Operating manual

EASY204-DP
PROFIBUS-DP Slave Interface

07/04 AWB2528-1401GB

MOELLER

Think future. Switch to green.

1st published 2001, edition date 04/01
2nd edition 05/2003
3rd edition 07/2004
See revision protocol in the "About this manual" chapter

Authors: Ronny Happ, Dieter Bauerfeind
Editor: Thomas Kracht
Translator: Terence Osborn

Contents

About This Manual

List of revisions

The following essential changes have been made since the previous version:

Edition date	Page	Description	New	Modification
08/02	33	Section "PROFIBUS certification"	✓	
05/03	gen.	easy800/MFD	✓	
07/04	gen.	easy700	✓	

Target readership

This manual has been produced for automation technicians and engineers. A thorough knowledge of the PROFIBUS-DP fieldbus and the programming of a PROFIBUS-DP master is required. You should also be familiar with the operation of the easy control relay or MFD multi-function display.

Other manuals on the device

The following operating manuals should be followed:

- easy412 control relay, easy600 (AWB2528-1304-GB)
- easy700 control relay, (AWB2528-1508-GB)
- easy800 control relay, (AWB2528-1423-GB)
- MFD-Titan multi-function display (AWB2528-1480-GB)

All manuals are available on the Internet for download as PDF files. For a fast search enter the documentation number as the search criterion at http://www.moeller.net/support:

Device designation

This manual uses the following short names for equipment types, as far as the description applies to all of these types:

- easy412 for EASY412-..-... devices
- EASY512-..-..., EASY7..-..-...

Type designation of the control relay, the point represents a placeholder for all characters used.

- easy500 for
  - EASY512-AB...
  - EASY512-AC
  - EASY521-DA…
  - EASY512-DC
- easy600 for
  - EASY6..-AC-RC(X)
  - EASY6..-DC.-.C(X)
- easy700 for
  - EASY719-AB…
  - EASY719-AC…
  - EASY719-DA…
  - EASY719-DC…
  - EASY721-DC…
- easy800 for
  - EASY819-...
  - EASY820-...
  - EASY821-...
  - EASY822-...
- MFD-CP8.. for
  - MFD-CP8-ME
  - MFD-CP8-NT

- easy-AB for
  - EASY512-AB...
  - EASY719-AB...
- easy-AC for
  - EASY412-AC-..
  - EASY512-AC-..
  - EASY6..-AC-RC(X),
  - EASY719AC
  - EASY8..-AC-...
- easy-DC for
  - EASY412-DC-..
  - EASY512-DC-..
  - EASY6..-DC-…
  - EASY719-DC-..
  - EASY8..-.DC-...
- easy-DA for
  - EASY412-DA...
  - EASY512-DA...
  - EASY719-DA...

Abbreviations

This manual uses abbreviations with the following meaning:

hex:	Hexadecimal (number system with base 16)
dec :	Decimal (number system with base 10)
bcd:	binary coded decimal code
VR:	**V**alue **r**ange
PC:	**P**ersonal **C**omputer

Writing conventions

Except for the first page of chapters and empty pages at the end, the top left of the page shows the chapter title and the top right of the page shows the current section for greater clarity.

▶ indicates actions to be taken.

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

Caution!
Warns of the possibility of serious damage and slight injury.

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

Indicates interesting tips and additional information

1 EASY204-DP

The EASY204-DP communication module was developed for automation tasks using the PROFIBUS-DP fieldbus. The EASY204-DP is a gateway and can only be used in conjunction with the easy600, easy700, easy800 or MFD basic units. The easy control relay or MFD device with a PROFIBUS-DP gateway always works as a slave station on the network.

System overview

The "easy" PROFIBUS-DPV0 slaves are integrated in a PROFIBUS-DP system.

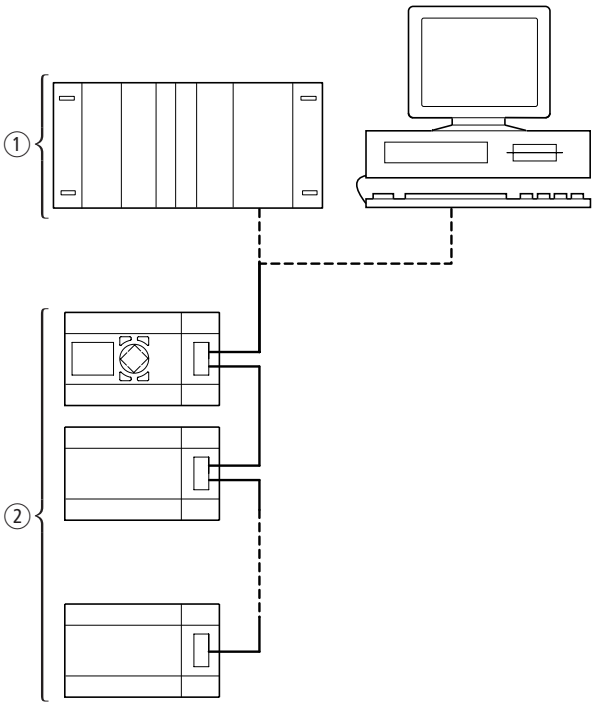


Figure 1: Integration of EASY204-DP in the DP network

① Master area, PLC or PC

② Slave area, e.g. easy /MFD with DP interface

Device setup

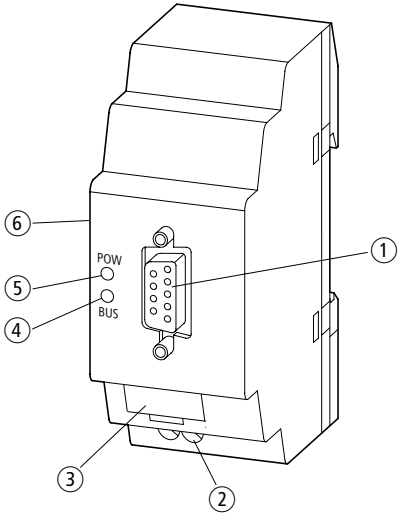


Figure 2: View of the device

① PROFIBUS-DP connection, 9-pole SUB-D socket
② 24 V DC supply voltage
③ Device designation plate
④ BUS communication LED
⑤ POW operation LED
⑥ EASY-LINK socket

Device function description

The EASY204-DP module allows the easy and MFD series devices to be connected to a PROFIBUS-DP communication network. The following data can be transferred by selecting the appropriate SDO/PDO:

easy600/700/800, MFD-CP8..

- S1 to S8
  Output data of the basic unit, RUN/STOP
  (read, as viewed from PROFIBUS-DP master)
- R1 to R16
  Input data of the basic unit, RUN/STOP
  (write, as viewed from PROFIBUS-DP master)
- All function relay data
  (read, as viewed from the PROFIBUS-DP master)
  - Timing relays
  - Counter relays
  - Time switches
  - Analog comparators
  - Weekday, time, summer/winter time (DST)
  - All states of the easy600 contacts.

- The setpoints of the function relays
  (write, as viewed from PROFIBUS-DP master)
  - Timing relays
  - Counter relays
  - Time switches
  - Analog comparators
  - Weekday, time, summer/winter time (DST)

easy800/MFD-CP8..

- All markers and easyNet data
- Function blocks
  (read/write, as viewed from the master)
  - Arithmetic function blocks
  - Frequency counters, high-speed counters, incremental encoder counters
  - 7-day and year time switch
  - Operating hours counter
  - PID controllers
  - PWM (pulse width modulation)
  - Real-time clock

Hardware and operating system requirements

The EASY204-DP expansion device operates with the easy600, easy700, easy800 and MFD basic units from the following operating system versions:

Basic unit		**EASY204-DP expansion device**	
Device version	OS version	Device version ≦ 04	Device version ≦ 05
easy600			
≧ 04	From 2.4	×	×
easy700			
≧ 01	From 1.10.xxx	–	×
easy800			
≧ 04	From 1.10.xxx	–	×
MFD-CP8..			
≧ 01	From 1.10.xxx	–	×

The device version of the appropriate basic unit or expansion device is specified on the right of the housing.
Example: EASY204-DP: 03-228xxxxxxx (03 = device version)
The operating system version (OS) of the corresponding basic unit can be read using EASY-SOFT. With easy700, easy800 and MFD-CP8.. devices it is also possible to read out the information directly from the device. Read the appropriate manual for further information for this.

List of revisions

Modifications from version 04 to version 05:

- Extension of the EASY-LINK protocol for connection to easy800.

Modifications from version 05 to version 06:

- Extension of the EASY-LINK protocol for connection to easy700.
- Cyclical data transfer (3-byte module) adapted in Byte 0 (see also chapter "What Happens If ...?" on page 229.

Improper use

easy must not be used as a replacement for safety PLCs such as

- Burner,
- Emergency-stop,
- Crane or
- Two-hand safety controls.

2 Installation

The same principles apply as for easy600, easy700, easy800 and MFD basic units with expansion devices.

Connecting EASY204-DP to the basic unit

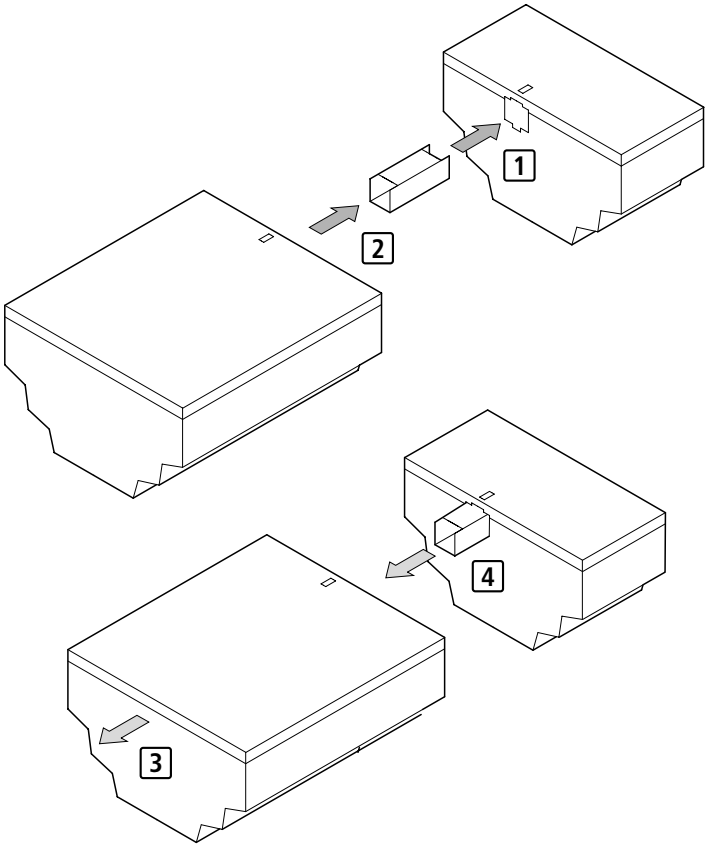


Figure 3: Fitting [1] + [2] or removing [3] + [4] the EASY204-DP to the basic unit

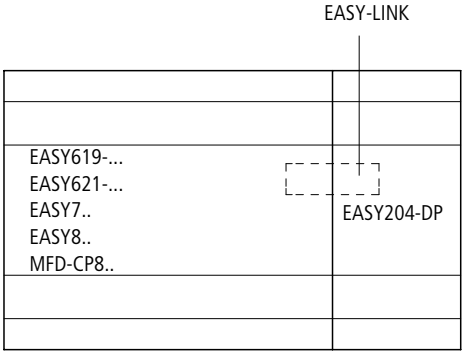


Figure 4: Connection between basic unit and EASY204-DP

Connecting the power supply

The EASY204-DP unit is run on a 24 V DC power supply (→ section "Technical Data" from page 235).

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

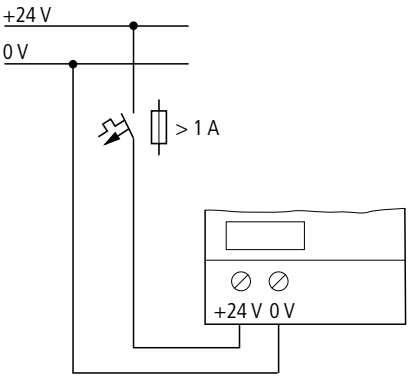
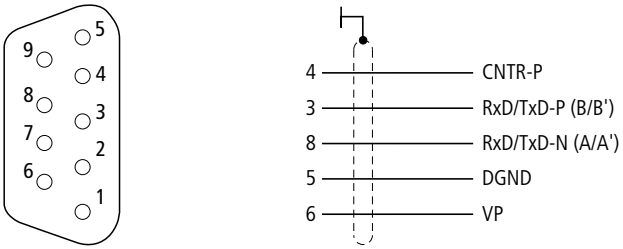


Figure 5: Standard connection

Connecting PROFIBUS-DP

Use a 9-pole SUB-D plug to connect the PROFIBUS-DP interface to the PROFIBUS-DP fieldbus. For this use the special PROFIBUS-DP plug and the special PROFIBUS-DP cable available from the Moeller range of accessories. The type of cable used determines the permissible maximum bus length and the transfer rate.

PROFIBUS-DP connection assignment



Pin	Signal name	Designation
1	Not assigned	-
2	Not assigned	-
3	RxD/TxD-P (B Line)	Receive/Send data P
4	CNTR-P / RTS	Request to Send
5	DGND	Data reference potential
6	VP	+5V DC for external bus connection
7	Not assigned	-
8	RxD/TxD-N (A-Line)	Receive/Send data N
9	Not assigned	-

Connections 3, 8 and the shield are sufficient for data transfer.

Bus terminating resistors

The first and last station in a bus segment must be connected to the bus with the bus terminating resistor switched on. The bus terminating resistor is switched externally. This external switch function can either be implemented as a separate bus terminating resistor or with a special Sub-D plug with an integrated bus termination.

