



Building Automation

Industrial Automation

Systems

User Manual

EASY222-DN DeviceNet Slave Interface

08/02 AWB2528-1427GB

MOELLER 

Think future. Switch to green.

For Immediate Delivery call KMParts.com at (866) 595-9616

All brand and product names are trademarks or registered trademarks of the owner concerned.

1st published 2002, edition date 08/02

© Moeller GmbH, 53105 Bonn

Author: Ronny Happ
Editor: Michael Kämper
Translator: Harold Schierbaum

All rights reserved, including those of the translation.

No part of this manual may be reproduced in any form (printed, photocopy, microfilm or any other process) or processed, duplicated or distributed by means of electronic systems without written permission of Moeller GmbH, Bonn.

Subject to alteration without notice.

For Immediate Delivery call KMParts.com at (866) 595-9616



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.
- Earth and short circuit.
- Cover or enclose neighbouring 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 equalisation. 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	5
	Target group	5
	Further manuals for this device	5
	Device designation	5
	Abbreviations and symbols	6
1	The EASY222-DN	7
	System overview	7
	Structure of the unit	8
	EASY222-DN Communication profile	9
	Hardware and Operating system requirements	9
	Use other than intended	10
2	Installation	11
	Connection of EASY222-DN to the basic unit	11
	Connecting the power supply	12
	DeviceNet connection	13
	– Terminal assignment DeviceNet	13
	– Terminating resistors	13
	EMC compatible wiring	14
	Potential isolation	15
	– Transmission rates – Automatic recognition of the baud rate	16
	– Maximum distances and bus cable lengths	16

3	Device operation	17
	Initial power on	17
	DeviceNet Slave address setting	17
	– Setting the address at the basic unit with display:	18
	– Setting the address by means of EASY-SOFT	19
	– Setting the address via the master PLC	19
	LED Status displays	20
	– Module status LED	20
	– Network Status LED	21
	Cycle time easy basic unit	22

4	DeviceNet functions	23
	DeviceNet communication profile	23
	– I/O Messages	23
	– Explicit Messages	24
	– Polled I/O connection	24
	– COS I/O connection	25
	– Cyclic I/O connection	25
	– UCMM	25
	– Offline Connection Set	25
	– Device Heartbeat Message	26
	– Device Device Shut Down Message	26
	EDS file	26
	Object Model	27
	– Management objects	28
	– Connection Objects	29
	– Application-specific objects	29
	– Identity Object	30
	– DeviceNet object	32
	– easy object	33
	Data exchange method Control commands	40
	– Overview	43
	– Timing relays T1 to T8: Setting parameters	44
	– Counter relays C1 to C8: Setting parameters	47
	– Switching timers: Setting the control bytes	49
	– Switching timers 1 to 4:	
	Setting the channels A to D	51
	– Setting the analogue value comparators 1 to 8	54
	– Setting the real-time clock	57

– Timing relays 1 to 8: Reading the process variables	59
– Timing relays 1 to 8: Reading the reference values	62
– Counter relays 1 to 8: Reading the process variables	64
– Counter relays 1 to 8: Reading the reference values	66
– Switching timers 1 to 4: Reading channel A to D	68
– Reading analogue inputs	73
– Reading the status of digital inputs, P buttons and operator keys	74
– Reading the status of the real-time clock	77
– Reading the status of timing relays, counter relays, switching timers and of analogue value comparators	79
– Reading the status of contactor relays (markers), text display and digital outputs	84
<hr/>	
5 What happens if...?	89
<hr/>	
Annex	91
Technical Data	91
Dimensions	94
EDS file	95
<hr/>	
Glossary	99
<hr/>	
Index	105

About this manual

Target group	<p>This manual is targeted at automation technicians and engineers.</p> <p>We presume a profound knowledge of programming fieldbus DeviceNet master systems DeviceNet and knowledge of the easy control relay.</p>
Further manuals for this device	<p>We principally refer to the Control Relay easy412, easy600 (AWB2528-1304-...) User Guide.</p>
Device designation	<p>The following short names for equipment types are used in this manual, as far as the description applies to all of these types:</p> <ul style="list-style-type: none">• easy600 for<ul style="list-style-type: none">– EASY6..-AC-RC(X)– EASY6..-DC-.C(X)• easy-AC for<ul style="list-style-type: none">– EASY412-AC-..– EASY6..-AC-RC(X)• easy-DC for EASY620/621-DC-.C(X)

Abbreviations and symbols

This manual uses abbreviations and symbols which have the following meaning:

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

► indicates actions to be taken.



Note!

Warns of the risk of slight damage to the product or components.



Caution!

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



Warning!

Warns of the risk of major damage to assets and of serious injury or even death.



Draws your attention to interesting tips and supplementary information

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.

1 The EASY222-DN

EASY222-DN was developed for automation tasks using the fieldbus DeviceNet system. EASY222-DN represents a "gateway" and can only be operated in combination with expandable easy600 base units. The control relay easy600 with DeviceNet gateway EASY222-DN always operates as network slave.

System overview

The easy DeviceNet slaves are integrated into a DeviceNet fieldbus system.

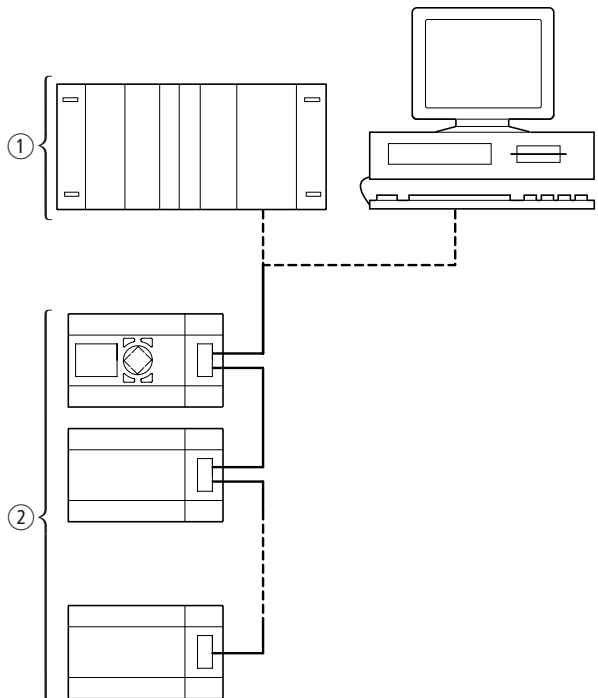


Figure 1: Implementation of EASY222-DN in the DeviceNet

- ① Master area, PLC (e.g.: SLC 500) or PC with CAN card
- ② Slave area, e.g.: Control relay easy with DeviceNet interface

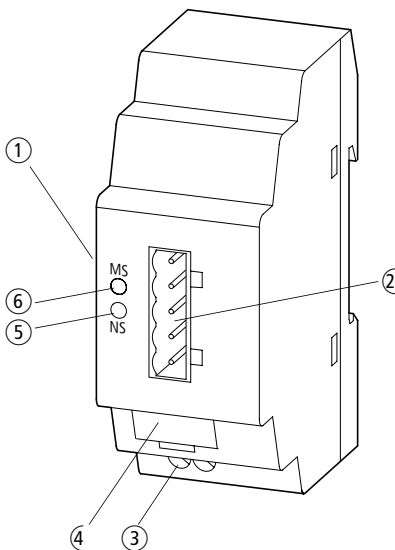
Structure of the unit

Figure 2: Structure of EASY222-DN

- ① EASY-LINK socket
- ② 5-pin DeviceNet connection to ODVA
- ③ Power supply 24 V $\overline{\text{---}}$
- ④ Equipment rating plate
- ⑤ Network Status LED NS
- ⑥ Module Status LED MS

**EASY222-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 easy 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 easy control relay to DeviceNet the PLC.
 - The **explicit connection set-up** is used for read/write access to function relay parameters in the easy 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 easy 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

**Hardware and Operating
system requirements**

The expansion module EASY222-DN operates in combination with the basic units easy619.. and easy621.. as of operating system V2.4.

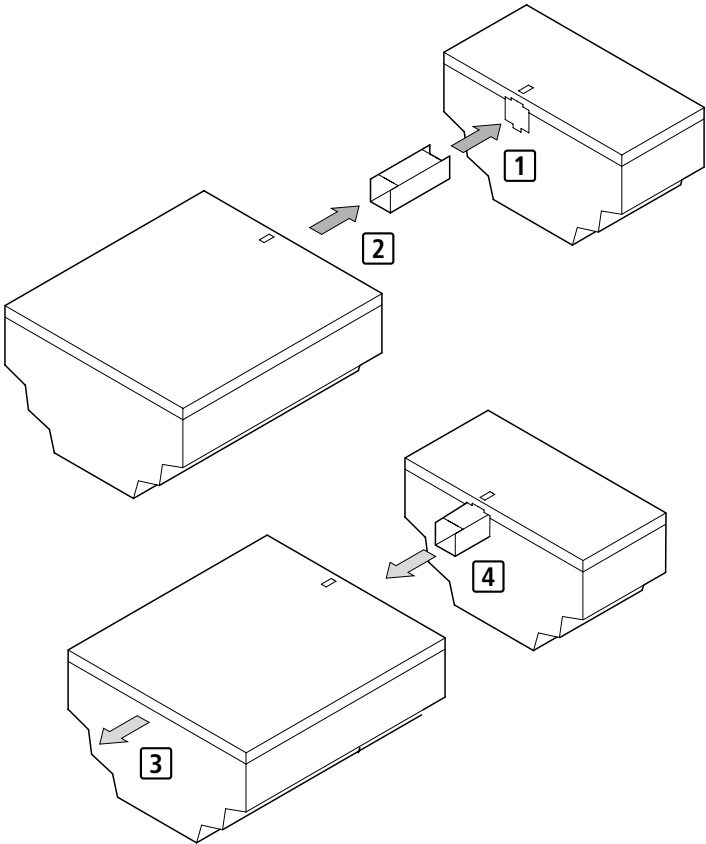
Use other than intended easy may not be used to replace safety-relevant control circuits such as:

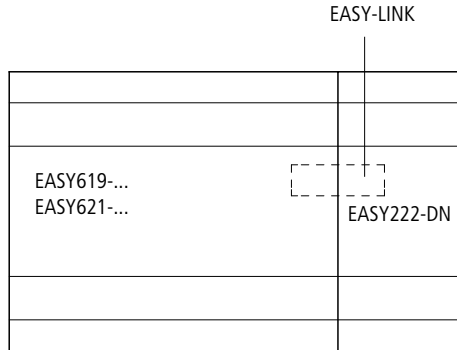
- furnace,
- emergency-stop,
- crane or
- two-hand safety controls.

