (bit IO, mode/parameter or output value).
0x46	the value selected by the type and index is not assigned with a constant.
0x9E	access to the FB data not possible (program download active).
0x9F	type is invalid (no defined FB, also dependant on the version of the addressed device).
0xA0	FB selected by type and instance does not exist in program.
0xA1	index relative to the defined FB type is invalid

Appendix

What happens if...?

Module Status LED MS	Possible cause	To correct or avoid error
OFF	No power at EZ222-DN.	Switch on the power supply.
Green	EZ222-DN is in standby mode.	None
Green flashing	EZ222-DN not configured.	Verify the correct setting of the MAC ID.
Red flashing	Invalid configuration	Check configuration data.
RED	Module error which cannot be resolved.	Replace the EZ222-DN.

Network Status LED NS	Possible cause	To correct or avoid error
OFF	• EZ222-DN without power or • communication is blocked at this channel because – of bus-off state or – power loss or – the channel was blocked explicitly.	• Switch on the EZ222-DN, • supply the mains voltage to the channel and • ensure that the channel is active.
Green	Although the channel is enabled, communication is not possible.	Check the communication function at the master PLC.
Green flashing	Normal mode	None
Red flashing	Communication error or the EZ222-DN may be defective.	Reset the module. If further errors occur, replace the EZ222-DN.
RED	Communication error.	Check the master PLC.

Technical Data

General		
Standards and regulations		EN 61000-6-1; EN 61000-6-2; EN 61000-6-3; EN 61000-6-4, IEC 60068-2-27, IEC 50178
Dimensions W × H × D	mm	35.5 × 90 × 56.5
Weight	g	150
Mounting		DIN 50022 rail, 35 mm screw fixing with fixing bracket EZB4-101-GF1 (accessories)
Climatic environmental conditions (Cold to IEC 60068-2-1, Heat to IEC 60068-2-2)		
Ambient temperature Installed horizontally/vertically	°C	–25 to +55
Condensation		Prevent condensation with suitable measures
Storage/transport temperature	°C	–40 to +70
Relative humidity (IEC 60068-2-30), no moisture condensation	%	5 to 95
Air pressure (operation)	hPa	795 to 1080
Corrosion resistance (IEC 60068-2-42, IEC 60068-2-43)		SO_2 10 cm^3/m^3, 4 days H_2S 1 cm^3/m^3, 4 days
Mechanical ambient conditions		
Pollution degree		2
Degree of protection (EN 50178, IEC 60529, VBG4)		IP20
Vibration (IEC 60068-2-6)		
constant amplitude 0.15 mm	Hz	10 to 57
constant acceleration 2 g	Hz	57 to 150
Shocks (IEC 60068-2-27) semi-sinusoidal 15 g/11 ms	Shocks	18
Drop (IEC 60068-2-31) height	mm	50
Free fall, when packed (IEC 60068-2-32)	m	1

Electromagnetic compatibility (EMC)		
Electrostatic discharge (ESD), (IEC/EN 61000-4-2, severity level 3)		
Air discharge	kV	8
Contact discharge	kV	6
Electromagnetic fields RFI), (IEC/EN 61000-3	V/m	10
Radio interference suppression (EN 55011, EN 55022), class		B
Burst (IEC/EN 61000-4-4, severity level 3)		
Power cables	kV	2
Signal cables	kV	2
High energy pulses (Surge) EZ-AC (IEC/ EN 61000-4-5), power cable symmetrical	kV	1
High-energy pulses (surge) of EZ DC current (IEC/EN 61 000-4-5, severity level 2), power cable symmetrical	kV	0,5
Line-conducted interference (IEC/EN 61000-4-6)	V	10
Dielectric strength		
Measurement of the clearance and creepage distance		EN 50178, UL508, CSA C22.2 No. 142
Dielectric strength		EN 50 178
Tools and cable cross-sections		
Conductor cross-sections		
Solid, minimum to maximum	mm^2	0.2 to 4
	AWG	22 to 12
Flexible with ferrule, minimum to maximum	mm^2	0.2 to 2.5
	AWG	22 to 12
Slot-head screwdriver, width	mm	3.5 × 0.8
Tightening torque	N/m	0,5