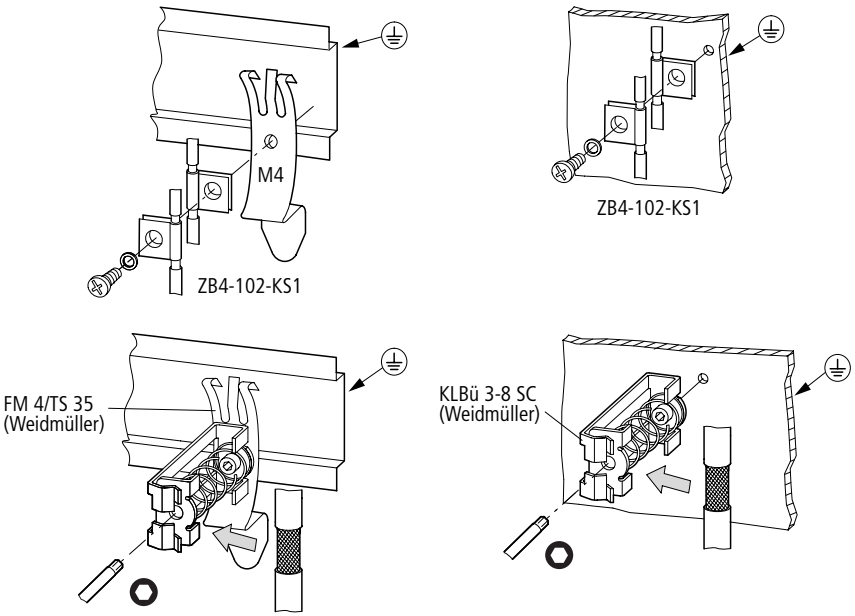
Moeller's PROFIBUS-DP data plug enables both bus terminating resistors to be switched on and off.

EMC wiring

Electromagnetic interference may have adverse effects on the communication fieldbus. This can be minimized in advance be taking suitable EMC measures. These include:

- System design in accordance with EMC requirements,
- EMC cabling and
- Measures that prevent the occurrence of large potential differences,
- Correct installation of the PROFIBUS system (cable, connection of bus connector,...).

The effects of electromagnetic interference can be significantly reduced by fitting the shield. The following two figures illustrate how to fit the shield.



Electrical isolation

The following electrical isolation should be provided for the interfaces of the EASY204-DP:

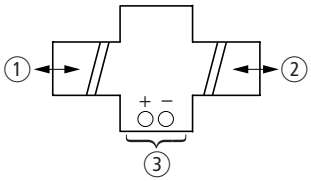


Figure 6: Potential isolation between the power supply and outputs

① Safe isolation of EASY-LINK 240 V AC
② Simple isolation of PROFIBUS-DP
③ 24 V DC supply voltage

Transfer rates – automatic baud rate detection

The EASY204-DP module automatically detects the baud rate used in the communication network after it is switched on. However, this requires that at least one station sends valid telegrams in the network.

The following transfer rates are supported:

The EASY204-DP module detects the transfer rate automatically. The following transfer rates are supported:

- 9.6 Kbit/s to 12000 Kbit/s

Maximum distances/bus cable lengths

Two types of bus cable are specified in IEC 61158. Cable type B should no longer be used with new applications because it is obsolete. Cable type A allows all transfer rates up to 12000 Kbit/s to be used. Cables for burial in the ground, festoon suspension and drum cables are also available.

The cable parameters are as follows:

Parameters	Cable type A
Surge impedance in Ω	135 ... 165 at 3 ... 20 MHz
Effective capacitance (pF/m)	< 30
Loop resistance (Ω/km)	< 110
Core diameter (mm)	> 0.64
Core cross-section (mm^2)	> 0.34

The cable parameters specified allow the following bus segment lengths.

Distance between stations when using Type A cable to IEC 61158:

Baud rate [Kbit/s]	Max. cable length Type A cable [m]
9.6	1200
19.2	1200
93.75	1200
187.5	1000
500	400
1500	200
3000	100
6000	100
12000	100

Distance between two stations when using Type B cable to IEC 61158:

Baud rate [Kbit/s]	Max. cable length Type B cable [m]
9.6	1200
19.2	1200
93.75	1200
187.5	1000
500	400
1500	–

3 Device Operation

Initial power up

- ▶ Before you switch on the device, verify that it is properly connected to the power supply, to the bus connector and to the basic unit.
- ▶ Switch on the power supply to the basic unit and the PROFIBUS-DP expansion unit.

The Power LED of the EASY204-DP is lit. The BUS LED is off (no communication via PROFIBUS-DP).

The GW message (intelligent station connected) is displayed on the basic unit.

Basic unit	Device version	GW display
easy600	04	Static
easy700	From 01	Flashing
easy800	04	Static
	From 05	Flashing
MFD-CP8..	01	Static
	From 02	Flashing

As soon as the device is integrated in the PROFIBUS-DP network, the BUS LED is continuously lit ("static") and the GW message is statically displayed, also on devices with a flashing GW message.

→ Valid data is only transferred via PROFIBUS-DP to the basic unit if the GW is displayed statically.

If the PROFIBUS-DP unit is factory set, the station address of the PROFIBUS-DP station must be set.

Setting the PROFIBUS-DP station address

Every PROFIBUS-DP station requires an unambiguous address in the PROFIBUS-DP structure. There are two ways of setting the PROFIBUS-DP addresses on the EASY204-DP:

- Using the integrated display and keypad on the easy or MFD-Titan basic unit
- Using EASY-SOFT Version 3.01 or higher on the PC.

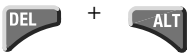
Address range: 001 to 126

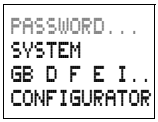
Setting the address on the basic unit with a display

Requirements:

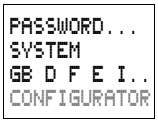
- The basic unit (easy600, easy700, easy800 or MFD-Titan) and the EASY204-DP expansion unit must be fed with power.
- The basic unit has been unlocked (no password activated).
- The basic unit has a valid operating system version (→ page 15).
- The basic unit must be in STOP mode.
- The EASY204-DP is not communicating with the PROFIBUS-DP master (Bus LED is off).

▶ Enter the System menu by pressing DEL + ALT simultaneously.





▶ Use cursor buttons ∧ or ∨ to select CONFIGURATOR

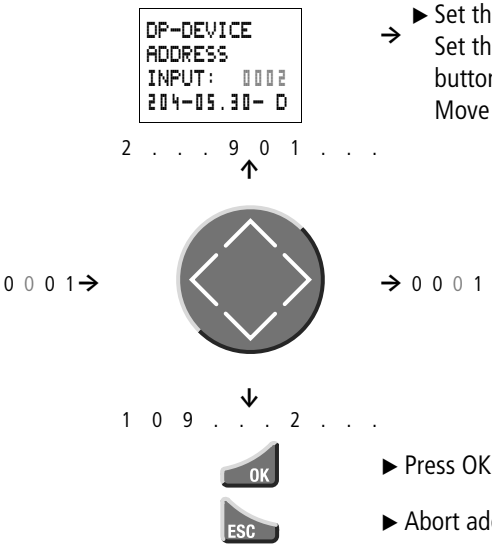


▶ Confirm your entry with OK

▶ With easy800/MFD devices select the LINK...

▶ Confirm with OK.

easy600 and easy 700 devices show the following dialog immediately:

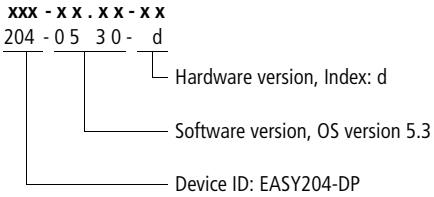


▶ Set the address:
Set the value of the current digit with ∧ or ∨ buttons.
Move to the next digit with < or >.

▶ Press OK to accept the address or

▶ Abort address entry.

Information on the 4th display line:



Setting the address using EASY-SOFT

With EASY-SOFT, version 3.1
‹Menu → Online → Configure Expansion Devices›

From EASY-SOFT version 4.01 or higher:
Choose → Communication → Configuration → Expansion Devices → EASY204-DP.

→ The menu is only available in Communication View, therefore activate the Communication tab.

Status LEDs

The EASY204-DP expansion unit has two LEDs.

POW LED, Function

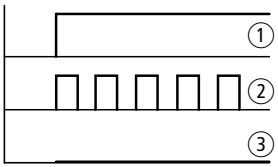


Figure 7: Function of the POW LED

① LED continuously lit:
- – Power supply present
- – Communication with the basic unit aborted

② LED flashing:
- – Power supply present
- – Communication with the basic unit correct

③ LED not lit:
- – No power supply present
- – Communication with the basic unit aborted

BUS LED, Function

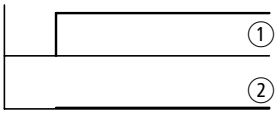


Figure 8: Function of the BUS LED

① LED continuously lit:
- – PROFIBUS-DP communication correct

② LED not lit:
- – No PROFIBUS-DP communication present

Cycle time of EASY basic unit

Communication between the basic unit and EASY204-DP via EASY-LINK increases the cycle time of the basic unit.

In extreme cases the cycle time may increase by 40 ms.

This should be taken into account for the reaction times of the basic unit.

4 PROFIBUS-DP Functions

Slave modules

The EASY204-DP expansion module is a PROFIBUS-DP slave in compliance with IEC 61186/EN 50170.

You can select the following EASY204-DP slave modules via the PROFIBUS-DP Configurator in the master PLC by using the appropriate GSD file. These modules are described in detail in chapters 5, 6 and 7.

Module designation	Inputs (Bytes	Outputs (Bytes	Inputs/outputs (Bytes	Service	Supported devices	Code in GSD file	Page
Control level							
1: Control commands, 7 bytes	–	–	7	• Time • Image • Function blocks	easy600	0xB6	45
2: Control commands, 9 bytes	–	–	9	• Real-time clock • Image • Function blocks	easy700, easy800, MFD-CP8..	0xB8	89
Input/output level							
3: Inputs, 3 bytes	3	–	–	• Read data: S1 – S8 • Operating mode	easy600, easy700, easy800, MFD-CP8..	0x92	36
4: Outputs, 3 bytes	–	3	–	• Write data: R1 – R8, R9 – R16 • Operating mode		0xA2	38
5: Inputs, 1 byte	1	–	–	• Read data: S1 – S8		0x90	38
6: Outputs, 1 byte	–	1	–	• Write data: R1 – R8, R9 – R16		0xA0	42

Observe the requirements of the operating system
→ page 15!

Diagnostics data

The EASY204-DP PROFIBUS-DP device features the standard diagnostics in accordance with the PROFIBUS specification.

Two additional diagnostics bytes are also sent.

Byte 0	Length of additional diagnostics bytes
Fixed 02_{hex}	00000010
Byte 1	Status of EASY-LINK
Value 00_{hex}	EASY-LINK is connected
Value 01_{hex}	EASY-LINK is disconnected

GSD file

A PROFIBUS-DP GSD file is required for selecting the device and for running it on the PROFIBUS-DP communication bus. The GSD file contains standard PROFIBUS station descriptions and is contained in the Appendix of this manual. The file "Moel4d10.gsd" can be obtained at the following Internet address:

- http://www.moeller.net → Support →...
- http://easy.moeller.net → Download →...
- ftp://ftp.moeller.net/EASY/GSD-FILES/

Follow the links on these pages.

Application modules

All easy600 functions that are available via the EASY204-DP are supported by Moeller PLCs PS4-341, PS416, XControl and the Siemens S7 PLC. The following PLC application modules offer a convenient option for implementing the data exchange between the EASY control relay and the master PLC.

The following application modules are available:

PLC	Application module Application note	File	
		German	English
Moeller PLCs			
PS4-341 and PS416			
easy600	S40-AM-K6-D/GB	s40amk6d.exe	s40amk6g.exe
easy800/MFD	AN2700K21D/GB	an27k21d.exe	an27k21g.exe
XC PLCs			
easy600	S40-AM-K6-D/GB	xs-easydp_d.exe	xs-easydp_g.exe
Siemens PLCs			
SIMATIC S7-300			
easy600	S7-AM-K6-D/GB	s7amk6d.exe	s7amk6g.exe

Both the application modules and application notes listed in the table and other modules for the user-friendly configuration of easy800 and MFD-Titan functions can be downloaded from the following Internet address:

- http://www.moeller.net/automation/...
- ftp://ftp.moeller.net/AUTOMATION/APPLICATION_MODULES/

PROFIBUS certification

EASY204-DP was certified as a PROFIBUS-DP device by the PROFIBUS User Organisation. EASY204-DP contains the PROFIBUS VPC3+ interface.

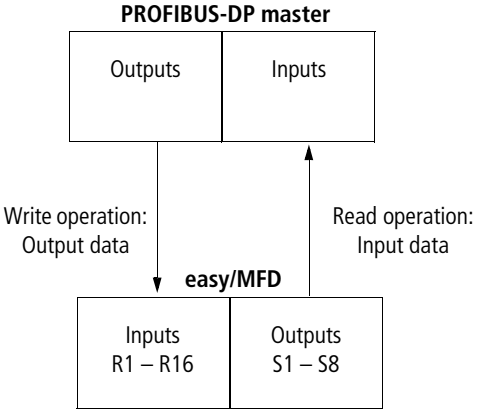
→ Irregular operation may occur under the following conditions:

- When in a multimaster system Class I and Class II DP masters with parameter or configuration data access the slave at the same time (highly unlikely).
- Or if other masters based on PROFIBUS layer 2 are connected.

5 Inputs/Outputs, easy600/700/800/ MFD Operating Mode

The appropriate module must be selected in the slave configuration in order for I/O data to be transferred between the EASY204-DP slave and a PROFIBUS-DP master.

→ The terms "input data" and "output data" are used from the point of view of the PROFIBUS-DP master.



"Inputs 3 bytes" module: operating mode, S1 – S8

The normal PROFIBUS-DP master data exchange with the EASY204-DP slave is via input data bytes 0, 1, 2.

Byte	Meaning	Value
0	Scan the operating mode	→ table 1
1	Scan status of the easy outputs S1 to S8	→ table 2
2	Not assigned	00_{hex}

Requirement:
The "Inputs, 3 bytes" module must have been selected.

→ The output data and control commands can now only be used if you have selected the appropriate modules as well.

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

Table 1: Byte 0: Operating mode

"easy" operating mode	Bit							
	7	6	5	4	3	2	1	0 Stop/Run
with debounce	0	0	0	1	0	0	0	0/1
without debounce	0	0	1	0	0	0	0	0/1

Example:

Value 21_{hex} "easy"/MFD is in Run mode and is working with input debounce.