2 Installation

Applicable are the same guidelines as for easy600 basic units with expansion modules.

Connection of EASY222-DN to the basic unit





Connecting the power supply

The device EASY222-DN operates with a 24 VDC supply voltage (→ section "Power supply", Page 93).



Warning!

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

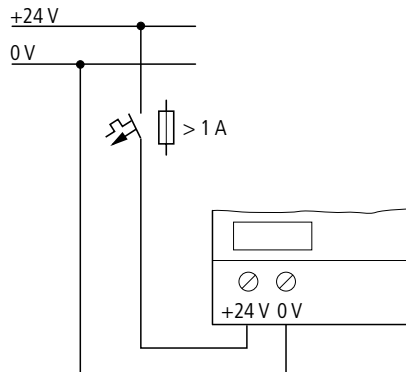


Figure 3: Power supply EASY222-DN

DeviceNet connection

A 5-pole DeviceNet plug connects the DeviceNet interface of the device 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.

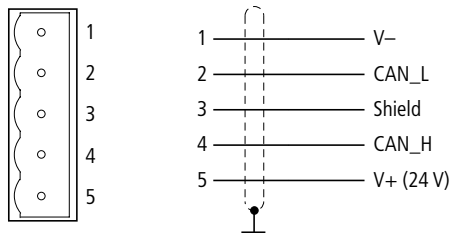
Terminal assignment DeviceNet

Figure 4: 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 EASY222-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 station 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.

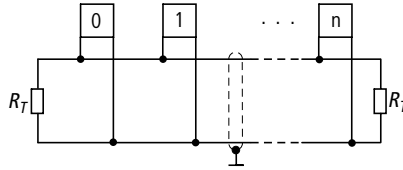


Figure 5: Terminating resistors R_T : CAN_H and CAN_L terminals
 $R_T = 120 \Omega$

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.

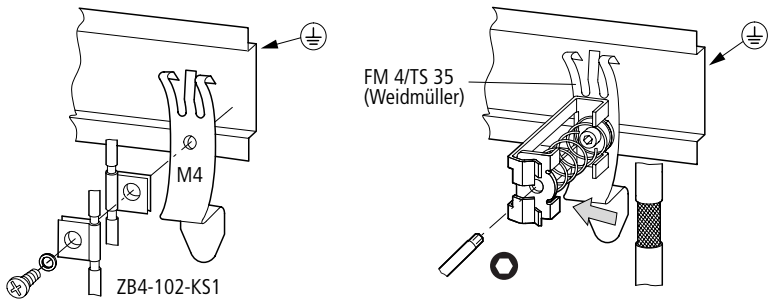


Figure 6: Shielding connection to the mounting rail

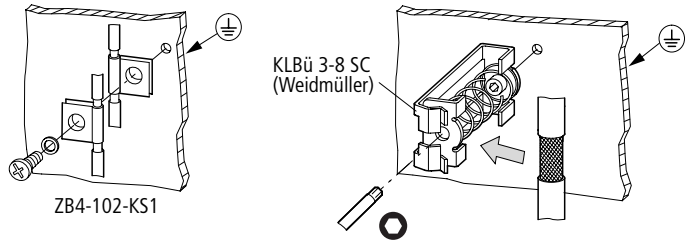


Figure 7: Shielding connection to the mounting plate

Potential isolation

The following potential isolation specifications apply to the interfaces of the EASY222-DN:

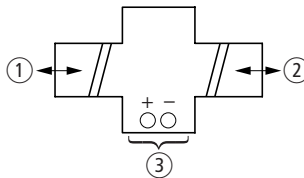


Figure 8: Potential isolation between the supply voltage and outputs

- ① Safe electrical isolation between EASY-LINK and the 240 VAC mains
- ② Simple electrical isolation to the DeviceNet communication bus
- ③ Power supply 24 VDC

Transmission rates – Automatic recognition of the baud rate

After it is switched on, the EASY222-DN module automatically recognises the baud 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 Device operation

Initial power on

Before you switch on the unit, 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 EASY222-DN.

If the EASY222-DN has factory settings, you need to define the DeviceNet node address.

DeviceNet Slave address setting

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 EASY222-DN:

- Using the integrated display and keyboard on the easy basic unit
- Using EASY-SOFT V3.01 or higher 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

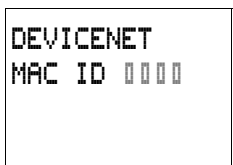
- The basic unit easy600 and EASY222-DN are connected to the power supply.
- The basic unit is not password-protected.
- The operating system version 2.4 or higher is installed on the basic unit .
- The basic unit is in STOP mode.
- No active communication between the EASY222-DN and the DeviceNet master.



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

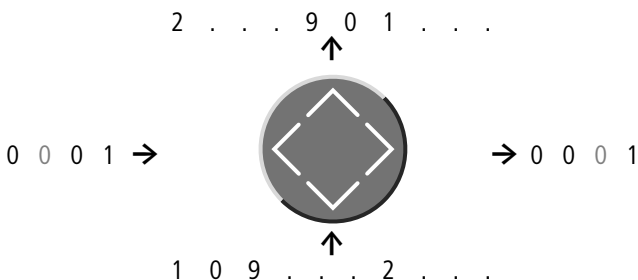
▶ Use the cursor keys ^ or v to change to the KONFIGURATOR.

▶ Confirm with OK.



The DEVICENET menu appears.

- ▶ Set the address by means of the cursor keys:
 - Set the current numeric value via ^ or v.
 - You can change the current numeric value via < or >.





▶ Accept the address with OK.



▶ Cancel address input.

Setting the address by means of EASY-SOFT

EASY-SOFT V3.1 or higher:

Menu → Online → Configuration of expansion units

After you have modified the MAC ID via the basic unit, restart the EASY222-DN by switching power off and on.

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.

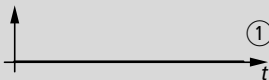
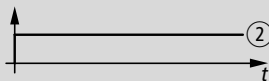


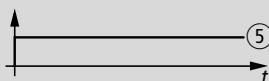
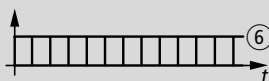
You can also use various other software packages to modify the MAC ID, e.g. by sending an explicit message. Do so by using the corresponding service of the DeviceNet object (Section "DeviceNet object", Page 32).

LED Status displays

The expansion module EASY222-DN is equipped with two indicator LEDs for quick diagnostics. EASY222-DN monitors itself as well as the DeviceNet communication bus.

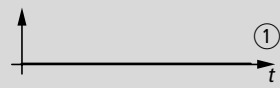
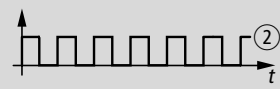

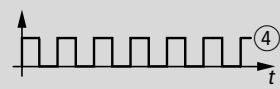
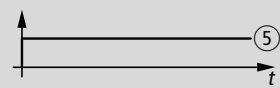
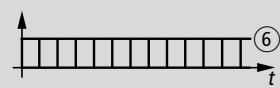
Module status LED

The dual-color LED (GREEN/RED) indicates the status of EASY222-DN. It monitors whether the device is fully functional and operates without fault.

OFF	No power supply at the EASY222-DN.	
GREEN	EASY222-DN is in normal operational state.	
GREEN flashing	EASY222-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 EASY222-DN.	
RED	A fatal error has occurred EASY222-DN. EASY222-DN must be replaced.	
GREEN-RED flashing	EASY222-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 EASY222-DN.

OFF	EASY222-DN is offline. Either it is performing a DUP_MAC_ID test or power is missing at the device or bus.	
GREEN Flashing	EASY222-DN is online. Communication has not yet been established.	
GREEN	EASY222-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. EASY222-DN has shut down communication.	
GREEN-RED flashing	EASY222-DN has detected a network access error and is now in communication error state.	

Cycle time easy basic unit Network traffic between the easy600 basic unit and the EASY222-DN via EASY-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.

4 DeviceNet functions

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 exist 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 between the 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 (easy basic unit) are processed by means of explicit messages. The master PLC can thus read/write access the parameters of the following functions.