Power supply		
Rated voltage		
Rated value	V DC	24 (–15, +20)
Permissible range	V DC	20.4 to 28.8
Residual ripple	%	< 5
Input current at 24 V DC, typical	mA	200
Voltage dips, IEC/EN 61131-2	ms	10
Power loss at 24 V DC, typical	W	4,8
LED displays		
Module Status LED MS	Color	Green/red
Network Status LED NS	Color	Green/red
DeviceNet		
Device connection		5-pole socket
Electrical isolation		Bus to power supply (simple) Bus and power supply to EZ/EZD basic unit (safety isolation)
Function		DeviceNetSlave
INTERFACE		DeviceNet (CAN)
Bus protocol		DeviceNet
Baud rate, automatic detection up to	kbps	500
Bus termination resistors		Separate installation at the bus possible
Bus addresses, accessible via EZ basic unit with display or EZSoft		0 to 63
Services		
Module inputs		all data S1 to S8 EZ/EZD
Module outputs		all data R1 to R16 EZ/EZD
Module control commands		Read/Write Weekday, time-of-day, summer/winter time All parameters of the EZ/EZD functions

Dimensions

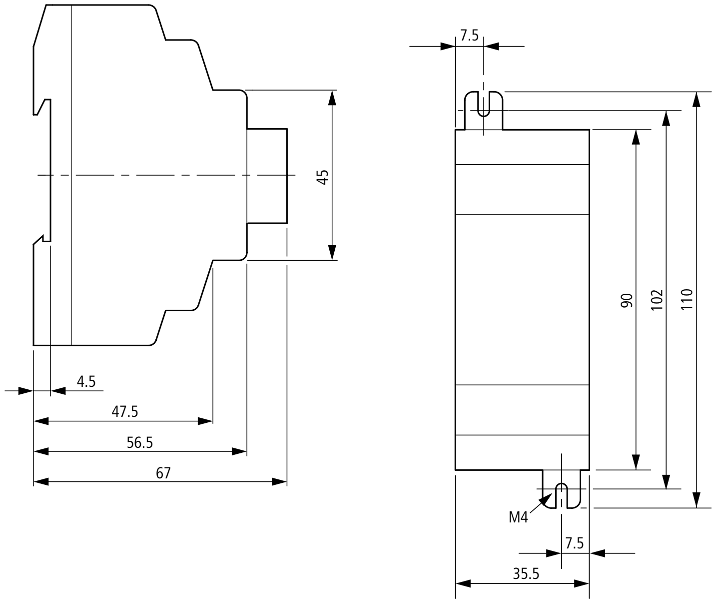


Figure 13: EZ222-DN dimensions in [mm]

Glossary

This glossary refers to topics related to DeviceNet.

Acknowledge	Acknowledgement returned by the receiving station after having received a signal.
Active metallic component	Conductor or conductive component that is live when in operation.
Address	Number that identifies a memory area, systems or module within a network, for example.
Addressing	Assignment or setting of an address for a module in the network, for example.
Analog	Value, such as voltage, that is infinitely variable and proportional. Analog signals can acquire any value within specific limits.
Automation product	I/O controlling device that is interconnected to a system process. PLCs represent a special group of automation products.
Baud	Unit for the data transfer rate. One baud is equivalent to the transmission of one bit per second (bps).
Baud rate	Unit of measure of the data transmission speed in bit/s.
Bidirectional	Operation in both directions.
Bit	Abbreviation for the term “binary digit“. Represents the smallest information unit of a binary system. Its significance can be 1 or 0 (Yes/No decision).
Bus	Bus system for data exchange, for example between the CPU, memory and I/O. A bus can consist of several parallel segments, e.g the data bus, address bus, control bus and power supply bus.
Bus cycle time	Time interval in which a master provides services to all slaves or nodes of a bus system, i.e. writes data to their outputs and reads inputs.
Bus line	Smallest unit connected to the bus. Consists of the PLC, a module and a bus interface for the module.