Table 2: Byte 1: Status of S1 to S8 on the basic unit

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							

Example:

Value 19_{hex} S5, S4 and S1 are active.

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

- Whilst the control command is being executed, the inputs will remain in the state before the control command was called.
- After the "Control commands" data exchange has been completed, the input bytes are refreshed.

"Inputs 1 byte" module: S1 – S8

When this module is selected, the master only receives 1 byte (coil output data S1 to S8) via PROFIBUS.

Byte	Meaning	Value
0	Scan status of the easy outputs S1 to S8	→ table 2 on page 37

Requirement:
The "Inputs, 1 byte" module must have been selected.

→ The output data and control commands can now only be used if you have selected the appropriate modules as well.

"Outputs 3 bytes" module: operating mode, R9 - R16, R1 – R8

The normal PROFIBUS-DP master data exchange with the EASY204-DP slave is provided with output data bytes 0, 1, 2.

Byte	Meaning	Value
0	Set operating mode	→ table 3
1	Set/reset the easy/MFD inputs R9 to R16	→ table 4
2	Set/reset the easy/MFD inputs R1 to R8	→ table 5

Requirement:
The "Outputs; 3 bytes" module must have been selected.

→ The output data and control commands can now only be used if you have selected the appropriate modules as well.

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

Table 3: Byte 0: Operating mode

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

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

Explanation
Value $34_{hex} = 00110100_{bin}$:
This value sets the easy/MFD status from STOP to RUN. It is only interpreted as a command and therefore does not permit an additional transfer of data. The index value 14_{hex} must be used in this situation.

Value $44_{hex} = 01000100_{bin}$:
This value sets the easy/MFD status from RUN to STOP. It is also used only as command and is therefore works in the same way as the RUN command.

This should be observed without fail up to device version 05!:
Value $14_{hex} = 00010100_{bin}$:
Byte 0 must always contain this value if data is to be written to the easy/MFD basic unit via the gateway.

→ Even if the I/O of a control relay can be assigned directly to a specific memory area of the master PLC, the correct data structure format (e.g.: input data byte 0 = 14 hex must nevertheless still be observed.

From version 06 this is no longer necessary.

Table 4: Byte 1: Write status of R9 to R16

EASY6.. Input	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							

Example:

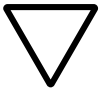
Value 19_{hex} R13, R12 and R9 should be active.

Table 5: Byte 2: Write status of R1 to R8

EASY6.. Input	Bit 7	6	5	4	3	2	1	0
R1								0/1
R2							0/1	
R3						0/1		
R4					0/1			
R5				0/1				
R6			0/1					
R7		0/1						
R8	0/1							

Example:

Value $2B_{hex}$ R6, R4, R2 and R1 should be active.



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

- Whilst the control command is being executed, the inputs will remain in the state before the control command was called.
- After the "Control commands" data exchange has been completed, the output bytes are refreshed.

"Outputs 1 byte" module: R1 – R8

When this module is selected, the master only sends 1 byte (coil output data S1 to S8) via PROFIBUS.

Byte	Meaning	Value
0	Status of R1 to R8	→ table 5 on page 41

Requirement:
The "Outputs; 1 byte" module must have been selected.

→ The input data and control commands can now only be used if you have selected the appropriate modules as well.

Note on using the 1 byte modules
The 1 byte modules are not available in all device combinations. If any problems occur in handling, first check the state of the GW message in the status display of the basic unit:

GW static:	The 1-byte mode can be used
GW flashing:	Check the device version of the EASY204-DP and the basic unit. If these are valid, check the configuration in the PROFIBUS network and the Configurator.

Table 6: Possible device combinations for using the 1-byte mode

Basic unit	**Easy204-DP**
Easy600:	Version 05 or 06
Easy700:	Version 06
easy800	
Version ≦ 04:	Version 05 or 06
Version ≧ 05:	Version 06
MFD	
Version ≦ 02:	Version 05 or 06
Version ≧ 03:	Version 06

If the 1-byte mode cannot be used, use the 3-byte mode and ensure that the value = 0x14 is entered in Byte 0. Without this additional entry no valid values will be detected by the basic unit. As soon as the device is ready for operation and data communication is running, this is indicated by the static display of the GW message on the LCD of the basic unit.

6 Control Commands for easy600

Data exchange procedure

The "Control commands 7 bytes" module allows extended data exchange of the easy600 on the PROFIBUS-DP communication bus. This allows you to transfer services from the following areas:

- "Date and time, Summer/winter time",
- "Read/write function blocks" and
- "Read/write image".

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

A special command code in byte 0 is used to activate the services required. Data bytes 1 to 6 are used to write or read the values concerned.

Attention!

Whilst the control command is being executed, the input and output data will remain in the state before the control command was called. Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

Caution!
Only those values specified for the command code should be used.

Check the values that you write in order to avoid malfunctions.

Requirement:
The "Control commands 7 byte" module must have been selected.

Data can only be written if the "easy" basic unit with the LCD display is showing the Status display.

The master initiates the data exchange of the control commands and the addressed slave responds.

During the communication 7 bytes of data are transferred on PROFIBUS.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6

Byte 0, Toggle byte
Byte 0 contains the Toggle bit and the command. It is used to activate the sending of a control command.

Byte 0, Toggle byte								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	0/1	x	x	x	x	x	x	x
	Toggle bit	Command						

Procedure

- ▶ To send a command, bit 7 seven must be toggled, i.e. set either from 1 to 0 or from 0 to 1.
- ▶ Then poll the toggle bit for the coupling modules response until it has the same status as the toggle bit sent. This status indicates to the master that the response to the sent command is valid.
- ▶ Do not send a new command until you have received a response (changing of the toggle bit), otherwise the response of the previous command will be overwritten before it can be read.

→ In order to use input/output data and control commands simultaneously:

Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

Date and time, Summer/ winter time **Telegram structure**

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	Read	3C	–	0/1	0	1	1	1	1	0	0
	Write	2A	–	1/0	0	1	0	1	0	1	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Write successful	–	C1/41	1/0	1	0	0	0	0	0	1
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Weekday										
	Read operation	00	→ table 7								
	For write operation	→ table 7	00								
2	Hour										
	Read operation	00	→ table 8								
	For write operation	→ table 8	00								
3	Minute										
	Read operation	00	→ table 9								
	For write operation	→ table 9	00								
4	Summer/winter time										
	Read operation	00	→ table 10								
	For write operation	→ table 10	00								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 7: Byte 1: Weekday (value range 00 to 06)

Weekday	Bit							
	7	6	5	4	3	2	1	0
Monday = 0	0	0	0	0	0	0	0	0
Tuesday = 1	0	0	0	0	0	0	0	1
Wednesday = 2	0	0	0	0	0	0	1	0
Thursday = 3	0	0	0	0	0	0	1	1
Friday = 4	0	0	0	0	0	1	0	0
Saturday = 5	0	0	0	0	0	1	0	1
Sunday = 6	0	0	0	0	0	1	1	0

Table 8: Byte 2: Hour (value range 00 to 23)

Value (bcd)	Value 10 Bit				Value 1 Bit			
	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1
....								
9	0	0	0	0	1	0	0	1
....								
14	0	0	0	1	0	1	0	0
...								
23	0	0	1	0	0	0	1	1

Table 9: Byte 3: Minute (value range 00 to 59)

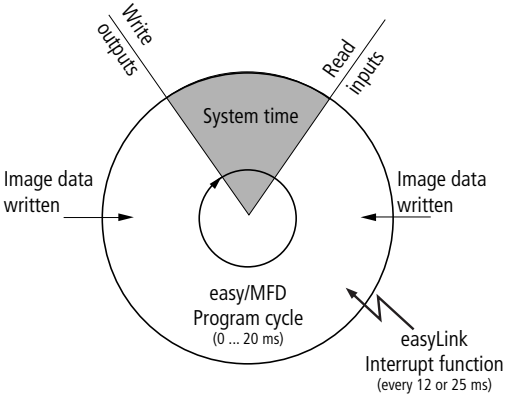
Value (bcd)	Value 10 Bit				Value 1 Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
...								
10	0	0	0	1	0	0	0	0
...								
21	0	0	1	0	0	0	0	1
...								
42	0	1	0	0	0	0	1	0
...								
59	0	1	0	1	1	0	0	1

Table 10: Byte 4: Winter/summer time (value range 00 to 01)

Value (bcd)	Value 10 Bit				Value 1 Bit			
Function	7	6	5	4	3	2	1	0
Winter time	0	0	0	0	0	0	0	0
Summer time	0	0	0	0	0	0	0	1

Read/write image

General notes on working with image data



When writing to image data, it must be remembered that an image (e.g. inputs, outputs,...) used in the easy800/MFD program is also written cyclically by the actual program. The only image data that is unchanged is the data that is not used in the program and is therefore not overwritten in the program cycle. This operating principle also means that an image written via EASYLINK, such as output data is only then output at the physical outputs of the easy800/MFD when the control relay is in Run mode.

Overview0

Operands	Meaning	Read/ write	Comm and	Page
M1 – M16, Q1 – Q8, D1 – D8	"Read status of markers, digital outputs and text display markers"	Read	40	52
P1 – P4, ESC/OK/DEL/ALT	"Read status of P buttons and operating buttons"	Read	3E	55
T1 – T8, C1 – C8, ⓨ1 – ⓨ4, A1 – A8,	"Read status of timing relays, counter relays, time switches and analog value comparators"	Read	3F	56

Read status of markers, digital outputs and text display markers

The following command will read the logical state of all markers M1 to M16, digital outputs Q1 to Q7, text display markers D1 to D8.

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	Read	40	–	1/0	1	0	0	0	0	0	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Status of markers M1 to M8	00	→ table 11								
2	Status of markers M9 to M16	00	→ table 12								
3	Status of digital outputs Q1 to Q8	00	→ table 13								
4	Status of text display markers D1 to D8	00	→ table 14								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 11: Byte 1: Status of markers M1 to M8

	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							

Table 12: Byte 2: Status of markers M9 to M16

	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							

Table 13: Byte 3: Status of digital outputs Q1 to Q8

	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							

Table 14: Byte 4: Status of text display markers D1 to D8

	Bit							
	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							

Read status of P buttons and operating buttons

The following command is used to read the logical state of the digital pushbutton inputs P1 to P4.

The status of the pushbuttons is only displayed if

- a P button is used in the circuit diagram and
- the pushbuttons are activated on the device.

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	Read	3E	–	1/0	0	1	1	1	1	1	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Status of timing relay	00	→ table 15								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 15: Byte 1: Status of pushbuttons

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 pressed/pressed		0/1						
ALT not actuated/actuated	0/1							

Read status of timing relays, counter relays, time switches and analog value comparators

The following command reads the logic state of all timing relays, counters, time switches and analog value comparators.

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	Read	3F	–	1/0	0	1	1	1	1	1	1
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Status of timing relay	00	→ table 16								
2	Counter relay status	00	→ table 17								
3	Time switch status	00	→ table 18								
4	Analog value comparator status	00	→ table 19								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 16: Byte 1: Status of timing relays

	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							

Table 17: Byte 2: Status of the counter relays

	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							

Table 18: Byte 3: Status of time switches

	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
01								0/1
02							0/1	
03						0/1		
04					0/1			
				0				
			0					
		0						
	0							

Table 19: Byte 4: Status of analog value comparators

	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							

Read/write function blocks — Overview

Operands	Meaning	Command	Page
A1 – A8	"Write analog value comparators (function, comparator values)"	22_{hex} – 29_{hex}	60
C1 – C8	"Read counter relay actual value"	49_{hex} – 50_{hex}	63
	"Write counter relay setpoint"	09_{hex} – 10_{hex}	65
I1 – I16	"Reading analog and digital inputs (I7, I8, I1 to I16)"	$3D_{hex}$	68
T1 – T8	"Read timing relay actual value (time base, actual value, switch function)"	41_{hex} – 48_{hex}	71
	"Write timing relay actual value (time base, setpoint, switch function)"	01_{hex} – 08_{hex}	74
1 – 4	"Read time switch (channel, ON time, OFF time)"	$2B_{hex}$ – $3A_{ex}$	81
	"Write time switch (channel, ON time, OFF time)"	12_{hex} – 21_{hex}	85

Write analog value comparators (function, comparator values)

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	A1	22	–	0	0	1	0	0	0	1	0
	A2	23	–	0	0	1	0	0	0	1	1
	A3	24	–	0	0	1	0	0	1	0	0
	A4	25	–	0	0	1	0	0	1	0	1
	A5	26	–	0	0	1	0	0	1	1	0
	A6	27	–	0	0	1	0	0	1	1	1
	A7	28	–	0	0	1	0	1	0	0	0
	A8	29	–	0	0	1	0	1	0	0	1
	Response[1]										
	Write successful	–	C1/41	1/0	1	0	0	0	0	0	1
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Comparators	→ table 35	Invalid								
2	Comparison value for comparison with constant	→ table 36									

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

The comparison values and the function are part of an "*.eas file". If these values are changed, the original "*.eas file" will no longer match the file in EASY6...

Remember this feature when uploading, downloading or comparing "easy" circuit diagrams with EASY-SOFT.

When downloading from the PC the latest version of the "*.eas" is overwritten.

The comparison shows that the circuit diagrams are not identical.

Table 20: Byte 1: Control byte analog value comparator: Comparator

Meaning	Bit							
	7	6	5	4	3	2	1	0
Compare: "≧"								0
Compare: "≦"								1
I7 with I8						0	0	
I7 with constant						0	1	
I8 with constant						1	0	
Fixed			0	0	0			
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Processing	1							

Table 21: Byte 2: Comparison value for comparison with constant

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
63	0	1	1	0	0	0	1	1

Example
The analog value comparator A8 has the following settings:

- Compare I7 < 4.7 V

The master initiates the command to reduce the comparison value to 4.2 V.