- Timing relays
- Timer relays
- Switching timers
- Analogue value comparators
- The weekday, time-of-day, summer/winter time

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
easy600 output range, RUN/STOP (inputs at the DeviceNet master)
- 3 bytes of input data
R1 to R16
easy600 input range, RUN/STOP (outputs of the DeviceNet master)

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 bytes of diagnostics data of the easy control relay coupling module status

Cyclic I/O connection

Message triggering is timer-controlled when operating with a cyclic I/O connection.

UCMM

The DeviceNet gateway provides an option of configuring dynamic connection objects via the UCMM port (Unconnected Message Manager Port).

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 initialised 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.

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 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.

EDS file

You can implement EASY222-DN into the DeviceNet structure by means of a standardised 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 (easy object) for functions of the easy basic unit. These functions are accessed by means of explicit messages.

You can either order the current version of the EDS file directly at Moeller or download updates of this file from the Moeller homepage:

<http://easy.moeller.net> → Download → ...

Follow the "Link" on this page.

A printed version of the EDS file can be found in the annex (→ section "EDS file", Page 95).

Object Model

EASY222-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 EASY222-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 easy object.

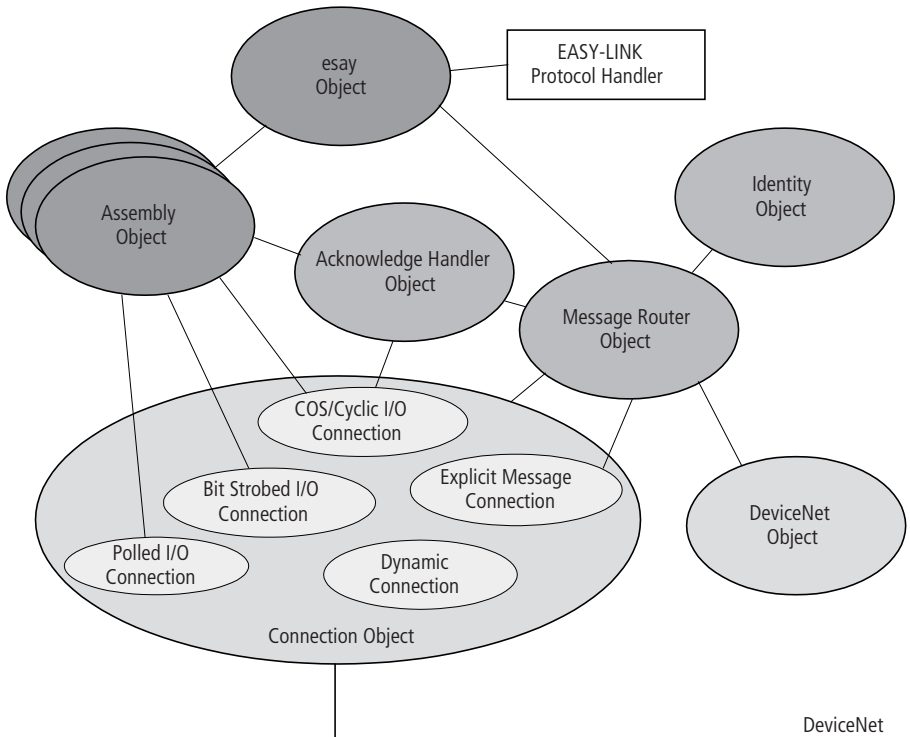


Figure 9: DeviceNet objects

Object class	Class ID [DEC]	Instance ID [DEC]	Messages
Standard DeviceNet objects			
Identity Object	1	1	Explicit
Message Router Object	2	1	Explicit
DeviceNet Object	3	1	Explicit
Assembly Object	4	100 to 102	Explicit & I/O
Connection Object	5	1 to 4, 5 to 14	Explicit
Acknowledge Handler Object	43	1	Explicit
Manufacturer-specific DeviceNet Object			
easy Object	100	1	Explicit

The objects shown in Figure9 can be split into three groups:

- Management objects
- Connection objects
- Application-specific objects

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.

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.

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.

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.

Application-specific objects

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

Application Objects – easy 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.

Assembly Objects

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

Identity Object

Class ID: 01_{hex}, Instance ID: 01_{hex}

Attribute ID	Access	Name	Description	Size [byte]
1	Read	Vendor ID	The ODVA specifies the Vendor ID. For Moeller GmbH, this is 248 _{dec} .	2
2	Read	Device type	The EASY222-DN belongs to the communication adapters category. Its value is 12 _{dec} .	2
3	Read	Product code	The product code is defined by Moeller: 650 _{dec} . It describes the model number.	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 EASY222-DN is stored as hex value in ASCII format.	12
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

Service code

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

Service code value	Service name	Description
05 _{hex}	Reset	Calls the reset function of the communication module EASY222-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

Class ID: 03_{hex}, Instance ID: 01_{hex}

The DeviceNet object instance is used to configure the communication module EASY222-DN and to define the physical environment. The Service Codes used for the Identity Object also apply in this case.

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 EASY222-DN via the fieldbus by means of this attribute. Range of values: 0 to 63 _{dec} . (→ section "DeviceNet Slave address setting", Page 17)	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 "Transmission rates – Automatic recognition of the baud rate", Page 16).	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 values shows how often a Bus-Off event has occurred. Range of values: 0 to 255.	1

easy object

Class ID: 64_{hex}, Instance ID: 01_{hex}

The easy object can be used to access easy600 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 Networks), you can map these data directly to the corresponding memory areas of a PLC.

Attribute ID	Access	Name	Description	Size [byte]
1	Read	easy Status	This attribute can be used to read the status of easy (RUN or STOP).	1
2	Read	Coupling Module Status	This attribute can be used to read the status of EASY-LINK.	1
3	Read	Inputs – Send Data	easy transfers the input data to the DeviceNet bus. The easy outputs S1 to S8 must be used for this function. The structure of these 3 bytes is described in detail under Section "Input data", Page 35.	3
4	Read/Write	Outputs – Receive Data	The DeviceNet bus transfers the data to easy. The easy inputs R1 to R16 must be used for this function. The structure of these 3 bytes is described in detail under Section "Output data", Page 37.	3
5	Read/Write	Predefined Outputs	This attribute can be used to preset the output data ("R" data) at the EASY222-DN during start-up. The structure of these 3 bytes is described in detail under Section "Output data", Page 37.	3

Service code

The easy object instance supports the following services.

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 the value of a selected attribute to the device.
32 _{hex}	Extended access	This service can be used to address the supplementary parameters of the control relay: <ul style="list-style-type: none"> • Time-of-day easy600, • Timing relays • Timer relays, • Switching timers, • Actual/setpoint value and • Analogue value comparators.

An explicit message transfer is used for extended access to easy-specific parameters. This transfer protocol allows the exchange of control data.

Change of State I/O connection

Diagnostics data: 2 Byte

Byte	Meaning	Value	Meaning
0	easy status (attribute ID 1)	00 _{hex}	Static value.
1	Coupling module status (attribute ID 2)	00 _{hex}	The basic unit is connected to the EASY222-DN gateway via EASY-LINK.
		04 _{hex}	The basic unit is either switched off or disconnected from the EASY222-DN gateway via EASY-LINK.



When communication between the basic unit easy600 and the expansion unit EASY222-DN goes down, a corresponding error code will be generated in the third data byte. Furthermore, the value 00_{hex} will be set in the R/S data transferred to the gateway.

Polled I/O connection**Input data**

Attribute ID: 3

The input data for normal data exchange between the DeviceNet master and the EASY222-DN slave are formed by the bytes 0, 1 and 2.

Byte	Meaning
0	Mode
1	Status of the easy outputs S1 to S8
2	Not used

Data input area (easy outputs S1 to S8, operating mode)

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

Byte 0: Mode

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

0 = status "0"
 1 = status "1"

Example:
 Value 21_{hex} = 00100001_{bin}:
 easy is in RUN mode and operates with input delay

Byte 1: Status of easy outputs S1 to S8

easy600 output	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} = 0001 1001_{bin}:
 S5, S4 and S1 are active

Byte 2: not used



If control commands and I/O data are used concurrently:

- The inputs retain their previous state until the 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

Attribute ID: 4

The output data for normal data exchange between the DeviceNet master and the EASY222-DN slave are formed by the bytes 0, 1 and 2.

Byte	Meaning
0	Mode
1	Status of the easy inputs R9 to R16
2	Status of the easy inputs R1 to R8

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

Byte 0: Mode

easy 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"

Description:

Value $14_{\text{hex}} = 00010100_{\text{bin}}$:

Byte 0 must always contain this value if data are to be written to the easy600 basic unit via the EASY222-DN gateway.

Value $34_{\text{hex}} = 00110100_{\text{bin}}$:

This value sets the easy status from STOP to RUN. It is interpreted only as command and thus does not permit an additional transfer of data. The index value 14_{hex} must be used in this situation.

Value $44_{\text{hex}} = 01000100_{\text{bin}}$:

This value sets the easy status from RUN to STOP. It is used only as instruction and is thus based on the same operating principle as the RUN command.

Value $00_{\text{hex}} = 00000000_{\text{bin}}$:

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}).

Byte 1: Status easy600 inputs R9 to R16

easy600 output	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} = 00011001_{bin}:

Enable R13, R12 and R9.

Byte 2: Status easy600 inputs R1 to R8

easy600 output	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 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.

Data exchange method Control commands

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

- Time-of-day easy600,
- Timing relays
- Counter relays,
- Switching timers,
- Process values/reference values,
- Process values/reference values of analogue value comparators,
- Status of inputs, outputs, contactor relays, function relays



A DeviceNet connection of the easy control relay to an SLC 500 requires specific control and handshake routines in the PLC program for the execution of the following control commands.