Bus system	All units as a whole which communicate across a bus.
Byte	A sequence of 8 bits
Capacitive coupling	Capacitive (electrical) coupling develops between two conductors carrying different potentials. Typical interference sources are, for example parallel signal cables, contactor relays and static discharge.
Chassis ground	All interconnected inactive equipment parts which are not subject to hazardous fault voltage.
Code	Data transfer format
Coding element	Two-part element for the unambiguous allocation of electronic and basic module.
Command modules	Command-capable modules are modules with an internal memory that are capable of executing particular commands (such as output substitute values).
Common potential	Electrical interconnection of the reference potentials of the control and load circuit of I/O modules.
CONFIGURE...	Systematic arrangement of the I/O modules of a station.
COS I/O connection	COS (Change Of State) I/O connections are used to set up event-controlled connections, i.e. the DeviceNet devices automatically generate messages when a status has changed.
	2 byte diagnostics data of the EZ control relay
	Coupling module status
CPU	Abbreviation for “Central Processing Unit“. Central unit for data processing. Represents the core element of a computer.
Cyclic I/O connection	Message triggering is timer-controlled when operating with a cyclic I/O connection.
Device Heartbeat Message	A DeviceNet unit can use the Device Heartbeat Message function to broadcast its native status at set time intervals. These messages are configured in the Identity Object.
Device Shut Down Message	A device shutting down due to internal errors or states can log off at the PLC by means of the Device Shut Down Message.

Digital	Represents a value that can acquire only definite states within a finite set, e.g. a voltage. Mostly defined as "0" and "1".
DIN	Abbreviation for "Deutsches Institut für Normungen e. V." (German Institute for Standardization).
Dual Code	Natural binary code. Frequently used code for absolute measurement systems.
Earth	Defines in electrical engineering the conductive earth whose electrical potential is equal to zero at any point. The electrical potential in the area of earthing devices might not be equal to zero. In this case, one refers to "Reference ground".
Earth electrode	One or several components with direct and good contact to earth.
Earthing	Represents the connection of an electrically conductive component to the equipotential earth via a grounding device.
Earthing tape	Flexible conductor, mostly braided. Interconnects inactive parts of equipment, e.g. the doors of a control panel and the switch cabinet body.
EDS	This EDS file primarily defines the Polled I/O Connection, the COS I/O Connection and the Cyclic I/O Connection of the gateway. It does not contain data or parameters (EZ object) for functions of the EZ basic unit. These functions are accessed by means of explicit messages.
EEPROM	Abbreviation for “Electrically Erasable Programmable Read-only Memory“.
Electrical equipment	Comprises all equipment used for the generation, conversion, transfer, distribution and application of electrical energy, e.g. power lines, cables, machines, controllers.
EMC	Abbreviation for "Electromagnetic Compatibility". Defines the ability of electrical equipment to operate error-free and without causing a negative influence within a certain environment.
EN	Abbreviation for “European Norm”.

Equipotential bonding	Adaptation of the electrical level of the body of electrical equipment and auxiliary conductive bodies by means of an electrical connection.
ESD	Abbreviation for “Electrostatic Discharge”.
Field power supply	Power supply for the field devices and signal voltage.
Fieldbus	Data network on the sensor/actuator level. The fieldbus interconnects the devices at field level. Characteristic feature of the fieldbus is the highly reliable transfer of signals and real-time response.
Galvanic coupling	Galvanic coupling generally develops between two circuits using a common cable. Typical interference sources are starting motors, static discharge, clocked devices and potential difference between the component enclosure and their common power supply.
GND	Abbreviation for “GROUND” (zero potential).
Hexadecimal	Numerical system with the base 16. The count starts at 0 to 9 and continues with the letters A, B, C, D, E and F.
I/O	Abbreviation for “Input/Output“.
Impedance	Alternating current-resistance of a component or of a circuit consisting of several components at a specific frequency.
Inactive metallic parts	Touch-protected conductive components, isolated electrically from active metallic parts by means of an insulation, but subject to fault-voltage.
Inductive coupling	Inductive (magnetic) coupling develops between two current-carrying conductors. The magnetic effect generated by the currents induces an interference voltage. Typical interference sources are, for example transformers, motors, mains cables installed parallel and RF signal cables.
Lightning protection	Represents all measures for preventing system damage due to overvoltage caused by lightning strike.
Low-impedance connection	Connection with low alternating-current resistance.
LSB	Abbreviation for “Least Significant Bit“. Bit with the least significant value.