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Command: A8	29	0	0	1	0	1	0	0	1
	Response: Write successful	–	0	1	0	0	0	0	0	1
1	Comparators	→	1	0	0	0	0	0	1	1
2	Comparison value for comparison with constant	2A	0	0	1	0	1	0	1	0

The slave responds with the following telegram:

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Response: Write successful	41	0	1	0	0	0	0	0	1
1	Comparators	Invalid								
2	Comparison value for comparison with constant	00								

Read counter relay actual value

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	C1	49	–	1/0	1	0	0	1	0	0	1
	C2	4A	–	1/0	1	0	0	1	0	1	0
	C3	4B	–	1/0	1	0	0	1	0	1	1
	C4	4C	–	1/0	1	0	0	1	1	0	0
	C5	4D	–	1/0	1	0	0	1	1	0	1
	C6	4E	–	1/0	1	0	0	1	1	1	0
	C7	4F	–	1/0	1	0	0	1	1	1	1
	C8	50	–	1/0	1	0	1	0	0	0	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Invalid	00	→	x	x	x	x	x	x	x	x
2	Counter relay actual value (low byte)	00	→ table 22								
3	Counter relay actual value (high byte)	00	→ table 23								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 22: Byte 2: Counter relay actual value (low byte)

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
FF	1	1	1	1	1	1	1	1

Table 23: Byte 3: Counter relay actual value (high byte)

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
FF	1	1	1	1	1	1	1	1

Example

The master initiates the command for reading counter relay C5:

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Command: C5	4D	1	1	0	0	1	1	0	1
1 – 6		00								

The slave responds with the following values:

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Response: Read successful	42	1	1	0	0	0	0	1	0
1	Invalid		x	x	x	x	x	x	x	x
2	Counter relay actual value (low byte)	67	0	1	1	0	0	1	1	1
3	Counter relay actual value (high byte)	12	0	0	0	1	0	0	1	0

Actual value of counter relay C5: 4711
(value $1267_{hex} = 4711_{dec}$)

Write counter relay setpoint

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	C1	09	–	1/0	0	0	0	1	0	0	1
	C2	0A	–	1/0	0	0	0	1	0	1	0
	C3	0B	–	1/0	0	0	0	1	0	1	1
	C4	0C	–	1/0	0	0	0	1	1	0	0
	C5	0D	–	1/0	0	0	0	1	1	0	1
	C6	0E	–	1/0	0	0	0	1	1	1	0
	C7	0F	–	1/0	0	0	0	1	1	1	1
	C8	10	–	1/0	0	0	1	0	0	0	0
	Response[1]										
	Write successful	–	C1/41	1/0	1	0	0	0	0	0	1
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Parameters menu	→ table 24	00								
2	Setpoint value (low byte)	→ table 25	00								
3	Setpoint value (high byte)	→ table 26	00								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Value range of the counter values: 0000 to 9999

→ Keep within the value range.

The value is part of an "*.eas file". If these values are changed, the original "*.eas file" will no longer match the file in EASY6...

Remember this feature when uploading, downloading or comparing "easy" circuit diagrams with EASY-SOFT.

When downloading from the PC the latest version of the "*.eas" is overwritten.

The comparison shows that the circuit diagrams are not identical.

Table 24: Byte 1: Counter relay control byte

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

Table 25: Byte 2: Counter value (low byte)

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
FF	1	1	1	1	1	1	1	1

Table 26: Byte 3: Counter value (high byte)

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
FF	1	1	1	1	1	1	1	1

Example: Change counter relay setpoint
The master initiates the command to change the setpoint value of counter relay C8 to 1542:

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Command: C8	10	0	0	0	1	0	0	0	0
1	Appears in the parameter menu	→ Bit value	1	0	0	0	0	0	0	0
2	Setpoint value (low byte)	06	0	0	0	0	0	1	1	0
3	Setpoint value (high byte)	06	0	0	0	0	0	1	1	0

$0606_{hex} = 1542_{dec}$

The slave responds with the following telegram:

Byte	Meaning		Bit							
		Slave	7	6	5	4	3	2	1	0
0	Response: Write successful	41	0	1	0	0	0	0	0	1
1 – 3		00								

Reading analog and digital inputs (I7, I8, I1 to I16)

The following command is used to read the values of both analog inputs I7, I8 (only EASY...-DC-..) and the logical states of the digital inputs I1 to I16.

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	Read	3D	–	0	0	1	1	1	1	0	1
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Analog value of I7	00	→ table 27								
2	Analog value of I8	00	→ table 28								
3	Status of inputs I1 to I8	00	→ table 29								
4	Status of inputs I9 to I12, I15, I16	00	→ table 30								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 27: Byte 1: Analog value I7

Analog value I7(hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
64	0	1	1	0	0	1	0	0

Examples: $0A_{hex}$ = 1 V, 32_{hex} = 5 V, 64_{hex} = 10 V

Table 28: Byte 2: Analog value I8

Analog value I8 (hex)	Bit							
	7	6	5	4	3	2	1	0
00_{hex}	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
64_{hex}	0	1	1	0	0	1	0	0

Examples: $0A_{hex}$ = 1 V, $3C_{hex}$ = 6 V, $5A_{hex}$ = 9 V

Table 29: Byte 3: Status of inputs I1 to I8

Value	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							

Value 0 = switched off, Value 1 = switched on

Table 30: Byte 4: Status of inputs I9 to I12, I15, I16

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

Value 0 = switched off, Value 1 = switched on

→ I13 = 0, I14 = 0
If I14 = 1, EASY204-DP has disconnected from the basic unit.

I15, I16 are the short-circuit signals for EASY...-DC-.. transistor versions.

Read timing relay actual value (time base, actual value, switch function)

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	T1	41	–	1/0	1	0	0	0	0	0	1
	T2	42	–	1/0	1	0	0	0	0	1	0
	T3	43	–	1/0	1	0	0	0	0	1	1
	T4	44	–	1/0	1	0	0	0	1	0	0
	T5	45	–	1/0	1	0	0	0	1	0	1
	T6	46	–	1/0	1	0	0	0	1	1	0
	T7	47	–	1/0	1	0	0	0	1	1	1
	T8	48	–	1/0	1	0	0	1	0	0	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Timing relay, time base, control status	00	→ table 31								
2	Time actual value (low byte)	00	→ table 32								
3	Time actual value (high byte)	00	→ table 33								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 31: Byte 1: Timing relay function, time base, control status

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
Single pulse						1	0	0
Flashing						1	0	1
Time base "s"				0	0			
Time base "M:S"				0	1			
Time base "H:M"				1	0			
Not assigned			0					
Appears in the parameter menu		0						
Does not appear in the parameter menu		1						
Timing relay not processed by operating system	0							
Timing relay processed by operating system	1							

Table 32: Byte 2: Time actual value (low byte)

Value (hex)	Bit							
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
FF	1	1	1	1	1	1	1	1

Table 33: Byte 3: Time actual value (high byte)

Value (hex)	Bit 7	6	5	4	3	2	1	0
00_{hex}	0	0	0	0	0	0	0	0
FF_{hex}	1	1	1	1	1	1	1	1

Example

The master initiates the command for reading timing relay T1:

Byte	Meaning	Value (hex)	Bit 7	6	5	4	3	2	1	0
0	Command: T1	41	1	1	0	0	0	0	0	1
1 – 3		00								

The slave responds with the following values:

Byte	Meaning	Value (hex)	Bit 7	6	5	4	3	2	1	0
0	Response: Read successful	C2	1	1	0	0	0	0	1	0
1	Trigger coil activated, M:S time base, on-delayed, Parameter display +	→	1	0	0	0	1	0	0	0
2	Time actual value (low byte)	10	0	0	0	1	0	0	0	0
3	Time actual value (high byte)	0E	0	0	0	0	1	1	1	0

Value Set time = $0E10_{hex}$ = 3600

3600 s = 60:00 M:S

Write timing relay actual value (time base, setpoint, switch function)

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	T1	01	–	1/0	0	0	0	0	0	0	1
	T2	02	–	1/0	0	0	0	0	0	1	0
	T3	03	–	1/0	0	0	0	0	0	1	1
	T4	04	–	1/0	0	0	0	0	1	0	0
	T5	05	–	1/0	0	0	0	0	1	0	1
	T6	06	–	1/0	0	0	0	0	1	1	0
	T7	07	–	1/0	0	0	0	0	1	1	1
	T8	08	–	1/0	0	0	0	1	0	0	0
	Response[1]										
	Write successful	–	C1/41	1/0	1	0	0	0	0	0	1
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Timing relay function, time base, Parameters menu	→ table 34	Invalid								
2	Time value --.xx with time base "S"	→ table 35									
3	Time value xx.-- with time base "S" or "--:xx" with time base M:S	→ table 36									
4	Time value xx.-- with time base "M:S" or "--:xx" with time base "H:M"	→ table 37									
5	Time value "xx.--" with time base "H:M"	→ table 38									
6	Hour value in days	00	→ table 39								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

→ Time values over 60s are converted to minutes.
Time values over 60 min. are converted to hours.
Time values over 24 h are converted to days.

The value range of the times and the timing relay setpoint are part of an "*.eas file". If these values are changed, the original "*.eas file" will no longer match the file in EASY6...

Remember this characteristic when uploading, downloading or comparing "easy" circuit diagrams with EASY-SOFT.

When downloading from the PC the latest version of the "*.eas" is overwritten.

The comparison shows that the circuit diagrams are not identical.

Value range of the time values

- "S" 00.00 to 99.99
- "M:S" 00:00 to 99:59 (M = 00 to 99, S = 00 to 59)
- "H:M" 00:00 to 99:59 (H = 00 to 99, M = 00 to 59)

→ Only the bytes reserved for the required time base should be used.

Table 34: Byte 1: Timing relay control byte

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
Single pulse						1	0	0
Flashing						1	0	1
Time base "s"				0	0			
Time base "M:S"				0	1			
Time base "H:M"				1	0			
Not assigned			0					
Does not appear in the parameter menu		1						
Appears in the parameter menu		0						
Processing	1							

Table 35: Byte 2: Time value "--.xx" time base "S"

Value (bcd)	Value --.x- Bit				Value --.-x Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
05	0	0	0	0	0	1	0	1
17	0	0	0	1	0	1	1	1
42	0	1	0	0	0	0	1	0
99	1	0	0	1	1	0	0	1

Table 36: Byte 3: Time value "xx.--" Time base "S"
Time value "xx.--", Time base "M:S"

Value (bcd)	Value x-.-- "S" Value --:x- "-:S" Bit				Value -x.-- "S" Value --:-x "-:S" Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
05	0	0	0	0	0	1	0	1
17	0	0	0	1	0	1	1	1
42	0	1	0	0	0	0	1	0
99	1	0	0	1	1	0	0	1

Table 37: Byte 4: Time value "xx.--" Time base "M:S"
Time value "--:xx", Time base "H:M"

Value (bcd)	Value x-.-- "M:-" Value --:x- "-:M" Bit				Value -x.-- "M:-" Value --:-x "-:M" Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
05	0	0	0	0	0	1	0	1
17	0	0	0	1	0	1	1	1
42	0	1	0	0	0	0	1	0

Table 38: Byte 5: Time value "xx:--" Time base "H:M"

Value (bcd)	Value x-:-- "H:-" Bit				Value -x:-- "H:-" Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
05	0	0	0	0	0	1	0	1
17	0	0	0	1	0	1	1	1

Table 39: Byte 6: Time value in days

Value (bcd)	Bit				Days value Bit			
	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
04	0	0	0	0	0	1	0	0

Example 1: timing relay
The master initiates the command for writing timing relay T8 with the following values:

Switch function	On-delayed
Time base	"s"
Setpoint time	50

T8 is to be assigned the time base "M:S" and the setpoint time 30 minutes, 25 seconds.

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Command: T8	08_{hex}	0	0	0	0	1	0	0	0
1	Timing relay function, time base, Parameters menu	→	1	0	0	0	1	0	0	0
2	Time value --.xx with time base "S"	→	0	0	0	0	0	0	0	0
3	Time value xx.-- with time base "S" or "--:xx" with time base M:S	25_{bcd}	0	0	1	0	0	1	0	1
4	Time value xx.-- with time base "M:S" or "--:xx" with time base "H:M"	30_{bcd}	0	0	1	1	0	0	0	0
5	Time value "xx.--" with time base "H:M"	→	0	0	0	0	0	0	0	0
6	Hour value in days	→	0	0	0	0	0	0	0	0

The slave responds with the following values:

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Response: Write successful	$C1_{hex}$	0	1	0	0	0	0	0	1
1	Invalid	→	x	x	x	x	x	x	x	x
2 – 6	Same content as with master									

Example 2: timing relay

The master initiates the write command for timing relay T1 with the following values:

Switch function	Off-delayed
Time base	"M:S"
Setpoint time	10:30

T1 is to be assigned the time base "H:M" and the setpoint time 95 hours, 53 minutes. 95 hours = 3 days, 19 hours

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Command: T1	01_{hex}	1	0	0	0	0	0	0	1
1	Timing relay function, time base, Parameters menu	→ Bit value	1	0	0	1	0	0	0	1
2	Time value --.xx with time base "S"	→ Bit value	0	0	0	0	0	0	0	0
3	Time value xx.-- with time base "S" or "--:xx" with time base M:S	25_{bcd}	0	0	0	0	0	0	0	0
4	Time value xx.-- with time base "M:S" or "--:xx" with time base "H:M"	53_{bcd}	0	1	0	1	0	0	1	1
5	Time value "xx.--" with time base "H:M"	23_{bcd}	0	0	1	0	0	0	1	1
6	Hour value in days	03_{bcd}	0	0	0	0	0	0	1	1

The slave responds with the following values:

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Response: Write successful	$C1_{hex}$	1	1	0	0	0	0	0	1
1	Invalid	→	x	x	x	x	x	x	x	x
2 – 6	Same content as with master									

Read time switch (channel, ON time, OFF time)