The application note AN2700K17D supports the control commands of EASY222-DN. It provides subroutines in the program for controlling the required "Explicit Messages", i.e. programming will be replaced by the call and the parameter assignment of the subroutine. Parameters are assigned by means of an integer file.

The self-extracting application note AN2700K17D.exe is available for download on the Moeller server.

<ftp://ftp.moeller.net> → AUTOMATION → APPLICATION_NOTES → an27k17d.exe

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. The attribute value and the control byte are identical.



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 unless 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 method needs to be defined in order to ensure safe data exchange between the master and slave stations.



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.

Overview

The tables below provide an overview of addresses available at the easy basic unit and of the corresponding addressing. The chapter below describes the detailed coding of data to be transferred.

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

Attribute ID	Action	Operand	Size [byte]	Page
01 _{hex} to 08 _{hex}	Write	Time parameters T1 to T8	3	44
09 _{hex} to 10 _{hex}	Write	Counter parameters C1 to C8	3	47
11 _{hex}	Write	Switching timer control bytes	4	49
12 _{hex} to 21 _{hex}	Write	Switching timers 1 to 4: channel A to D	5	51
22 _{hex} to 29 _{hex}	Write	Analogue value comparator 1 to 8	2	54
2A _{hex}	Write	Real-time clock	4	57
2B _{hex} to 32 _{hex}	Read	Timing relays 1 to 8: actual values	5	59
33 _{hex} to 3A _{hex}	Read	Timing relays 1 to 8: setpoint values	4	62
3B _{hex} to 42 _{hex}	Read	Counter relays 1 to 8: actual values	4	64
43 _{hex} to 4A _{hex}	Read	Counter relays 1 to 8: setpoint values	4	66
4B _{hex} to 5A _{hex}	Read	Switching timers 1 to 4: channel A to D	7	68
5B _{hex}	Read	Analogue value inputs	3	73
5C _{hex}	Read	Status of the digital inputs, P buttons and operator control keys	4	74
5D _{hex}	Read	Real-time clock	5	77
5E _{hex}	Read	Status of the timing relay, counter relay, switching timers and analogue value comparators	5	79
5F _{hex}	Read	Status of the contactor relay (marker), text display and digital outputs	5	84

Timing relays T1 to T8: Setting parameters

Attribute ID	Action
01 _{hex}	Set parameters for timing relay T1
02 _{hex}	Set parameters for timing relay T2
03 _{hex}	Set parameters for timing relay T3
04 _{hex}	Set parameters for timing relay T4
05 _{hex}	Set parameters for timing relay T5
06 _{hex}	Set parameters for timing relay T6
07 _{hex}	Set parameters for timing relay T7
08 _{hex}	Set parameters for timing relay T8

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Control byte	Low reference value	High reference value	–	–	–	–
Slave	Answer byte	00	00	00	00	00	00

Control byte: Setting the switching function of the timing relay

Meaning	Bit							
	7	6	5	4	3	2	1	0
On-delayed,						0	0	0
off-delayed.						0	0	1
on-delayed with random switching,						0	1	0
Off-delayed with random switching,						0	1	1
Pulse shaping						1	0	0
Flashing						1	0	1
Timebase in [ms]				0	0			
Timebase in [s]				0	1			
Timebase in [min]				1	0			

Meaning	Bit							
	7	6	5	4	3	2	1	0
Not used			0					
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution	1							

Example:

Value $89_{\text{hex}} = 10001001_{\text{bin}}$

Timing relay operates with off-delay, timebase in [s].

Timing relays: Setting the reference value (byte 1 and byte 2)

Bytes 1 and 2 determine the reference value for the timing relay. The reference value is based on the selected timebase. When the control byte is set to seconds, the low value is based on seconds and the high value on the next higher timebase (minute). The value range for each byte in this case is 0 to 59_{dec} ($3B_{\text{hex}}$). The table below results:

Timebase	Low value	High value
milliseconds	0 to 59 (10 ms)	0 to 59 s
seconds	0 to 59 s	0 to 59 min
minute	0 to 59 min	0 to 59 h

Example:

Low value 11_{hex} : Equivalent to 17 s, timebase in [s]

high value $2D_{\text{hex}}$: Equivalent to 45 min, timebase in [s]

Answer byte

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of the "set

time reference value X" command. Otherwise, if write access was not possible easy rejects this action.

Meaning	Bit							
	7	6	5	4	3	2	1	0
fixed	0							
fixed		1	0					
EASY222-DNwrite request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:
 Value 41_{hex} = 0100001_{bin}:
 The last service was executed.



If a specified reference value lies out of the permissible range, EASY222-DN will generate an error message.

Counter relays C1 to C8: Setting parameters

Attribute ID	Action
09 _{hex}	Set counter parameter C1
0A _{hex}	Set counter parameter C2
0B _{hex}	Set counter parameter C3
0C _{hex}	Set counter parameter C4
0D _{hex}	Set counter parameter C5
0E _{hex}	Set counter parameter C6
0F _{hex}	Set counter parameter C7
10 _{hex}	Set counter parameter C8

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Control byte	Low reference value	High reference value	–	–	–	–
Slave	Answer byte	00	00	00	00	00	00

Control byte:

Meaning	Bit							
	7	6	5	4	3	2	1	0
Not used			0	0	0	0	0	0
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution	1							

Example:
 Value 80_{hex} = 1000000_{bin}:
 The reference value will be written to the selected timing relay and appears in the parameter menu.

Setting the reference value (byte 1 and byte 2)

These two bytes determine the reference value of the counter relay. The reference value can be set within the range from 0 to 9999_{dec}. Here you need to convert the required decimal into the equivalent hexadecimal value and then split it up into the low-byte and high-byte.

Example:

Reference value = 4318_{dec} = 10DE_{hex}:

Low-value: DE_{hex}

High-value: 10_{hex}

Answer byte

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of the "set time reference value X" command.

Otherwise, if write access was not possible easy rejects this action.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN write request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:

Value 41_{hex} = 01000001_{bin}:

The last service was executed.



If a specified reference value lies out of the permissible range, EASY222-DN will generate an error message.

Switching timers: Setting the control bytes

Attribute ID	Action
11 _{hex}	Set the switching timer

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Control byte switching timer 1	Control byte switching timer 2	Control byte switching timer 3	Control byte switching timer 4	–	–	–
Slave	Answer byte	00	00	00	00	00	00

Control byte switching timer 1 to 4

Each one of the four switching timers of the easy basic unit can be enabled by means of a specifically assigned control byte. The structure of the control byte is shown below:

Meaning	Bit							
	7	6	5	4	3	2	1	0
The status of the control byte will be changed					0	0	0	1
The status of the control byte will not be changed					0	0	0	0
Status:do not include timer relay X in the circuit diagram	0	0	0	0				
Status: include timer relay X in the circuit diagram	1	0	0	0				

It is not necessarily required to set all four control bytes. The last bit of each control bytes determines whether the switching timer is used or not.

Examples

Byte	Value		Result
	hex	bin	
0	00	0000 0000	The status of switching timer 1 is not changed.
1	01	0000 0001	The status of switching timer 2 is changed to "not enabled".
2	80	10000000	The status of switching timer 3 is not changed.
3	81	10000001	Switching timer 4 is enabled in the circuit diagram.



Changes are not accepted in the circuit diagram until the corresponding switching timer is called again. For example, if a daily timer is active between 12.00 and 13.00 p.m., status changes made within this time will not be accepted until the timer has become inactive again, i.e either before 12.00 o'clock midday or after 13.00 p.m.

Answer byte

The first byte returned by the slave contains the acknowledgement. easy" confirms the execution of the "set time reference value X" command.

Otherwise, easy rejects this action if write access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
fixed	0							
fixed		1	0					
EASY222-DN write request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:

Value 41_{hex} = 01000001_{bin}:

The last service was executed.



At least one switching timer must be addressed by the call of this attribute. Otherwise EASY222-DN generates an error message. Furthermore, the addressed switching timer must exist in the circuit diagram of the basic unit.

Switching timers 1 to 4: Setting the channels A to D

Attribute ID	Action
12 _{hex}	Set channel A of switching timer 1
13 _{hex}	Set channel B of switching timer 1
14 _{hex}	Set channel C of switching timer 1
15 _{hex}	Set channel D of switching timer 1
16 _{hex}	Set channel A of switching timer 2
17 _{hex}	Set channel B of switching timer 2
18 _{hex}	Set channel C of switching timer 2
19 _{hex}	Set channel D of switching timer 2
1A _{hex}	Set channel A of switching timer 3
1B _{hex}	Set channel B of switching timer 3
1C _{hex}	Set channel C of switching timer 3
1D _{hex}	Set channel D of switching timer 3
1E _{hex}	Set channel A of switching timer 4
1F _{hex}	Set channel B of switching timer 4
20 _{hex}	Set channel C of switching timer 4
21 _{hex}	Set channel D of switching timer 4

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Control byte: Start of day End of day	Minute ON	Hour ON	Minute OFF	Hour OFF	–	–
Slave	Answer byte	00	00	00	00	00	00

Control byte (Weekday: starting/ending, parameter menu display)

Each channel of a weekly timer is assigned a control byte that defines the start/stop conditions. The table below shows the precise structure of this control byte.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Day ON								
no day set						0	0	0
Monday						0	0	1
Tuesday						0	1	0
Wednesday						0	1	1
Thursday						1	0	0
Friday						1	0	1
Saturday						1	1	0
Sunday						1	1	1
Day OFF								
no day set			0	0	0			
Monday			0	0	1			
Tuesday			0	1	0			
Wednesday			0	1	1			
Thursday			1	0	0			
Friday			1	0	1			
Saturday			1	1	0			
Sunday			1	1	1			
Appears in the parameter menu								
No	1	0						
Yes	0	0						

Example:
Value 31_{hex} = 00110001_{bin}:

The previously selected channel X of weekly timer Y is active Monday through Saturday.