Master	Station or node in a bus system that controls communication between the other stations of the bus system.
Master/Slave Mode	Operating mode in which a station or node of the system acts as master that controls communication on the bus.
Mode	Operating mode.
Module bus	Represents the internal bus of an XI/ON station. Used by the XI/ON modules for communication with the gateway. Independent of the fieldbus.
MSB	Abbreviation for “Most Significant Bit“. Bit with the most significant value.
Multimaster Mode	Operating mode in which all stations or nodes of a system have equal rights for communicating on the bus.
NAMUR	Abbreviation for "Normen-Arbeitsgemeinschaft für Mess- und Regeltechnik" (Standards Work Group for Instruments and Controls). NAMUR proximity switches represent a special category of 2-wire proximity switches. They are highly resistant to interference and reliable due to their special construction, e.g. low internal resistance, few components and short design.
Noise emission (EMC)	Testing procedure to EN 61000-6-4
Noise immunity (EMC)	Testing procedure to EN 61000-6-2
Offline Connection Set	The Offline Connection Set allows communication with a device that is in communication error state but not in bus-off state due to an ambiguous address. It is usually no longer possible to address this device on the network, and it must be initialized manually by switching it off and on. The Offline Connection Set can be used in this situation to address such a device on the network.
Overhead	System management time. Required once for each data transfer cycle.
Parameter assignment	Definition of parameters for individual bus stations or their modules in the configuration software of the DeviceNet master.
PLC	Abbreviation for Programmable Logic Controller.

Polled I/O connection	A polled I/O connection is used to establish a conventional master/slave relation between a PLC and a DeviceNet device, and represents a PtP connection between two stations on the fieldbus. The master (client) transmits a polling request to the slave (server), and this answers with a polling response. • 3 bytes of output data S1 to S8 EZ/EZD output range, RUN/STOP (inputs at the DeviceNet master) • 3 bytes of input data R1 to R16 EZ/EZD input range, RUN/STOP (outputs of the DeviceNet master)
Potential-free	Galvanic isolation between the reference potentials of the control and load circuit of I/O modules.
Protected against short-circuit	Property of electrical equipment. Short-circuit-proof equipment has the ability to withstand the thermal and dynamic loads that may occur at the location of installation on account of a short-circuit.
Protective conductor	Conductor required for human body protection against hazardous currents. Abbreviation: PE (“Protective Earth”).
Radiation coupling	Radiated coupling occurs when an electromagnetic wave makes contact with a conductor structure. The impact of the wave induces currents and voltages. Typical interference sources are, for example ignition circuits (spark plugs, commutators of electrical motors) and transmitters (e.g. radio-operated devices), which are operated near the corresponding conductor structure.
Reference ground	Earth potential in the area of grounding devices. May have a potential other than the zero of “earth” potential.
Reference potential	Represents a reference point for measuring and/or visualizing the voltage of any connected electrical circuits.
Repeater	Amplifier for signals transferred across a bus.

Response time	In a bus system this represents the time interval between the transmission of a read request and receiving the answer. Within an input module, it represents the time interval between the signal change at an input and its output to the bus system.
Screen	Term that describes the conductive covering of cables, cubicles and cabinets.
Serial	Describes an information transfer technique. Data are transferred in a bit-stream across the cables.
Shielding	Refers to all measures and equipment used to connect system parts to the screen.
Slave	Station or node in a bus system that is subordinate to the master.
Station	Function unit or module, consisting of several elements.
Terminating resistor	Terminating resistor at the start and end of a bus cable. Prevents interference due to signal reflection and is used for the adaptation of bus cables. Bus terminating resistors must always be the last unit at the end of a bus segment.
Topology	Geometrical network structure, or circuit arrangement.
UART	Abbreviation for “Universal Asynchronous Receiver/ Transmitter”. A UART represents a logical circuit used to convert an asynchronous serial data stream into a parallel bit stream and vice versa.
UCMM	The DeviceNet gateway provides an option of configuring dynamic connection objects via the UCMM port (Unconnected Message Manager Port).
Unidirectional	Operating in one direction.

Index

Eaton Electrical
1000 Cherrington Parkway
Moon Township, PA 15108-4312
USA
Tel: 1-800-525-2000
www.EatonElectrical.com

Printed in USA
Publication No. MN05013002E
May 2005