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	01 channel A	2B	–	1/0	0	1	0	1	0	1	1
	01 channel B	2C	–	1/0	0	1	0	1	1	0	0
	01 channel C	2D	–	1/0	0	1	0	1	1	0	1
	01 channel D	2E	–	1/0	0	1	0	1	1	1	0
	02 channel A	2F	–	1/0	0	1	0	1	1	1	1
	02 channel B	30	–	1/0	0	1	1	0	0	0	0
	02 channel C	31	–	1/0	0	1	1	0	0	0	1
	02 channel D	32	–	1/0	0	1	1	0	0	1	0
	03 channel A	33	–	1/0	0	1	1	0	0	1	1
	03 channel B	34	–	1/0	0	1	1	0	1	0	0
	03 channel C	35	–	1/0	0	1	1	0	1	0	1
	03 channel D	36	–	1/0	0	1	1	0	1	1	0
	04 channel A	37	–	1/0	0	1	1	0	1	1	1
	04 channel B	38	–	1/0	0	1	1	1	0	0	0
	04 channel C	39	–	1/0	0	1	1	1	0	0	1
	04 channel D	3A	–	1/0	0	1	1	1	0	1	0
	Response[1]										
	Read successful	–	C2/42	1/0	1	0	0	0	0	1	0
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Invalid	00	→	x	x	x	x	x	x	x	x
2	Weekday, Parameter menu display	00	→ table 40								
3	Minute (switch point ON)	00	→ table 41								
4	Hour (switch point ON)	00	→ table 42								
5	Minute (switch point OFF)	00	→ table 43								
6	Hour (switch point OFF)	00	→ table 44								

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

Table 40: Byte 2: Weekday, starting, ending, Parameters menu

	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ON day								
None 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
OFF day								
None 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			
Switch time								
ON > OFF		1						
ON < OFF		0						
Appears in the Parameters menu								
No	1							
Yes	0							

Table 41: Byte 3: Minute (ON time)

Value (bcd)	Bit 7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
59	0	1	0	1	1	0	0	1

Table 42: Byte 4: Hour (ON time)

Value (bcd)	Bit 7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
23	0	0	1	0	0	0	1	1

Table 43: Byte 5: Minute (OFF time)

Value (bcd)	Bit 7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
59	0	1	0	1	1	0	0	1

Table 44: Byte 6: Hour (OFF time)

Value (bcd)	Bit 7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0
....	...	...	...	...	...	...	...	...
23	0	0	1	0	0	0	1	1

Example
The master initiates the command to read the values of channel "A" of 4:

Byte	Meaning	Value (hex)	Bit							
			7	6	5	4	3	2	1	0
0	Command: 4 channel A	37	0	0	1	1	0	1	1	1
1 – 6		00								

The slave responds with the following values:

- Day: Monday (001) to Friday (101)
- ON: 19:00
- OFF: 06:30
- Switch time ON > OFF (1)
- Channel appears in the Parameters menu

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Response: : Read successful	42	0	1	0	0	0	0	1	0
1	Invalid		x	x	x	x	x	x	x	x
2	Weekday, Parameter menu display	69_{bcd}	0	1	1	0	1	0	0	1
3	Minute (switch point ON)	00_{bcd}	0	0	0	0	0	0	0	0
4	Hour (switch point ON)	19_{bcd}	0	0	0	1	1	0	0	1
5	Minute (switch point OFF)	30_{bcd}	0	0	1	1	0	0	0	0
6	Hour (switch point OFF)	06_{bcd}	0	0	0	0	0	1	1	0

Write time switch (channel, ON time, OFF time)

Telegram structure

Byte	Meaning	Value (hex), sent by		Bit							
		Master	Slave	7	6	5	4	3	2	1	0
0	Command [1]										
	01 channel A	12	–	1/0	0	0	1	0	0	1	0
	01 channel B	13	–	1/0	0	0	1	0	0	1	1
	01 channel C	14	–	1/0	0	0	1	0	1	0	0
	01 channel D	15	–	1/0	0	0	1	0	1	0	1
	02 channel A	16	–	1/0	0	0	1	0	1	1	0
	02 channel B	17	–	1/0	0	0	1	0	1	1	1
	02 channel C	18	–	1/0	0	0	1	1	0	0	0
	02 channel D	19	–	1/0	0	0	1	1	0	0	1
	03 channel A	1A	–	1/0	0	0	1	1	0	1	0
	03 channel B	1B	–	1/0	0	0	1	1	0	1	1
	03 channel C	1C	–	1/0	0	0	1	1	1	0	0
	03 channel D	1D	–	1/0	0	0	1	1	1	0	1
	04 channel A	1E	–	1/0	0	0	1	1	1	1	0
	04 channel B	1F	–	1/0	0	0	1	1	1	1	1
	04 channel C	20	–	1/0	0	1	0	0	0	0	0
	04 channel D	21	–	1/0	0	1	0	0	0	0	1
	Response[1]										
	Write successful	–	C1/41	1/0	1	0	0	0	0	0	1
	Command rejected	–	C0/40	1/0	1	0	0	0	0	0	0
1	Weekday, Parameter menu display	00	→ table 40								
2	Minute (switch point ON)	00	→ table 41								
3	Hour (switch point ON)	00	→ table 42								
4	Minute (switch point OFF)	00	→ table 43								
5	Hour (switch point OFF)	00	→ table 44								
6	Not used										

1) Note the Byte 0 Bit 7 data exchange procedure, → page 45.

The values for minute and hour of the switch points are part of an "*.eas file". If these values are changed, the original "*.eas file" will no longer match the file in EASY6...

Remember this feature when uploading, downloading or comparing "easy" circuit diagrams with EASY-SOFT.

When downloading from the PC the latest version of the "*.eas" is overwritten.

The comparison shows that the circuit diagrams are not identical.

Example
The master initiates the command to write the following data to channel C of ▯2:

- Day: Tuesday (010) to Saturday (110)
- ON: 10:00
- OFF: 17:30
- Switch point ON < OFF (0)
- Channel does not appear in the Parameters menu (1)

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Command: ▯2 channel C	18_{hex}	0	0	0	1	1	0	0	0
1	Weekday, Parameter menu display	$B2_{hex}$	1	0	1	1	0	0	1	0
2	Minute (switch point ON)	00_{bcd}	0	0	0	0	0	0	0	0
3	Hour (switch point ON)	10_{bcd}	0	0	0	1	0	0	0	0
4	Minute (switch point OFF)	30_{bcd}	0	0	1	1	0	0	0	0
5	Hour (switch point OFF)	17_{bcd}	0	0	0	1	0	1	1	1
6	Not used									

The slave responds with the following telegram:

Byte	Meaning	Value	Bit							
			7	6	5	4	3	2	1	0
0	Response: Write successful	41_{hex}	0	1	0	0	0	0	0	1
1 – 6		00								

7 Control commands for easy700

Data exchange procedure

The "Control commands 9 bytes" module allows extended data exchange of the easy700 on the PROFIBUS-DP communication bus. This allows you to transfer services from the following areas:

- "Read/write date and time" (page 92)
- "Read/write image data" (page 96) and
- "Read/write function block data" (page 119).

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

Attention!

Whilst a control command is being executed, the input and output data will remain in the state before the control command was called. Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

Caution!
Only those values specified for the command code should be used.

Check the values that you write in order to avoid malfunctions.

Requirement:
The "Control commands 9 byte" module must have been selected.

The master initiates the data exchange of the control commands and the addressed slave responds.

During communication 9 data bytes (byte 0 = toggle byte, bytes 1 to 8 information bytes) are sent via PROFIBUS.

The basic telegram structure is shown in the following diagram.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8

Byte 0 – Toggle byte
Byte 0 is used to activate the sending of a control command with the toggle function.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
01_{hex}/ 861_{hex}	0/1	0	0	0	0	0	0	1
	Toggle bit	fixed						

Procedure

- ▶ To send a command, bit 7 seven must be toggled, i.e. set either from 1 to 0 or from 0 to 1.
- ▶ Then poll the toggle bit for the coupling modules response until it has the same status as the toggle bit sent. This status indicates to the master that the response to the sent command is valid.
- ▶ Do not send a new command until you have received a response (changing of the toggle bit), otherwise the response of the previous command will be overwritten before it can be read.

In order to use input/output data and control commands simultaneously:

Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

All specified commands and parameters must be transferred in hexadecimal format.

The following tables show the different control commands possible. These essential control commands fall into three essential categories – real-time clock, image and function blocks.

Read/write date and time

→ Please also note the relevant description of the real-time clock provided in the easy700 manual (AWB2528-1508GB).

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command		
	Read	93	–
	Write	B3	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	05	05
3	Index	0 – 2[1]	0 – 2[1]
4 – 8	Data 1 – 4	depending on index, → table 45	

1) 0 = Time/date, → table 45
1 = Summer time, → table 46
2 = Winter time, → table 47

Table 45: Index 0 – date and time of real-time clock

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

Table 46: Index 1 – Summer time

Byte	Contents		Value (hex)
4	Data 1	Area	
		None	00
		Rule	01
		Automatic EU	02
		Automatic GB	03
		Automatic US	04
for "Area" = "Rule":			
5	Data 2	Summer time switching rule	→ table 48
6	Data 3		
7	Data 4		
8	Data 5		

Table 47: Index 2 – Winter time
(only valid if Area = "Rule" selected)

Byte	Contents		Value (hex)
4	Data 1	Area = Rule	01
5 – 8	Data 2 – 5	Winter time switching rule	→ table 48

Switching rule bit array

→ Please also read the detailed description in the easy700 manual (AWB2528-1508GB).

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

Table 48: Switching rule bit array

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

Read/write image data

Please also observe the relevant description of possible image data provided in the easy700 manual (AWB2528-1508GB) or in the EASY-SOFT Help.

The latest edition of the manual is available as a PDF file from the Internet at: http://www.moeller.net/support: Search Term: AWB2528-1508GB

The information provided in section "General notes on working with image data" on page 50 also applies to easy700.

Overview

Operands	Meaning	Read/ write	Type	Page
A1 – A16	"Analog value comparators/threshold comparators: A1 – A16"	Read	8C	98
C1 – C16	"Counters: C1 – C16"	Read	EE	99
D1 – D16	"Text function blocks: D1 – D16"	Read	94	100
I1 – I16	"Local inputs: I1 – I16"	Read	84	101
IA1 – IA4	"Local analog inputs: IA1 – IA4"	Read	8C	103
M1 – M16, N1 – N16	"Markers: M1 – M16/N1 – N16"	Write	86/87	105
M1 – M16, N1 – N16	"Markers: M1 – M16/N1 – N16"	Read	86/87	107
O1 – O4	"Operating hours counters: O1 – O4"	Read	EF	109
P1 – P4	"Local P buttons: P1 – P4"	Read	8A	110
Q1 – Q8	"Local outputs: Q1 – Q8"	Read	85	112
R1 – R16/ S1 – S8	"Inputs/outputs of EASY-LINK: R1 – R16/ S1 – S8"	Read	88/89	113
T1 – T16	"Timing relays: T1 – T16"	Read	ED	115
Y1 – Y4	"Year time switch: Y1 – Y8"	Read	91	116
Z1 – Z3	"Master reset: Z1 – Z3"	Read	93	117
H1 – H4	7-day time switch: 1 – 8	Read	90	118

Analog value comparators/threshold comparators: A1 – A16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	8C	8C
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 49
6	Data 2 (Low Byte)	00	→ table 49
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 49: Byte 5 to 6: Data 1 to 2

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

Counters: C1 – C16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	EE	EE
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 59
6	Data 2 (Low Byte)	00	→ table 59
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 50: Byte 5 to 6: Data 1 to 2

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

Text function blocks: D1 – D16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	94	94
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 51
6	Data 2 (High Byte)	00	→ table 51
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 51: Byte 5 to 6: Data 1 to 2

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

Local inputs: I1 – I16

This command string enables you to read the local inputs of the easy700 basic unit. The relevant input word is stored in Intel format.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1)]
2	Len	02	02
3	Type	84	84
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 52
6	Data 2 (High Byte)	00	→ table 52
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 52: Byte 5 to 6: Data 1 to 2

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

Local analog inputs: IA1 – IA4

The analog inputs on the easy700 basic unit (I7, I8, I11, I12) can be read directly via PROFIBUS-DP. The 16-bit value is transferred in Intel format (Low Byte first).

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	02	02
3	Type	8C	8C
4	Index	00 – 03[2]	00 – 03[2]
5	Data 1 (Low Byte)	00	→ table 53
6	Data 2 (High Byte)	00	→ table 53
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

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

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

Table 53: Example telegram for reading the value at the analog input

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response: Read successful	–	C2
2	Len	02	02
3	Type	8C	8C
4	Index	02[1]	02[1]
5	Data 1	00	4B
6	Data 2	00	03
7	Data 3	00	00
8	Data 4	00	00

1) 02 = Analog input I11

Byte 5 – Data 1 (Low Byte): $4B_{hex}$
Byte 6 – Data 2 (High Byte): 03_{hex}
→ corresponding 16-bit value: $034B_{hex} = 843$

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

$$\frac{10\ V}{1023} \times \text{IO bit value} \quad => \quad \frac{10\ V}{1023} \times 843 = 8.24\ V$$

Markers: M1 – M16/N1 – N16

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Write	8C	–
	Response:		
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Len	01	01
3	Type[2]		
	With M marker	86	86
	With N marker	87	87
4	Index[2]	00 – 0F	00 – 0F
5	Data 1 (Low Byte)[3]	00/01	00/01
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

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

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

Example:
Marker M13 is set.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Write	8C	–
	Response:		
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Len	01	01
3	Type		
	M marker	86	86
4	Index	0C	0C
5	Data 1	01	00
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Markers: M1 – M16/N1 – N16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type		
	M marker	86	86
	N marker	87	87
4	Index[2]	00	00
5	Data 1 (Low Byte)	00	→ table 54
6	Data 2 (Low Byte)	00	→ table 54
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

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

Table 54: Byte 5 to 6: Data 1 to 2

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

Example:
The N markers are read:

Byte	Meaning	Value (hex), sent by	
		Master	**Slave**
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type		
	N marker	87	87
4	Index	00	00
5	Data 1 (Low Byte)	00	04
6	Data 2 (Low Byte)	00	84
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

The markers N3, N11 and N16 are set.

Operating hours counters: O1 – O4

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	EF	EF
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 55
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 55: Byte 5: Data 1

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

Local P buttons: P1 – P4

The local P buttons are the display cursor buttons of the easy700 basic unit. You can scan the buttons in both RUN and STOP mode.