Setting the ON and OFF times (byte 1 to byte 4)

The table below shows the bytes which determine the precise ON and OFF times of a channel. The resolution is in seconds.

ON time		OFF time	
Byte 1:	Byte 2:	Byte 3:	Byte 4:
Minute ON	Hour ON	Minute OFF	Hour OFF
00 to 3B _{hex} (00 to 59 _{dec})	00 to 17 _{hex} (00 to 23 _{dec})	00 to 3B _{hex} (00 to 59 _{dec})	00 to 17 _{hex} (00 to 23 _{dec})



You must convert all decimal values into hexadecimal format and enter them accordingly.

Example:

Description	Attribute/Byte	Value
Data of channel A of switching timer 4:	Attribute ID	1E _{hex}
Day: Monday through Saturday The channel appears in the parameter menu	Byte 0:	31 _{hex} (see above)
ON 19:00	Byte 1:	00 _{hex}
	Byte 2:	13 _{hex}
OFF: 06:30	Byte 3:	1E _{hex}
	Byte 4:	06 _{hex}

Answer byte

The first byte returned by the slave contains the acknowledgement. easy confirms execution of the "set time reference value X" command.

Otherwise, easy rejects this action if write access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN write request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:

Value 41_{hex} = 0100001_{bin}:

The last service was executed.

Setting the analogue value comparators 1 to 8

The required analogue value for the comparators 1 to 8 can be selected via the attribute ID from the table below.

Attribute ID	Action
22 _{hex}	Set analogue value comparator 1
23 _{hex}	Set analogue value comparator 2
24 _{hex}	Set analogue value comparator 3
25 _{hex}	Set analogue value comparator 4
26 _{hex}	Set analogue value comparator 5
27 _{hex}	Set analogue value comparator 6
28 _{hex}	Set analogue value comparator 7
29 _{hex}	Set analogue value comparator 8

For the analogue value comparator you must furthermore define two additional bytes and write them to the easy basic unit.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Control byte:	(reference value) Constant	–	–	–	–	–
Slave	Answer byte	00	00	00	00	00	00

Control byte:

Meaning	Bit							
	7	6	5	4	3	2	1	0
Comparison								
\cong								0
\cong								1
I7 to I8						0	0	
I7 to constant (byte 1)						0	1	
I8 to constant (byte 1)						1	0	
Fixed			0	0	0			
Appears in the parameter menu								
No		1						
Yes		0						
Execution	1							

Example:

$82_{\text{hex}} = 1000010_{\text{bin}}$ means that the selected analogue value comparator will be enabled in the circuit diagram of the basic unit as soon as the analogue value input I7 \cong the defined constant (\rightarrow byte 1).

Reference value (byte 1)

The second byte contains the reference value constant. It has a value between 0 and 99 and is equivalent to a reference voltage of 0.0 to 9.9 V. You must also specify this value in hexadecimal format.

Example:

A reference value = 20_{hex} is equivalent to an analogue voltage of 3.2 V.

Answer byte

The first byte received from the slave contains the acknowledgement. easy confirms the execution of a "set analogue value comparator" command.

Otherwise, easy rejects this action if write access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN write request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:

Value 41_{hex} = 01000001_{bin}:

The last service was executed.

Setting the real-time clock

The real-time clock of control relay easy600 can be set by means of the following attributes and parameters described.

Attribute ID	Action
2A _{hex}	Set real-time clock

The real-time clock of easy600 requires the following parameters. The seconds value can not be configured and is therefore automatically set to 00 when you set the clock.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	Weekday	Hour	Minute	Summer/ winter time	–	–	–
Slave	Answer byte	00	00	00	00	00	00

Weekday (byte 0)

Range of values: 00 to 06.
This byte indicates the weekday by means of a numerical value between 0 (Monday) and 6 (Sunday).

Hour (byte 1)

range of values: 00 to 23_{dec} = 00 to 17_{hex}.
This byte is used to fetch the current time of the real-time clock of the basic unit. Here you also need to convert the range of values into hexadecimal format.

Minute (Byte 2)

Range of values: 00 to 59_{dec} = 00 to 3B_{hex}.
This byte is used to acquire minute value that is set at the real-time clock of the basic unit. Here you also need to convert the range of values into hexadecimal format.

Summer/winter time (byte 3)

Range of values: 00 to 01

00 = winter time or

01 = summer time.

Example:

It is Friday, the current time-of-day is 14:36 p.m. CET summer time.

Byte	0	1	2	3	4	5	6
Hex value	04	0E	24	01	–	–	–
Meaning	Friday	14	36	Summer time	–	–	–

Answer byte

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN write request rejected due to error				0	0	0	0	0
Write request OK				0	0	0	0	1

Example:

Value 41_{hex} = 01000001_{bin}:

The last service was executed.

Timing relays 1 to 8: Reading the process variables

The following attributes can be used to read the process variables of the various timing relays.

Attribute ID	Action
2B _{hex}	Read process variable of timing relay 1
2C _{hex}	Read process variable of timing relay 2
2D _{hex}	Read process variable of timing relay 3
2E _{hex}	Read process variable of timing relay 4
2F _{hex}	Read process variable of timing relay 5
30 _{hex}	Read process variable of timing relay 6
31 _{hex}	Read process variable of timing relay 7
32 _{hex}	Read process variable of timing relay 8

After the read request has been triggered only by sending the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Control byte:	Process variable Low value	Process variable High value	Random value	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:

Value 42_{hex} = 01000010_{bin}:

Read request OK, data follow.

Reading the switching function (byte 1)

	7	6	5	4	3	2	1	0
On-delayed,						0	0	0
Off-delayed.						0	0	1
On-delayed with random switching						0	1	0
Off-delayed with random switching						0	1	1
Pulse shaping						1	0	0
Flashing						1	0	1
Timebase in [ms]				0	0			
Timebase in [s]				0	1			
Timebase in [min]				1	0			
Not used			0					
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution	1							

Example:

Value 89_{hex} = 10001001_{bin}:

Timing relay operates with off-delay and seconds timebase.

Process variable (byte 2 and byte 3)

These two bytes define the process variable of the timing relay. This process variable depends on the set timebase. When the control byte is set to a seconds timebase, the low-value represents the SECONDS and the high-value the MINUTES. The maximum range of return values for each byte is 0 to 59_{dec} (3B_{hex}). The table below shows the result:

Timebase	Low-value	High-value
millisecond	0 to 59 (10 ms)	0 to 59 s
seconds	0 to 59 s	0 to 59 min
minutes	0 to 59 min	0 to 59 h

Example:

Low value 11_{hex}: Equivalent to 17 s, timebase in [s].

High value 2D_{hex}: Equivalent to 45 min, timebase in [s]

Random value (byte 4)

easy sets a random delay time between zero and the set reference time for relays operating with random switching characteristics. This reference time is specified at this byte in hexadecimal format.

Timing relays 1 to 8: Reading the reference values

The following attributes can be used to read the reference values of the various timing relays.

Attribute ID	Action
33 _{hex}	Read reference value of timing relay 1
34 _{hex}	Read reference value of timing relay 2
35 _{hex}	Read reference value of timing relay 3
36 _{hex}	Read reference value of timing relay 4
37 _{hex}	Read reference value of timing relay 5
38 _{hex}	Read reference value of timing relay 6
39 _{hex}	Read reference value of timing relay 7
3A _{hex}	Read reference value of timing relay 8

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Control byte:	Reference value Low value	Reference value High value	00	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:
 Value 42_{hex} = 01000010_{bin}:
 Read request OK, data follow.

Control byte (byte 1)

Meaning	Bit							
	7	6	5	4	3	2	1	0
On-delayed						0	0	0
Off-delayed						0	0	1
On-delayed with random switching						0	1	0
Off-delayed with random switching						0	1	1
Pulse shaping						1	0	0
Flashing						1	0	1
Timebase in [ms]				0	0			
Timebase in [s]				0	1			
Timebase in [min]				1	0			
Not used			0					
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution	1							

Example:
 Value 89_{hex} = 10001001_{bin}:
 Timing relay operates with off-delay and seconds timebase.

Reference value (byte 2 and byte 3)

These two bytes define the reference value of the timing relay. This reference value depends on the selected timebase. When the control byte is set to a seconds timebase, the low value represents the SECONDS and the high value the MINUTES. The maximum range of return values for each byte is 0 to 59_{dec} (3B_{hex}). The table below shows the result:

Timebase	Low-value	High-value
milliseconds	0 to 59 (10 ms)	0 to 59 s
seconds	0 to 59 s	0 to 59 min
minute	0 to 59 min	0 to 59 h

Example:

The low value_{hex} is equivalent to 17 s at a seconds timebase

The high value 2D_{hex} is equivalent to 45 min at a seconds timebase

Counter relays 1 to 8: Reading the process variables

The following attributes can be used to read the process variables of the various counter relays.