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

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	8A	8A
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 56
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 56: Byte 5: Data 1

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

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

Local outputs: Q1 – Q8

The local outputs can be read directly via the PROFIBUS-DP fieldbus.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1)]
2	Len	01	01
3	Type	85	85
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 57
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 57: Byte 5: Data 1

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

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

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

This service allows you to read the local R and S data and the data of the NET stations (1 – 8) transferred via EASYLINK, again from the relevant easy700 image.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1)]
2	Len	01	01
3	Type		
	for R data	88	88
	for S data	89	89
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 58
6	Data 2 (Low Byte)	00	→ table 58
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 58: Byte 5 to 6: Data 1 to 2

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

Timing relays: T1 – T16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	ED	ED
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 59
6	Data 2 (Low Byte)	00	→ table 59
7 – 8	Data 3 – 4	00	00

1) Possible causes → page 138

Table 59: Byte 5 to 6: Data 1 to 2

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

Year time switch: Y1 – Y8

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	91	91
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 60
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 60: Byte 5: Data 1

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

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

Master reset: Z1 – Z3

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	93	93
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 61
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 61: Byte 5: Data 1

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

7-day time switch: ⏲1 – ⏲8

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command: Read	88	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0[1]
2	Len	01	01
3	Type	90	90
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 62
6 – 8	Data 2 – 4	00	00

1) Possible causes → page 138

Table 62: Byte 5: Data 1

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

Example:
Data 1 = 2_{hex} → ⏲3 is active.

Read/write function block data

→ Please also observe the relevant description of the function blocks provided in the easy700 manual (AWB2528-1508GB) or in the EASY-SOFT Help.

General notes

Always note the following when working with function blocks:

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

Overview

Operands	Meaning	Read/ write	Type	Page
A1 – A16	"Analog value comparator/threshold comparator: A1 – A16"	Read/write	8D	120
C1 – C16	"Counter relays: C1 – C16"	Read/write	8F	123
O1 – O4	"Operating hours counters: O1 – O4"	Read/write	92	126
T1 – T16	"Timing relays: T1 – T16"	Read/write	8E	128
Y1 – Y8	"Year time switch: Y1 – Y8"	Read/write	A2	132
1 – 8	7-day time switch: 1 – 8	Read/write	A1	135

Analog value comparator/threshold comparator: A1 – A16

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	8D	8D
3	Instance[2]	00 – 0F	00 – 0F
4	Index	→ table 63	
5 – 8	Data 1 – 4	depending on index, → table 64	

1) Possible causes → page 138

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

Table 63: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 64		×	
01	Control byte → table 65		×	
02	Comparison value 1	I1[2]	×	c[1]
03	Comparison value 2	I2[2]	×	c[1]
04	Gain factor for I1 (I1 = F1 × I1)	F1[2]	×	c[1]
05	Gain factor for I2 (I2 = F2 × I2)	F2[2]	×	c[1]
06	Offset for value I1 (I1 = OS + actual value at I1)	OS[2]	×	c[1]
07	Switching hysteresis for value I2	HY[2]	×	c[1]

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

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

Table 64: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0xA3, Data 2 (Byte 6) = 0x03
→ Resulting 16-bit value = $03A3_{hex}$

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

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

Table 65: Index 01 – Control byte

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

1) Status 1 if comparison condition is fulfilled.

Counter relays: C1 – C16

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	8F	8F
3	Instance[2]	00 – 0F	00 – 0F
4	Index	→ table 66	
5 – 8	Data 1 – 4	depending on index, → table 67	

1) Possible causes → page 138

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

Table 66: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 67		×	
01	Control byte → table 68		×	
02	Actual value	S1[2]	×	c[1]
03	Counter setpoint 2	S2[2]	×	c[1]

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

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

Table 67: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0x07

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

Table 68: Index 01 – Control byte

Data 1	Bit	7	6	5	4	3	2	1	0
FB output		–	–	–	–	C4	RE3	D2	Q11

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

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

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

Explanation:

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

Counter status = 786

Operating hours counters: O1 – O4

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	92	92
3	Instance[2]	00 – 03	00 – 03
4	Index	→ table 69	
5 – 8	Data 1 – 4	depending on index, → table 70	

1) Possible causes → page 138

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

Table 69: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 70		×	
01	Control byte → table 71		×	
02	Actual value	S1[2]	×	c[1]
03	Counter setpoint 2	S2[2]	×	c[1]

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

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

Table 70: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0x01

Meaning:
The values appear in the Parameter menu.

Table 71: Index 01 – Control byte

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

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

Example:
Index 02/03

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

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

Timing relays: T1 – T16

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	8E	8E
3	Instance[2]	00 – 0F	00 – 0F
4	Index	→ table 72	
5 – 8	Data 1 – 4	depending on index, → table 73	

1) Possible causes → page 138

2) easy provides 16 timing relays T1 to T16 for use as required. These can be addressed using the instance (0 – F).

Table 72: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 73		×	
01	Control byte → table 74		×	
02	Actual value 1	T	×	c[1]
03	Time setpoint 1	S1[2]	×	c[1]
04	Time setpoint 2	S2[2]	×	c[1]

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

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

Table 73: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0xAC

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

Table 74: Index 01 – Control byte

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

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

Example:
The time setpoint 1 is to be read:

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

Explanation:

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

Meaning depending on set time base:

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

Year time switch: Y1 – Y8

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	A2	A2
3	Instance[2]	00 – 07	00 – 07
4	Index	→ table 75	
5 – 8	Data 1 – 4	depending on index, → table 76	

1) Possible causes → page 138

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

Table 75: Operand overview

Index (hex)	Operand	Read	Write
00	Parameters → table 76	×	
01	Control byte → table 77	×	
	Channel A	×	c[1]
11	Time point ON	×	c[1]
12	Time point OFF	×	c[1]
	Channel B	×	c[1]
21	Time point ON	×	c[1]
22	Time point OFF	×	c[1]
	Channel C	×	c[1]
31	Time point ON	×	c[1]
32	Time point OFF	×	c[1]
	Channel D	×	c[1]
41	Time point ON	×	c[1]
42	Time point OFF	×	c[1]

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

2) The switch points are transferred in data bytes Data 1 – Data 3.

Table 76: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0x03 → The values of the year time switch of channel A and B in the parameter menu.

Table 77: Index 01 – Control byte

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

1) Status 1, if the count condition is fulfilled.

Channel A, Index 11/12
Index 0x11 channel A ON time
Index 0x12 channel A OFF time
Data 1 (Byte 5) – Day
Data 2 (Byte 6) – Month
Data 3 (Byte 7) – Year

Example:
The year time switch channel A is to be activated on the 21.04.2004.

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

The year time switch channel B is to be deactivated on the 05.11.2012.

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

7-day time switch: 1 – 8

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 90	
1	Command:		
	Read	89	–
	Write	8D	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0[1]
2	Type	A1	A1
3	Instance[2]	00 – 07	00 – 07
4	Index	→ table 78	→ table 78
5 – 8	Data 1 – 4	depending on index, → table 79	

1) Possible causes → page 138

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

Table 78: Operand overview

Index (hex)	Operand		Read	Write
00	Parameters → table 79		×	
01	Control byte → table 80		×	
11	Channel A	Day on/off	×	c[1]
12		On time	×	c[1]
13		Off time	×	c[1]
21	Channel B	Day on/off	×	c[1]
22		On time	×	c[1]
23		Off time	×	c[1]
31	Channel C	Day on/off	×	c[1]
32		On time	×	c[1]
33		Off time	×	c[1]
41	Channel D	Day on/off	×	c[1]
42		On time	×	c[1]
43		Off time	×	c[1]

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

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

Table 79: Index 00 – Parameters

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

Example:
Data 1 (Byte 5) = 0x03

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

Table 80: Index 01 – Control byte

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

1) Status 1, if the count condition is fulfilled.

Channel A, Index 11/12/13
Index 0x11 channel A Weekday on/off
Data 1 (Byte 5) – Weekday on
Data 2 (Byte 6) – Weekday off
0x01 = Sunday ... 0x07 = Saturday

The 16-bit value equals 0x00 if the channel is not used.

Index 0x12 – On time (2 Byte)
Index 0x13 – Off time (2 Byte)
Data 1 (Byte 5) – Hour
Data 2 (Byte 6) – Minute

Example: On time at 13:43 p.m.
Data 1 = 0x0D
Data 2 = 0x2B

Analysis – error codes via EASY-LINK

The easy700 basic unit will return a defined error code in the event of an incorrectly selected operating mode or an invalid telegram. The error code transferred has the following structure:

Telegram structure

Byte	Meaning	Slave transmits (value hex)
0	Toggle byte	→ page 90
1	Response	
	Command rejected	C0
2	Type	00
3	Instance	00
4	Index	00
5	Error code	→ table 81

Table 81: Error codes

Error code	Description
0x01	Unknown telegram transmitted.
0x02	Unknown object transmitted.
0x03	Unknown command transmitted.
0x04	Invalid instance transmitted.
0x05	Invalid parameter set transmitted.
0x06	An attempt was made to write to a variable that is not a constant.
0x0C	The device is in an invalid device mode. STOP → RUN or RUN → STOP
0x0D	Invalid display access. Exit the menu level so that the status display is showing in the display. The clock cannot be written to.
0xF0	Attempt made to control an unknown parameter.
0xF1	Impermissible value

8 easy800/MFD Control Commands

Data exchange procedure

The Control commands 9 bytes module allows extended data exchange of the easy800 and the MFD-Titan on the PROFIBUS-DP communication bus. This allows you to transfer services from the following areas:

- "Read/write date and time" (page 144)
- "Read/write image data" (page 148) and
- "Read/write function block data" (page 169).

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

Attention!

Whilst a control command is being executed, the input and output data will remain in the state before the control command was called. Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

Caution!
Only those values specified for the command code should be used.

Check the values that you write in order to avoid malfunctions.

Requirement:
The "Control commands 9 byte" module must have been selected.

The master initiates the data exchange of the control commands and the addressed slave responds.

During communication 9 data bytes (byte 0 = toggle byte, bytes 1 to 8 information bytes) are sent via PROFIBUS.

The basic telegram structure is shown in the following diagram.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8

Byte 0 – Toggle byte
Byte 0 is used to activate the sending of a control command with the toggle function.

Bit	7	6	5	4	3	2	1	0
01_{hex}/ 861_{hex}	0/1	0	0	0	0	0	0	1
	Toggle bit	fixed						

Procedure
- ▶ To send a command, bit 7 seven must be toggled, i.e. set either from 1 to 0 or from 0 to 1.
- ▶ Then poll the toggle bit for the coupling modules response until it has the same status as the toggle bit sent. This status indicates to the master that the response to the sent command is valid.
- ▶ Do not send a new command until you have received a response (changing of the toggle bit), otherwise the response of the previous command will be overwritten before it can be read.

→ In order to use input/output data and control commands simultaneously:

Only after the "Control commands" data exchange has been completed, will the I/O data be refreshed.

All specified commands and parameters must be transferred in hexadecimal format.

The following tables show the different control commands possible. These essential control commands fall into three essential categories – real-time clock, image and function blocks.

Version history

The following table provides an overview of modifications and new features of the different easy800 device versions:

Effect on easy-Link	easy800, device version		
	From 02	From 04	From 05
Support for complete PDO access			
R data writable	✓	✓	✓
S data readable	✓	✓	✓
Support for complete SDO access			
Function blocks	–	MR, A, AR, BV, C, CF, CH, CI, CP, D, DB, GT, HW, HY, OT, PT, SC, T, BC, BT, DC, FT, LS, NC, PW, ST, VC	
Image data			
Read	–	IW, IA, ID, QW, QA, P, RW, SW, M, MB, MW, MD	
Write	–	QW, QA, M, MB, MW, MD	M, MB, MW, MD
Clock functions	–	✓	✓
Rule option for winter/summer (DST) time change	–	–	✓

Read/write date and time

→ Please also note the relevant description of the real-time clock provided in the easy800 manual.

The latest edition of the manual is available as a PDF file from the Internet at: http://www.moeller.net/support: Search Term: AWB2528-1423GB

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command		
	Read	93	–
	Write	B3	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	05	05
3	Index	00	00
4 – 8	Data 1 – 5		
	Read operation	00	→ table 82
	For write operation	→ table 82	00

Table 82: Byte 4 – 8: Data 1 – 5

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

Winter/summer time, DST

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command		
	Read	93	–
	Write	B3	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	05	05
3	Index	01	01
4 – 8	Data 1 – 5		
	Read operation	00	→ table 83
	For write operation	→ table 83	00

Table 83: Byte 4 – 8: Data 1 – 5

Byte	Contents		Value (hex)
4	Data 1	Area	
		None	00
		Manual	01
		Automatic EU	02
		Automatic GB	03
		Automatic US	04
5	Data 21	Set summer time day (1 to 28, 29, 30, 31 depending on month and year)	00 – 3B
6	Data 31	Set Summer time month (1 to 12)	01 – 1F
7	Data 41	Set winter time day (1 to 28, 29, 30, 31 depending on month and year)	01 – 0C
8	Data 51	Set winter time month (1 to 12)	00 – 63

1) The additional parameters Data 2 to Data 5 for automatic DST change are only relevant if you have set the "Manual" parameter for Data 1.

Example
The real-time clock of the easy800 is to be set to Friday 23.05.2003, 14:36.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Write	B3	–
	Response: Write successful	–	C1
2	Len	05	05
3	Index	00	00
4	Data 1	0E	00
5	Data 2	24	00
6	Data 3	17	00
7	Data 4	05	00
8	Data 5	03	00

→ All values must be transferred as hexadecimal values.

Read/write image data

→ Please also observe the relevant description of possible image data provided in the easy800 manual or in the EASY-SOFT Help.

The latest edition of the manual is available as a PDF file from the Internet at: http://www.moeller.net/support: Search Term: AWB2528-1423GB

The information provided in section "General notes on working with image data" on page 50 also applies to easy800.

Overview

Operands	Meaning	Read/ write	Command	Page
IW0	"Read local inputs IW0"	Read	01	149
IW1 – IW8	"Read inputs of the stations IW1 to IW8"	Read	01	151
IA1 – IA4	"Read local analog inputs IA1 to IA4"	Read	02	152
ID1 – ID16	"Read local diagnostics ID1 to ID16"	Read	03	154
QW0, QW1 – QW8	"Read and write local QW0 outputs/ outputs of the stations QW1 to QW8"	Read/write	04	156
QA1	"Reading and writing local analog output QA1"	Read/write	05	158
P1 – P4	"Reading local P buttons"	Read	06	159
R1 – R16 S1 – S8	"Reading RW.. inputs/SW.. outputs from EasyLink"	Read	07/09	161
RN1 – RN32 SN1 – SN32	"Reading receive data network RN1 .. RN32/send data network SN1 .. SN32"	Read	08/0A	163
M...	"Reading and writing markers"	Read/write	0B – 0E	165

Read local inputs IW0

This command string enables you to read the local inputs of the easy800/MFD. The relevant input word is stored in Intel format.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	01	01
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 84
6	Data 2 (High Byte)	00	→ table 84
7 – 8	Data 3 – 4	00	00

Table 84: Byte 5 to 6: Data 1 to 2

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

Example: Read local inputs IW0

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Read	91	–
	Response: Read successful	–	C2
2	Len	02	02
3	Type	01	01
4	Index	00	00
5	Data 1	00	C4
6	Data 2	00	02
7	Data 3	00	00
8	Data 4	00	00

All values must be transferred as hexadecimal values.

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

Read inputs of the stations IW1 to IW8

The easy800 and MFD devices can be remotely expanded very simply using the EASYNET. The service offered here makes it possible to implement read access to the inputs of individual NET stations.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	01	01
4	Index	01 – 08[1]	01 – 08[1]
5	Data 1 (Low Byte)	00	→ table 84 on page 149.
6	Data 2 (High Byte)	00	
7 – 8	Data 3 – 4	00	00

1) Corresponds to address of network station

Read local analog inputs IA1 to IA4

The analog inputs on the easy800 and MFD basic units can be read directly via PROFIBUS-DP. The 16-bit value is transferred in Intel format (LowByte first).

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	02	02
4	Index	01 – 04[1]	01 – 04[1]
5	Data 1 (Low Byte)	00	See example
6	Data 2 (High Byte)	00	See example
7 – 8	Data 3 – 4	00	00

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

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

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Read	91	–
	Response: Read successful	–	C2
2	Len	02	02
3	Type	02	02
4	Index	01[1]	01[1]
5	Data 1	00	D9
6	Data 2	00	02
7	Data 3	00	00
8	Data 4	00	00

1) 01 = Analog input 1

Byte 5 – Data 1 (Low Byte): $D9_{hex}$

Byte 6 – Data 2 (High Byte): 02_{hex}

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

Read local diagnostics ID1 to ID16

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

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	03	03
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 85
6	Data 2 (High Byte)	00	→ table 85
7 – 8	Data 3 – 4	00	00

Table 85: Byte 5 to 6: Data 1 to 2

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

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

Example

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

Read and write local QW0 outputs/outputs of the stations QW1 to QW8

You can read and write the local outputs directly via PROFIBUS-DP. However, the outputs are only switched externally if the device is in Run mode and the addressed output is not being used in the circuit diagram. → section "Read/write image data" on page 148.

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command		
	Read	91	–
	Write	B1	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	02	02
3	Type	04	04
4	Index[1]	00/01 – 08	00/01 – 08
5	Data 1		
	Read operation	00	→ table 85
	For write operation	→ table 86	00
6 – 8	Data 2 – 4	00	00

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

Table 86: Byte5: Data

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

Reading and writing local analog output QA1

The commands provided can be used to access the local analog output of the easy800 or MFD basic unit. When writing to the analog output, however, the value will only be output externally if the device concerned is in Run mode and the image concerned has not been overwritten by actual program. → section "Read/write image data" on page 148.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command		
	Read	91	–
	Write	B1	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	02	02
3	Type	05	05
4	Index	00	00
5 – 6	Data 1 – 2		
	Read operation	00	See example
	For write operation	See example	00
7 – 8	Data 3 – 4	00	00

Example
The analog output is to output a value of approx. 5 V.

500 = $01F4_{hex}$ Byte 5 – Data 1 (LowByte) : $F4_{hex}$
Byte 6 – Data 2 (HighByte): 01_{hex}

Reading local P buttons

The local P buttons are the display cursor buttons of the easy800/MFD basic unit. You can scan the buttons in both Run and Stop mode.

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

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

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	06	06
4	Index	00	00
5	Data 1 (Low Byte)	00	→ table 87
6 – 8	Data 2 – 4	00	00

Table 87: Byte 5: Data

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

Reading RW.. inputs/SW.. outputs from EasyLink

This service allows you to read the local R and S data and the data of the NET stations (1 – 8) transferred via EASYLINK, again from the relevant easy800/MFD image.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	02	02
3	Type	For RW: 07	For RW: 07
		For SW: 09	For SW: 09
4	Index	00/01 – 08[1]	00/01 – 08[1]
5	Data 1 (Low Byte)	00	→ table 88
6	Data 2 (High Byte)	00	→ table 88
7 – 8	Data 3 – 4	00	00

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

Table 88: Byte 5 to 6: Data 1 to 2

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

Reading receive data network RN1 .. RN32/send data network SN1 .. SN32

EASYNET allows a point-to-point connection to be implemented between the individual NET stations. The RN and SN data are used for the data exchange (see the easy800 manual).

→ The RN SN data of the local device (Index = 0) to which the EASY204-DP is fitted cannot be scanned. In this case the command would be denied with the $0C_{hex}$ signal.

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	91	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Len	04	04
3	Type	For RN1 – RN32: 08	
		For SN1 – SN32: 0A	
4	Index	01 – 08[1]	01 – 08[1]
5 – 8	Data 1 – 4	00	→ table 89

1) Corresponds to NET-ID

Table 89: Byte 5 to 8: Data 1 to 4

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

Reading and writing markers

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command		
	Read	91	–
	Write	B1	–
	Response		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Len	→ table 90	→ table 90
3	Type		
4	Index		
5 – 8	Data 1 – 4		
	Read operation	00	→ "Example 1" on page 167
	For write operation	→ "Example 2" on page 168	00

Table 90: Byte 2 – 4: Len, Type, Index

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

If required, refer to the more detailed description of the marker allocation in the easy800 manual. Only a small extract of this manual is shown at this point in order to illustrate the allocation principle.

Attention!
The function blocks and DW markers (32-bit values) of easy800/MFD operate with signed values.

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

The relevant marker values are transferred in Intel format. In other words, the first byte is the low byte (Byte 5) and the last byte the high byte.

Example 1
Read marker bit M62

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Read	91	–
	Response: Read successful	–	C2
2	Len	01	01
3	Type	0B	0B
4	Index	3E	3E
5	Data 1	00	01
6	Data 2	00	00
7	Data 3	00	00
8	Data 4	00	00

Result: Data 1 = 01_{hex} → M62 was set

Example 2
Write marker word MW32 with 823

$823_{dec} = 337_{hex} \rightarrow$ Data 1 = 37_{hex}, Data 2 = 03_{hex}

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Write	B1	–
	Response: Write successful	–	C1
2	Len	02	02
3	Type	0D	0D
4	Index	20	20
5	Data 1	37	00
6	Data 2	03	00
7	Data 3	00	00
8	Data 4	00	00

Read/write function block data

Please also note the relevant description of the function blocks provided in the easy800 manual.

The latest edition of the manual is available as a PDF file from the Internet at: http://www.moeller.net/support: Search Term: AWB2528-1423GB)

General notes

Always note the following when working with function blocks:

- The relevant data is transferred in Intel format. In other words, the first byte is the low byte (Byte 5) and the last byte (byte 8) the high byte.
- The maximum data length is 4 bytes. All values must be transferred in hexadecimal format.
- All 32-bit values are treated as signed values. When transferring 32-bit values, ensure that the appropriate value range is suitable for long integers, i.e. signed. 32-bit value: –2147483648 .. 0 .. +2147483647

Overview

Operands	Meaning	Read/ write	Type	Page
A01 – A32	"Analog value comparators A01 .. A32"	Read/write	11	171
AR01 – AR32	"Arithmetic function blocks AR01 .. AR32"	Read/write	12	173
BC01 – BC32	"Block compare function blocks BC01 .. BC32"	Read/write	25	175
BT01 – BT32	"Block transfer function blocks BT01 .. BT32"	Read/write	26	177
BV01 – BV32	"Boolean sequence function blocks BV01 .. BV32"	Read/write	13	179
C01 – C32	"Counters C01 .. C32"	Read/write	14	181
CF01 – CF04	"Frequency counters CF01 .. CF04"	Read/write	15	183
CH01 – CH04	"High-speed counters CH01 .. CH04"	Read/write	16	185
CI01 – CI02	"Incremental encoder counters CI01 .. CI02"	Read/write	17	187
CP01 – CP32	"Comparators CP01 .. CP32"	Read/write	18	189
D01 – D32	"Text output function blocks D01 ..D32"	Read/write	19	191
DB01 – DB32	"Data function blocks DB01 .. DB32"	Read/write	1A	194
DC01 – DC32	"PID controllers DC01 .. DC32"	Read/write	27	196
FT01 – FT32	"Signal smoothing filters FT01 .. FT32"	Read/write	28	199
GT01 – GT32	"Receive network data function blocks GT01 .. GT32"	Read	1B	201
HW01 – HW32	"7-day time switches HW01 .. HW32"	Read	1C	203
HY01 – HY32	"Year time switches HY01 .. HY32"	Read	1D	206
LS01 – LS32	"Value scaling function blocks LS01 .. LS32"	Read/write	29	209
MR01 – MR32	"Master reset function blocks MR01 .. MR32"	Read	0F	211
NC01 – NC32	"Numerical converters NC01 .. NC32"	Read/write	2A	213
OT01 – OT04	"Operating hours counters OT01 .. OT04"	Read/write	1E	215
PT01 – PT32	"Send network data function blocks PT01 .. PT32"	Read	1F	217
PW01 – PW02	"Pulse width modulation function blocks PW01 .. PW02"	Read/write	2B	219
SC01	"Synchronize clock function block SC01"	Read	20	221
ST01	"Set cycle time function block ST01"	Read/write	2C	222
T01 – T32	"Timing relays T01 .. T32"	Read/write	21	224
VC01 – VC32	"Value limitation function blocks VC01 .. VC32"	Read/write	2D	227

Analog value comparators A01 .. A32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	11	11
3	Instance	01 – 20	01 – 20
4	Index	→ table 91	→ table 91
5 – 8	Data 1 – 4	00	depending on index, → table 92, 93

Table 91: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 92		×	
01	Mode, → table 93		×	
02	Comparison value 1	I1	×	c[1]
03	Gain factor for I1 (I1 = F1 × value)	F1	×	c[1]
04	Comparison value 2	I2	×	c[1]
05	Gain factor for I2 (I2 = F2 × value)	F2	×	c[1]
06	Offset for value I1	OS	×	c[1]
07	Switching hysteresis for value I2 (the value of HY is for both positive and negative hysteresis.)	HY	×	c[1]

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

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

Table 92: Index 0: Bit IO

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

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

Table 93: Index 1 - Mode

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

Arithmetic function blocks AR01 .. AR32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	12	12
3	Instance	01 – 20	01 – 20
4	Index	→ table 94	→ table 94
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 95, 96
	Write operation	depending on index, → table 95, 96	00

Table 94: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 95		×	
01	Mode, → table 96		×	
02	First operand	I1	×	c[1]
03	Second operand	I2	×	c[1]
04	Result	QV	×	

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

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

Table 95: Index 0: Bit IO

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

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

Table 96: Index 1 - Mode

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

Block compare function blocks BC01 .. BC32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	25	25
3	Instance	01 – 20	01 – 20
4	Index	→ table 97	→ table 97
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 98, 99
	Write operation	depending on index, → table 98, 99	00

Table 97: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 98		×	
01	Mode, → table 99		×	
02	Source range 1	I1	×	c[1]
03	Target range 2	I2	×	c[1]
04	Number of elements to compare: 8 (max. 192 bytes)	NO	×	c[1]

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

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

Table 98: Index 0 – Bit IO

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

1) Activates the function block on status 1.

2) Status 1 if the data ranges are equal; status 0 if not equal

Error outputs

3) Status 1 if the number of elements exceeds the source or target range.

4) Status 1 if the source and target range overlap.

5) Status 1 if the source or target range are outside of the available marker range (offset error)

Table 99: Index 1 - Mode

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

Block transfer function blocks BT01 .. BT32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	26	26
3	Instance	01 – 20	01 – 20
4	Index	→ table 100	→ table 100
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 101, 102
	Write operation	depending on index, → table 101, 102	00

Table 100: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 101		×	
01	Mode, → table 102		×	
02	Source range 1	I1	×	c[1]
03	Target range 2	I2	×	c[1]
04	Number of elements to compare: max. 192 bytes	NO	×	c[1]

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

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

Table 101: Index 0 – Bit IO

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

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

Error outputs

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

Table 102: Index 1 - Mode

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

Boolean sequence function blocks BV01 .. BV32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	13	13
3	Instance	01 – 20	01 – 20
4	Index	→ table 103	→ table 103
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 104, 105
	Write operation	depending on index, → table 104, 105	00

Table 103: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 104		×	
01	Mode, → table 105		×	
02	First operand	I1	×	c[1]
03	Second operand	I2	×	c[1]
04	Operation result	QV	×	

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

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

Table 104: Index 0: Bit IO

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

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

Table 105: Index 1 - Mode

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

Counters C01 .. C32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	14	14
3	Instance	01 – 20	01 – 20
4	Index	→ table 106	→ table 106
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 107
	Write operation	depending on index, → table 107	00

Table 106: Operand overview

Index (hex)	Operand		Value	Read	Write
00	Bit IO		→ table 107	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

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

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

Table 107: Index 0: Bit IO

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

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

Frequency counters CF01 .. CF04

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	15	15
3	Instance	01 – 04	01 – 04
4	Index	→ table 108	→ table 108
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 109
	Write operation	depending on index, → table 109	00

Table 108: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 109		×	
01	Mode/Parameter		–	–
02	Upper setpoint	SH	×	c[1]
03	Lower setpoint	SL	×	c[1]
04	Actual value in Run mode	QV	×	

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

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

Table 109: Index 0: Bit IO

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

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

High-speed counters CH01 .. CH04

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	16	16
3	Instance	01 – 04	01 – 04
4	Index	→ table 110	→ table 110
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 111
	Write operation	depending on index, → table 111	00

Table 110: Operand overview

Index (hex)	Operand		Value	Read	Write
00	Bit IO		→ table 111	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