Attribute ID	Action
3B _{hex}	Read process variable of counter relay 1
3C _{hex}	Read process variable of counter relay 2
3D _{hex}	Read process variable of counter relay 3
3E _{hex}	Read process variable of counter relay 4
3F _{hex}	Read process variable of counter relay 5
40 _{hex}	Read process variable of counter relay 6
41 _{hex}	Read process variable of counter relay 7
42 _{hex}	Read process variable of counter relay 8

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Control byte	Process variable Low value	Process variable High value	00	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:

Value 40_{hex} = 01000000_{bin}:

Read request error, no data follow.

Control byte (byte 1)

Meaning	Bit							
	7	6	5	4	3	2	1	0
Not used			0	0	0	0	0	0
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution (will be processed in the circuit diagram)	1							

Example:

Value $80_{hex} = 10000000_{bin}$:

The actual value of the counter relay is set and appears in the parameter menu.

Process variable (byte 2 and byte 3)

These two bytes define the process variable of the counter relay. The value of the process variable can lie within the range 0 to 9999_{dec} . In order to determine the corresponding process variable, you need to convert the 16-bit hexadecimal low and high values into the decimal format.

Example:

High value: 10_{hex}

Low value: DE_{hex}

$10DE_{hex} = 4318_{dec}$

Counter relays 1 to 8: Reading the reference values

The following attributes can be used to read the reference values of the various counter relays.

Attribute ID	Action
43_{hex}	Read reference value of counter relay 1
44_{hex}	Read reference value of counter relay 2
45_{hex}	Read reference value of counter relay 3
46_{hex}	Read reference value of counter relay 4

Attribute ID	Action
47 _{hex}	Read reference value of counter relay 5
48 _{hex}	Read reference value of counter relay 6
49 _{hex}	Read reference value of counter relay 7
4A _{hex}	Read reference value of counter relay 8

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Control byte:	Reference value Low-value	Reference value High-value	00	00	00

Answer byte (byte 0)

The first byte received from the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:
 Value 40_{hex} = 01000000_{bin}:
 Read request error, no data follow.

Control byte (byte 1)

Meaning	Bit							
	7	6	5	4	3	2	1	0
Not used			0	0	0	0	0	0
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Execution (is being processed in the circuit diagram)	1							

Example:

Value $80_{hex} = 10000000_{bin}$:

The reference value of the counter relay is set and appears in the parameter menu.

Reference value (byte 2 and byte 3)

These two bytes define the reference value of the counter relay. The reference value can lie within the value range 0 to 9999_{dec} . In order to determine the corresponding reference value, you need to convert the 16-bit hexadecimal low and high value into the decimal format.

Example:

High value: 10_{hex}

Low value: DE_{hex}

$10DE_{hex} = 4318_{dec}$

Switching timers 1 to 4: Reading channel A to D

The following attributes can be used to read the configuration data of specific switching timers and of their channels.

Attribute ID	Action
$4B_{hex}$	Read channel A of switching timer 1
$4C_{hex}$	Read channel B of switching timer 1
$4D_{hex}$	Read channel C of switching timer 1
$4E_{hex}$	Read channel D of switching timer 1

Attribute ID	Action
4F _{hex}	Read channel A of switching timer 2
50 _{hex}	Read channel B of switching timer 2
51 _{hex}	Read channel C of switching timer 2
52 _{hex}	Read channel D of switching timer 2
53 _{hex}	Read channel A of switching timer 3
54 _{hex}	Read channel B of switching timer 3
55 _{hex}	Read channel C of switching timer 3
56 _{hex}	Read channel D of switching timer 3
57 _{hex}	Read channel A of switching timer 4
58 _{hex}	Read channel B of switching timer 4
59 _{hex}	Read channel C of switching timer 4
5A _{hex}	Read channel D of switching timer 4

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Control byte: Switching timer	Control byte: Channel	Minute ON	Hour ON	Minute OFF	Hour OFF

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read

easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:
 Value 42_{hex} = 01000010_{bin}:
 Read request OK, data follow.

Control byte switching timer

Meaning	Bit							
	7	6	5	4	3	2	1	0
Not being processed	0	0	0	0	0	0	0	0
Execution (is being processed in the circuit diagram)	1	0	0	0	0	0	0	0

Example:
 Value 80_{hex} = 10000000_{bin}:
 The addressed switching timer is used in the circuit diagram.

Control byte channel

(Weekday: starting/ending, parameter menu display)
 Each channel of a weekly switching timer is assigned a

control byte that defines the start/stop conditions. The table below shows the precise structure of this control byte.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Day ON								
No day set						0	0	0
Monday						0	0	1
Tuesday						0	1	0
Wednesday						0	1	1
Thursday						1	0	0
Friday						1	0	1
Saturday						1	1	0
Sunday						1	1	1
Day OFF								
no day set			0	0	0			
Monday			0	0	1			
Tuesday			0	1	0			
Wednesday			0	1	1			
Thursday			1	0	0			
Friday			1	0	1			
Saturday			1	1	0			
Sunday			1	1	1			
Appears in the parameter menu								
No	1	0						
Yes	0	0						

Example:
 Value 31_{hex} = 00110001_{bin}:
 The previously selected channel X of weekly timer Y is active Monday through Saturday.

ON/OFF times (byte 3 to byte 6)

The table below shows the bytes which determine the precise ON and OFF times of a channel. The resolution is in seconds.

ON time		OFF time	
Byte 3:	Byte 4:	Byte 5:	Byte 6:
Minute ON	Hour ON	Minute OFF	Hour OFF
00 to 3B _{hex} (00 to 59 _{dec})	00 to 17 _{hex} (00 to 23 _{dec})	00 to 3B _{hex} (00 to 59 _{dec})	00 to 17 _{hex} (00 to 23 _{dec})



easy returns hexadecimal values. You may have to convert the corresponding values into decimal format.

Example:

Byte	Value	Description
0	42 _{hex}	The read request has been executed. Data follow.
1	80 _{hex}	The addressed switching timer is used in the circuit diagram.
2	31 _{hex} (see above)	Day: Monday through Saturday The channel appears in the parameter menu
3	00 _{hex}	ON 19:00
4	13 _{hex}	
5	1E _{hex}	OFF: 06:30
6	06 _{hex}	

Reading analogue inputs

Attribute ID	Action
5B _{hex}	Read analogue inputs

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Analogue input I7	Analogue input I8	00	00	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:

Value 42_{hex} = 01000010_{bin}:

Read request OK, data follow.

Analogue inputs I7 and I8 (byte 1 and byte 2)

These two bytes contain the actual value at the analogue inputs I7 and I8. Their value lies between 00 and 99, which is equivalent to a voltage level of 0 to 9.9 V at the inputs.

The corresponding values are returned in hexadecimal format.

Example:

Byte	Value	Description
0	42 _{hex}	The read request has been executed. Data follow.
1	20 _{hex}	Voltage level at input I7 = 3.2 V.
2	31 _{hex}	Voltage level at input I8 = 4.9 V.

Reading the status of digital inputs, P buttons and operator keys

Attribute ID	Action
5C _{hex}	Read the status of the digital inputs, P buttons and operator control keys

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

Reading	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Inputs I8 to I1	Inputs I16 to I9	P buttons and operator control keys	00	00	00

Answer byte (byte 0)

The first byte received from the slave contains the acknowledgement. easy confirms the execution of a "read

EASY parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit								
	7	6	5	4	3	2	1	0	
Fixed	0								
Fixed		1	0						
EASY222-DN read request rejected due to error				0	0	0	0	0	0
Read request OK, data follow				0	0	0	1	0	

Example:
 Value 42_{hex} = 01000010_{bin}:
 Read request OK, data follow.

Status at inputs I1 to I8 (byte 1)

easy600 input	Bit							
	7	6	5	4	3	2	1	0
I1								0/1
I2							0/1	
I3						0/1		
I4					0/1			
I5				0/1				
I6			0/1					
I7		0/1						
I8	0/1							

0 = status "0"
 1 = status "1"

Example:
 Value 2B_{hex} = 00101011_{bin}:
 I6, I4, I2 and I1 are active.

Status at inputs I9 to I16 (byte 2)

easy600 input	Bit							
	7	6	5	4	3	2	1	0
I9								0/1
I10							0/1	
I11						0/1		
I12					0/1			
I13				0/1				
I14			0/1					
I15		0/1						
I16	0/1							

0 = status "0"
 1 = status "1"

Example:
 Value 19_{hex} = 00011001_{bin}:
 I13, I12 and I9 are active

Status of P buttons and operator control keys (byte 3)

Meaning	Bit							
	7	6	5	4	3	2	1	0
Status P1								0/1
Status P2							0/1	
Status P3						0/1		
Status P4					0/1			
ESC not actuated/actuated				0/1				
OK not actuated/actuated			0/1					
DEL not actuated/actuated		0/1						
ALT not actuated/actuated	0/1							

0 = status "0"
 1 = status "1"

Example:
Value 01_{hex} = 00000001_{bin}:
P1 active – or cursor > is actuated.

Reading the status of the real-time clock

Attribute ID	Action
5D _{hex}	Read real-time clock

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	3	–	–	–	–	–
Slave	Answer byte	Weekday	Hour	Minute	Time-of-Year	00	00

Answer byte (byte 0)
The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:
Value 42_{hex} = 01000010_{bin}:
Read request OK, data follow.

Weekday (byte 1)

Range of values: 00 to 06.

This byte indicates the weekday as numerical value between 0 (Monday) and 6 (Sunday).

Hour (byte 2)

Range of values: 00 to 23_{dec} = 00 to 17_{hex}.

This byte returns the current hour value of the basic unit's real-time clock. This is a hexadecimal value.

Minute (Byte 3)

Range of values: 00 to 59_{dec} = 00 to 3B_{hex}.

This byte returns the current minute value of the basic unit's real-time clock. This is a hexadecimal value.

Summer/winter time (byte 4)

Range of values: 00 to 01

00 = winter time or

01 = summer time.

Example:

Value	Byte							
	0	1	2	3	4	5	6	
hex	42	04	0E	24	01	–	–	
DEC	66	04	14	36	01	–	–	

It is Friday, the current time-of-day is set to CET summer time 14:36 p.m.

Reading the status of timing relays, counter relays, switching timers and of analogue value comparators

Attribute ID	Action
5E _{hex}	Read the status of: <ul style="list-style-type: none"> • Timing relays • Counter relays, • Switching timers and • analogue value comparators.

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Image of timing relays T8 to T1	Image of counter relays C8 to C1	Image of switching timers W4 to W1	Image of analogue value comparators A8 to A1	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. "easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:

Value 42_{hex} = 01000010_{bin}:

Read request OK, data follow.

Timing relay image: Status T1 to T8 (byte 1)

Meaning	Bit							
	7	6	5	4	3	2	1	0
T1								0/1
T2							0/1	
T3						0/1		
T4					0/1			
T5				0/1				
T6			0/1					
T7		0/1						
T8	0/1							

0 = status "0"

1 = status "1"

Example:

Value 2B_{hex} = 00101011_{bin}:

T6, T4, T2 and T1 are active.

Timing relay image: Status C1 to C8 (byte 2)

Meaning	Bit							
	7	6	5	4	3	2	1	0
C1								0/1
C2							0/1	
C3						0/1		
C4					0/1			
C5				0/1				
C6			0/1					
C7		0/1						
C8	0/1							

0 = status "0"
 1 = status "1"

Example:
 Value 19_{hex} = 00011001_{bin}:
 C5, C4 and C1 are active

Switching timer image: Status W1 to W4 (byte 3)

Meaning	Bit							
	7	6	5	4	3	2	1	0
W1								0/1
W2							0/1	
W3						0/1		
W4					0/1			

0 = status "0"
 1 = status "1"

Example:
 Value 08_{hex} = 00001000_{bin}:
 W3 is active.

Analogue value comparator image: Status A1 to A8
(byte 4)

Meaning	Bit							
	7	6	5	4	3	2	1	0
A1								0/1
A2							0/1	
A3						0/1		
A4					0/1			
A5				0/1				
A6			0/1					
A7		0/1						
A8	0/1							

0 = status "0"

1 = status "1"

Example:

Value $84_{\text{hex}} = 10001000_{\text{bin}}$:

A3 and A8 are active.

Reading the status of contactor relays (markers), text display and digital outputs

Attribute ID	Action
5F _{hex}	Read the status of: <ul style="list-style-type: none"> • Contactor relay (marker) • Text displays and • Digital outputs

After the read request has been triggered by sending only the respective attribute, easy returns the corresponding data.

sending	Data						
	Byte 0:	Byte 1:	Byte 2:	Byte 3:	Byte 4:	Byte 5:	Byte 6:
Master	–	–	–	–	–	–	–
Slave	Answer byte	Image of contactor relays M8 to M1	Image of contactor relays M16 to M9	Image of outputs Q8 to Q1	Image of text markers D8 to D1	00	00

Answer byte (byte 0)

The first byte returned by the slave contains the acknowledgement. easy confirms the execution of a "read easy parameter" request. Otherwise, easy rejects this action if read access was not possible.

Meaning	Bit							
	7	6	5	4	3	2	1	0
Fixed	0							
Fixed		1	0					
EASY222-DN read request rejected due to error				0	0	0	0	0
Read request OK, data follow				0	0	0	1	0

Example:

Value 42_{hex} = 01000010_{bin}:

Read request OK, data follow.

Contactor relay image: Status M1 to M8 (byte 1)

Meaning	Bit							
	7	6	5	4	3	2	1	0
M1								0/1
M2							0/1	
M3						0/1		
M4					0/1			
M5				0/1				
M6			0/1					
M7		0/1						
M8	0/1							

0 = status "0"

1 = status "1"

Example:

Value 2B_{hex} = 00101011_{bin}:

M6, M4, M2 and M1 are active.

Contactor relay image: Status M9 to M16 (byte 2)

Meaning	Bit							
	7	6	5	4	3	2	1	0
M9								0/1
M10							0/1	
M11						0/1		
M12					0/1			
M13				0/1				
M14			0/1					
M15		0/1						
M16	0/1							

0 = status "0"
 1 = status "1"

Example:
 Value 19_{hex} = 00011001_{bin}:
 M13, M12 and M9 are active

Digital outputs image: Status Q1 to Q8 (byte 3)

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

0 = status "0"
 1 = status "1"

Example:
 Value A8_{hex} = 10101000_{bin}:
 Q8, Q6 and Q4 are active.

Text marker image: Status D1 to D8 (byte 4)

	7	6	5	4	3	2	1	0
D1								0/1
D2							0/1	
D3						0/1		
D4					0/1			
D5				0/1				
D6			0/1					
D7		0/1						
D8	0/1							

0 = status "0"

1 = status "1"

Example:

Value $84_{\text{hex}} = 10000100_{\text{bin}}$:

D3 and D8 are active.

5 What happens if...?

Module status LED MS	Possible cause	To correct or avoid error
OFF	No power at EASY222-DN.	Switch on the power supply.
Green	EASY222-DN is in normal mode of operation.	None
Green flashing	EASY222-DN not configured.	Verify the correct setting of the MAC ID.
Red flashing	Invalid configuration	Check configuration data.
RED	Module error which can not be resolved.	Replace the EASY222-DN.

Network status LED NS	Possible cause	To correct or avoid error
OFF	<ul style="list-style-type: none"> • EASY222-DN without power or • communication is blocked at this channel because <ul style="list-style-type: none"> – of bus-off state or – power loss or – the channel was blocked explicitly. 	<ul style="list-style-type: none"> • Switch on the EASY222-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 EASY222-DN may be defective.	Reset the module. If further errors occur, replace the EASY222-DN.
RED	Communication error.	Check the master PLC.

Annex

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 ZB4-101-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 ₂ S 1 cm ³ /m ³ , 4 days
Mechanical ambient conditions		
Pollution degree		2
Degree of protection (EN 50178, IEC 60529, VBG4)		IP20
Oscillations (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 610004-3	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) easy-AC (IEC/EN 61000-4-5), power cable symmetrical	kV	1
High energy pulses (Surge) easy-DC (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	Colour	Green/red
Network Status LED NS	Colour	Green/red
DeviceNet		
Device connection		5-pole socket
Electrical isolation		Bus to power supply (simple) Bus and power supply to easy basic unit (safety isolation)
Function		DeviceNetSlave
INTERFACE		
Bus protocol		DeviceNet
Baud rate, automatic detection up to	kbps	500
Bus termination resistors		Separate installation at the bus possible
Bus addresses, accessible via easy basic unit with display or EASY-SOFT		0 to 63
Services		
Module inputs		all data S1 to S8 (easy600)
Module outputs		all data R1 to R16 (easy600)
Module control commands		Read/Write Weekday, time-of-day, summer/winter time All parameters of the easy functions

Dimensions

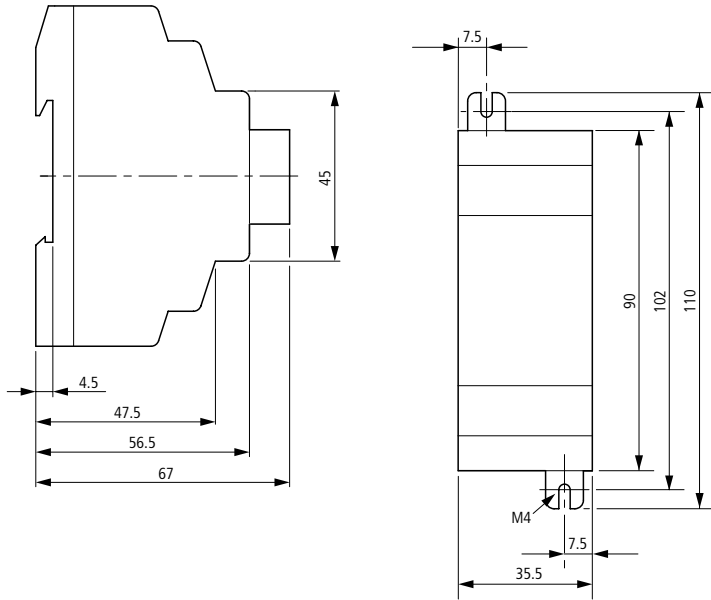


Figure 10: Dimensions EASY222-DN in mm

EDS file

```

$*****
$ Mbeller GmbH
$ Device: EASY222-DN
$ Version: V1.0
$ Date: 27.05.02
$ Author: Ronny Happ
$ Description: EDS file for easy DeviceNet slave module
$ Modifications:
$
$ Copyright (c) 2002 by Mbeller GmbH
$*****

```

[File]

```

$ File Description Section:
$ For more information about the meaning of each entry, please check
$ DeviceNet Specification Volume II Chapter 4-3.5.1

```

```

DescText      = "Mbeller DeviceNet Coupler easy 222-DN";
CreateDate    = 27-05-2002;
CreateTime    = 17:00:00;
ModDate      = 25-06-2002;
ModTime      = 11:00:00;
Revision     = 1.0;

```

[Device]

```

$ Device Description Section:
$ For more information about the meaning of each entry, please check
$ DeviceNet Specification Volume II Chapter 4-3.5.2

```

```

VendCode      = 248;           $ Identity Object - Vendor ID
ProdType      = 12;           $ Identity Object - Device Type
ProdCode      = 650;         $ Identity Object - Product Code
MajRev        = 1;           $ Identity Object - Major Revision
MinRev        = 1;           $ Identity Object - Minor Revision

```

```

                                $ Identity Object - Product Name
ProdName      = "EASY 222-DN";
VendName      = "Mbeller ElectroniX";
ProdTypeStr   = "Generic";
Catalog       = "Mbeller HPL order no. 233540";

[IO_Info]
$ I/O Characteristics Section:
$ For more information about the meaning of each entry, please check
$ DeviceNet Specification Volume II Chapter 4-3.5.3

Default       = 0x000D;           $ Cyclic, Change of State and Poll

PollInfo      =
0x000D,       $ Poll (OK to combine with Cyclic or COS)
2,            $ Default input = Input 2
1             $ Default output = Output 1

COSInfo       =
0x000D,       $ COS (OK to combine with Poll)
1             $ Default input = Input 1
2;           $ Default output = Output 2

CyclicInfo    =
0x000D,       $ Cyclic (OK to combine with Poll)
1,            $ Default input = Input 1
2;           $ Default output = Output 2

$ Input Connections
Input1        =
2,            $ 2 bytes are transferred
16,          $ all bits are significant
0x0004,       $ COS only
"Diagnostic Data from easy", $ Name
6, "20 04 24 64 30 03",    $ Assembly Object Instance 100,
                                $ Attribute 3
";           $ Help

```

```

Input2      =
             3,                $ 3 bytes are transferred
             24,               $ all bits are significant
             0x0001,           $ Poll only
             "Input Data from easy", $ Name
             6, "20 04 24 65 30 03", $ Assembly Object Instance 101,
                                     $ Attribute 3
             "";               $ Help
$ Output Connections
Output1     =
             3,                $ 3 bytes are transferred
             24,               $ all bits are significant
             0x0001,           $ Poll and COS
             "Output Data to easy", $ Name
             6, "20 04 24 66 30 03", $ Assembly Object Instance 102,
                                     $ Attribute 3
             "";               $ Help
Output2     =
             0,                $ 0 byte is transferred
             0,                $ all bits are significant
             0x0004,           $ Poll and COS
             "Acknowledge Handler", $ Name
             6, "20 2B 24 01 30 00", $ Acknowledge Handler
             "Acknowledge Handler"; $ Help

[ParamClass]
$ Parameter Class Section:
$ For more information about the meaning of each entry, please check
$ DeviceNet Specification Volume II Chapter 4-3.5.4 and Chapter 6-14.1

MaxInst     = 0;                $ no parameters are supported
Descriptor   = 0;                $
CfgAssembly = 0;                $ not used here

```

[Params]

\$ Parameter Section:

- \$ For more information about the meaning of each entry, please check
- \$ DeviceNet Specification Volume II Chapter 4-3.5.5 and Chapter 6-14.2

[EnumPar]

\$ Parameter Enumerated String Section:

- \$ For more information about the meaning of each entry, please check
- \$ DeviceNet Specification Volume II Chapter 4-3.5.6

[Groups]

\$ Parameter Groups Section:

- \$ Not used here
- \$ For more information about the meaning of each entry, please check
- \$ DeviceNet Specification Volume II Chapter 4-3.5.7

\$ End of File

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 assembly parts carrying live voltage during 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.
Analogue	Infinite proportional value, e.g. of a voltage. Analogue 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 for the data transmission speed in bps.
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-piece element for unique assignment of the electronic and basic modules.
Command modules	Represents modules with an internal memory set, capable of executing specific instructions (e.g. the output of substitute values).
Common potential	Electrical connection 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.
CPU	Abbreviation for "Central Processing Unit". Central unit for data processing. Represents the core element of a computer.
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.".
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 (easy object) for functions of the easy 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 "Electro Static 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 a 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".
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".

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". 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
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.
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. A short-circuit proof component withstands thermal and dynamic stress, which may develop at its installation location as a result of a short-circuit.
Protective conductor	Conductor required for human body protection against hazardous currents. Abbreviation: PE ("Protective Earth").
Radiation coupling	Radiation coupling develops when an electromagnetic wave meets a conductor structure, thus inducing 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 visualising 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 caused by signal reflexion and is used for the adaptation of bus cables. 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.
unidirectional	Operating in one direction.

Index

A	Address range	17
	Analogue value comparator, setting the function	54
	Analogue value comparator, setting the reference value	56
	Application Objects	29
	Application-specific objects	29
	Assembly Objects	29
	Auto baud recognition	16
<hr/>		
B	Bus cable lengths	16
<hr/>		
C	Communication profile	9
	Connection objects	29
	Control commands	40
	COS I/O connection	25
	Counter relays,	
	Reading process variables	64
	Reading reference values	66
	Setting reference values	48
	Cycle time	22
	Cyclic I/O connection	25
<hr/>		
D	Data exchange method	40
	DeviceNet	
	Connecting	13
	Object	29
	Terminal assignment	13
	DeviceNet terminal assignment	13
	Dimensions	94
<hr/>		
E	easy Object	29
	easy object	33
	EDS file	26
	Explicit Messages	24

H	Hardware requirements	9
	Heartbeat Message	26
<hr/>		
I	Identity Object	28
	Initial power on	17
<hr/>		
L	LED MS	89
	LED status displays	20, 89
<hr/>		
M	Message Router Object	28
	Module status LED	20, 89
	MS LED	20
<hr/>		
N	Network Status LED	21
	Network status LED	89
	NS LED	21, 89
<hr/>		
O	Offline Connection Set	25
	Operating system requirements	9
<hr/>		
P	Polled I/O connection	24, 35
	Potential isolation	15
	Power supply	12
<hr/>		
R	Reading outputs S1 to S8	36
	Reading the hour	72, 78
	Reading the minute	72, 78
	Reading the process variable at analogue inputs ..	73
	Reading the summer/winter time	78
	Reading the time-of-day	77
	Reading the weekday	78
	Response time of the basic unit	22

S	Set operating mode	37
	Setting inputs R1 to R8	39
	Setting R9 to R16 inputs	39
	Setting the hour	57
	Setting the minute	57
	Setting the slave address	17
	Setting the summer/winter time	58
	Setting the time-of-day	57
	Setting the weekday	57
	Shut Down Message	26
	State	
	Reading outputs S1 to S8	36
	Reading the mode	36
	Reading the status of analogue value comparators	79
	Reading the status of contactor relays (markers)	84
	Reading the status of digital outputs	84
	Reading the status of operator keys	74
	Reading the status of P buttons	74
	Reading the status of switching timers	79
	Reading the status of timing relays	79
	Reading the text display	84
	Status	
	Reading the input status	74
	Reading the status of counter relays	79
	Set operating mode	37
	Setting inputs R1 to R16	39
	Structure of the unit	8
	Switching timer	
	Reading the channel	68
	Reading the ON/OFF times	72
	Setting the channel	51
	Setting the OFF time	53
	Setting the ON time	53
	Switching timers	
	Setting the ON/OFF times	51
	System overview	7

T	Terminating resistors	13
	Timing relays	
	Reading actual values	59
	Reading reference values	62
	Reading the random value	61
	Reading the switching function	60
	Setting parameters	44
	Setting the reference value	45
	Setting the switching function	44
	Transmission rates	16
<hr/>		
U	UCMM	25

252m0010.eps@7 08/02 AWB2528-1427GB
252i0620.eps@8 08/02 AWB2528-1427GB
252i0500.eps@11 08/02 AWB2528-1427GB
252n0280.eps@12 08/02 AWB2528-1427GB
252s0450.eps@12 08/02 AWB2528-1427GB
252n0070.eps@13 08/02 AWB2528-1427GB
252s1200.eps@13 08/02 AWB2528-1427GB
252s1210.eps@14 08/02 AWB2528-1427GB
270i1870.eps@14 08/02 AWB2528-1427GB
270i1890.eps@14 08/02 AWB2528-1427GB
270i1880.eps@15 08/02 AWB2528-1427GB
270i1900.eps@15 08/02 AWB2528-1427GB
252n0400.eps@15 08/02 AWB2528-1427GB
252i0210.eps@18 08/02 AWB2528-1427GB
252i0230.eps@18 08/02 AWB2528-1427GB
252i0190.eps@18 08/02 AWB2528-1427GB
252i0180.eps@18 08/02 AWB2528-1427GB
252i0190.eps@19 08/02 AWB2528-1427GB
252i0200.eps@19 08/02 AWB2528-1427GB
252u0460.eps@20 08/02 AWB2528-1427GB
252u0450.eps@21 08/02 AWB2528-1427GB
252n010g.eps@27 08/02 AWB2528-1427GB
252v0110.eps@94 08/02 AWB2528-1427GB
252v0060.eps@94 08/02 AWB2528-1427GB