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

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

Table 111: Index 0: Bit IO

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

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

Incremental encoder counters CI01 .. CI02

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	17	17
3	Instance	01 – 02	01 – 02
4	Index	→ table 112	→ table 112
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 113
	Write operation	depending on index, → table 113	00

Table 112: Operand overview

Index (hex)	Operand		Value	Read	Write
00	Bit IO		→ table 113	×	
01	Mode/Parameter		–	–	–
02	Upper setpoint	SH	In integer range from –2147483648 to +2147483647	×	c[1]
03	Lower setpoint	SL		×	c[1]
04	Preset actual value	SV		×	c[1]
05	Actual value in Run mode	QV		×	

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

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

Table 113: Index 0 – Bit IO

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

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

Comparators CP01 .. CP32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	18	18
3	Instance	01 – 20	01 – 20
4	Index	→ table 114	→ table 114
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 115
	Write operation	depending on index, → table 115	00

Table 114: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 115		×	
01	Mode/Parameter		–	–
02	Comparison value	I1	×	c[1]
03	Comparison value	I2	×	c[1]

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

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

Table 115: Index 0 – Bit IO

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

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

Text output function blocks D01 ..D32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	19	19
3	Instance	01 – 20	01 – 20
4	Index	→ table 116	→ table 116
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 117
	Write operation	depending on index, → table 117	00

Table 116: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 117	×	
01	Mode/Parameter	–	–
02	Text line 1, column 1 - 4	×	
03	Text line 1, column 5 - 8	×	
04	Text line 1, column 9 - 12	×	
05	Text line 1, column 13 - 16	×	
06	Text line 2, column 1 - 4	×	
07	Text line 2, column 5 - 8	×	
08	Text line 2, column 9 - 12	×	
09	Text line 2, column 13 - 16	×	
10	Text line 3, column 1 - 4	×	
11	Text line 3, column 5 - 8	×	
12	Text line 3, column 9 - 12	×	
13	Text line 3, column 13 - 16	×	
14	Text line 4, column 1 - 4	×	
15	Text line 4, column 5 - 8	×	
16	Text line 4, column 9 - 12	×	
17	Text line 4, column 13 - 16	×	
18	Variable 1	×	c[1]
19	Variable 2	×	c[1]
20	Variable 3	×	c[1]
21	Variable 4	×	c[1]
22	Scaling minimum value 1	×	
23	Scaling minimum value 2	×	
24	Scaling minimum value 3	×	
25	Scaling minimum value 4	×	
26	Scaling maximum value 1	×	

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

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

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

Table 117: Index 0 – Bit IO

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

1) Text function block enable
2) Status 1, text function block is active

Data function blocks DB01 .. DB32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	1A	1A
3	Instance	01 – 20	01 – 20
4	Index	→ table 118	→ table 118
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 119
	Write operation	depending on index, → table 119	00

Table 118: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 119		×	
01	Mode/Parameter		–	–
02	Input value: value that is transferred to the QV output when the FB is triggered.	I1	×	c[1]
03	Output value	QV	×	

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

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

Table 119: Index 0 – Bit IO

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

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

PID controllers DC01 .. DC32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	27	27
3	Instance	01 – 20	01 – 20
4	Index	→ table 120	→ table 120
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 121, 122
	Write operation	depending on index, → table 121, 122	

Table 120: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 121		×	
01	Mode, → table 122		×	
02	Setpoint: –32768 to +32767	I1	×	c[1]
03	Actual value: –32768 to +32767	I2	×	c[1]
04	Proportional gain [%], Value range: 0 to 65535	KP	×	c[1]
05	Reset time [0.1 s], Value range: 0 to 65535	TN	×	c[1]
06	Rate time [0.1 s], Value range: 0 to 65535	TV	×	c[1]
07	Scan time = Time between function block calls. Value range: 0.1s to 6553.5s. If 0 is entered as the value, the scan time will be determined by the program cycle time.	TC	×	c[1]
08	Manual manipulated variable, value range: –4096 to +4095	MV	×	c[1]
09	Manipulated variable • Mode: UNI, value range: 0 to +4095 (12 bit) • Mode: BIP, value range: –4096 to +4095 (13 bit)	QV	×	

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

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

Table 121: Index 0 – Bit IO

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

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

Table 122: Index 1 - Mode

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

Signal smoothing filters FT01 .. FT32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	28	28
3	Instance	01 – 20	01 – 20
4	Index	→ table 123	→ table 123
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 124
	Write operation	depending on index, → table 124	00

Table 123: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 124		×	
01	Mode/Parameter		–	–
02	Input value, value range: –32768 to +32767	I1	×	c[1]
03	Recovery time [0.1 s], Value range: 0 to 65535	TG	×	c[1]
04	Proportional gain [%], Value range: 0 to 65535	KP	×	c[1]
05	Delayed output value, value range: –32768 to +32767	QV	×	

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

Table 124: Index 0 – Bit IO

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

1) Activates the function block on status 1.

Receive network data function blocks GT01 .. GT32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	1B	1B
3	Instance	01 – 20	01 – 20
4	Index	→ table 125	
5 – 8	Data 1 – 4	00	depending on index, → table 126, 127

Table 125: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 126	×	
01	Mode/Parameters, → table 127	×	–
02	Output value: actual value from the network QV	×	

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

Table 126: Index 0 – Bit IO

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

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

Table 127: Index 1 – Mode/Parameters (designation of PUT FB with data to be received)

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

1) Number of station sending the value. Possible station numbers: 01 to 08

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

7-day time switches HW01 .. HW32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	1C	1C
3	Instance	01 – 20	01 – 20
4	Index	→ table 128	
5 – 8	Data 1 – 4	00	depending on index, → table 129

Table 128: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO → table 129	×	
01	Mode/Parameter	–	–
02	Parameters → table 130	×	
	Channel A		
03	Channel B		
04	Channel C		
05	Channel D		

Table 129: Index 0 – Bit IO

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

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

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

Table 130: Index 2 – 5, Parameter channels A – D

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

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

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

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

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	81	80
1	Command: Read	92	–
	Response: Read successful	–	C2
2	Type	1C	1C
3	Instance	13	13
4	Index	02	02
5	Data 1	00	62
6	Data 2	00	0B
7	Data 3	00	7B
8	Data 4	00	25

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

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

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

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

Year time switches HY01 .. HY32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	1D	1D
3	Instance	01 – 20	01 – 20
4	Index	→ table 131	
5 – 8	Data 1 – 4	00	depending on index, → table 132

Table 131: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO → table 132	×	
01	Mode/Parameter	–	–
02	Parameters → table 133	×	
	Channel A		
03	Channel B		
04	Channel C		
05	Channel D		

Table 132: Index 0 – Bit IO

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

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

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

Table 133: Index 2 – 5, Parameter channels A – D

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

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

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

Example

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

Index 2 – 5, Parameter channels A – D

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

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

Index 2 – 5, Parameter channels A – D

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

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

Resulting telegram:

Byte	**Meaning**	**Value (hex), sent by**	
		Master	**Slave**
0	Toggle byte	81	80
1	Command: Write	B2	–
	Response: Write successful	–	C1
2	Type	1D	1D
3	Instance	0E	0E
4	Index	02	02
5	Data 1	8E	00
6	Data 2	06	00
7	Data 3	43	00
8	Data 4	19	00

→ Further information is available in the S40 Application Note AN27K21g.exe "EASY800/MFD-DP Data Handling Function Block for PS416 and PS4-341".

Value scaling function blocks LS01 .. LS32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	29	29
3	Instance	01 – 20	01 – 20
4	Index	→ table 134	→ table 134
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 135
	Write operation	depending on index, → table 135	

Table 134: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 135		×	
01	Mode/Parameter		–	–
02	Input value, value range: 32 bit	I1	×	c[1]
03	Interpolation point 1, X coordinate, value range: 32 bit	X1	×	c[1]
04	Interpolation point 1, Y coordinate, value range: 32 bit	Y1	×	c[1]
05	Interpolation point 2, X coordinate, value range: 32 bit	X2	×	c[1]
06	Interpolation point 2, Y coordinate, value range: 32 bit	Y2	×	c[1]
07	Output value: contains the scaled input value	QV	×	

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

Table 135: Index 0 – Bit IO

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

1) Activates the function block on status 1.

Master reset function blocks MR01 .. MR32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	0F	0F
3	Instance	01 – 20	01 – 20
4	Index		
	Bit IO	00	00
	Mode	01	01
5 – 8	Data 1 – 4	00	depending on index, → table 136, 137

Table 136: Index 0 – Bit IO

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

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

Table 137: Index 1 - Mode

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

Numerical converters NC01 .. NC32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	2A	2A
3	Instance	01 – 20	01 – 20
4	Index	→ table 138	→ table 138
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 139, 140
	Write operation	depending on index, → table 139, 140	00

Table 138: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 139	×	
01	Mode, → table 140	×	
02	Input value: I1 operand to be converted	×	c[1]
03	Output value: QV contains the conversion result	×	

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

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

Table 139: Index 0 – Bit IO

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

1) Activates the function block on status 1.

Table 140: Index 1 - Mode

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

Operating hours counters OT01 .. OT04

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	1E	1E
3	Instance	01 – 04	01 – 04
4	Index	→ table 141	→ table 141
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 142
	Write operation	depending on index, → table 142	00

Table 141: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 142	×	
01	Mode/Parameter	–	–
02	Upper threshold value I1	×	c[1]
03	Actual value of operating hours counter QV	×	

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

Table 142: Index 0 – Bit IO

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

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

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

Send network data function blocks PT01 .. PT32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	1F	1F
3	Instance	01 – 20	01 – 20
4	Index	→ table 143	
5 – 8	Data 1 – 4	00	depending on index, → table 144

Table 143: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 144	×	
01	Mode/Parameter	–	–
02	Input value: Setpoint that it transmitted to the NET network I1	×	

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

Table 144: Index 0 – Bit IO

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

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

Pulse width modulation function blocks PW01 .. PW02

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	2B	2B
3	Instance	01 – 02	01 – 02
4	Index	→ table 145	→ table 145
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 146
	Write operation	depending on index, → table 146	00

Table 145: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 146		×	
01	Mode/Parameter		–	–
02	Manipulated variable, value range: 0 to 4095 (12 bit)	SV	×	c[1]
03	Period duration [ms], Value range: 0 to 65535	PD	×	c[1]
04	Minimum on duration [ms], Value range: 0 to 65535	ME	×	c[1]

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

Table 146: Index 0 – Bit IO

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

1) Activates the function block on status 1.

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

Synchronize clock function block SC01

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	
1	Command: Read	92	–
	Response:		
	Read successful	–	C2
	Command rejected	–	C0
2	Type	20	20
3	Instance	01	01
4	Index	→ table 147	
5 – 8	Data 1 – 4	00	depending on index, → table 148

Table 147: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 148	×	
01	Mode/Parameter	–	–

Table 148: Index 0 – Bit IO

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

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

2) Status 1 if the trigger coil SC01T_ is also 1.

Set cycle time function block ST01

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	2C	2C
3	Instance	01	01
4	Index	→ table 149	→ table 149
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 150
	Write operation	depending on index, → table 150	00

Table 149: Operand overview

Index (hex)	Operand	Read	Write
00	Bit IO, → table 150	×	
01	Mode/Parameter	–	–
02	Cycle time in ms, I1 value range: 0 – 1000	×	c[1]

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

Table 150: Index 0 – Bit IO

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

1) Activates the function block on status 1.

Timing relays T01 .. T32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	21	21
3	Instance	01 – 20	01 – 20
4	Index	→ table 151	→ table 151
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 152, 153
	Write operation	depending on index, → table 152, 153	

Table 151: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 152		×	
01	Mode/Parameters, → table 153		×	
02	Setpoint 1: Time setpoint 1	I1	×	c[1]
03	Setpoint 2: Time setpoint 2 (with timing relay with 2 setpoints)	I2	×	c[1]
04	Actual value: Time elapsed in Run mode	QV	×	

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

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

Table 152: Index 0 – Bit IO

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

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

Table 153: Index 1 - Mode/Parameters

Mode	Data 1	Operating mode
	0	On-delayed
	1	On-delayed with random setpoint
	2	Off-delayed
	3	Off-delayed with random setpoint
	4	On and off delayed (two time setpoints)
	5	On and off delayed each with random setpoint (two time setpoints)
	6	Impulse transmitter
	7	Flashing relay (two time setpoints)
	8	Off-delayed, retriggerable (easy600 Mode)
	9	Off-delayed with random setpoint, retriggerable (easy600 Mode)
Para-meters	**Data 3**	**Operating mode**
	0	S (milliseconds)
	1	M:S (seconds)
	2	H:M (minutes)

Value limitation function blocks VC01 .. VC32

Telegram structure

Byte	Meaning	Value (hex), sent by	
		Master	Slave
0	Toggle byte	→ page 142	→ page 142
1	Command:		
	Read	92	–
	Write	B2	–
	Response:		
	Read successful	–	C2
	Write successful	–	C1
	Command rejected	–	C0
2	Type	2D	2D
3	Instance	01 – 20	01 – 20
4	Index	→ table 154	→ table 154
5 – 8	Data 1 – 4		
	Read operation	00	depending on index, → table 155
	Write operation	depending on index, → table 155	

Table 154: Operand overview

Index (hex)	Operand		Read	Write
00	Bit IO, → table 155		×	
01	Mode/Parameter		–	–
02	Input value	I1	×	c[1]
03	Upper limit value	SH	×	c[1]
04	Lower limit value	SL	×	c[1]
05	Output value: outputs the value present at input I1 within the set limits.	QV	×	

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

Table 155: Index 0 – Bit IO

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

1) Activates the function block on status 1.

Annex

What Happens If ...?

Event	Explanation	Remedy
POW LED not lit	No power supply	Connect and switch on power supply
POW LED flashing	Data transfer via EASY-LINK OK	
BUS LED not lit	No PROFIBUS-DP data communication	Connect and start PROFIBUS-DP
BUS LED lit	Data transfer via PROFIBUS-DP OK	
Slave not signalling	– No slave address set – No bus terminating resistor present – Cable, plug faulty – No power supply	– Set slave address – Set bus terminating resistors – Check connection – Provide power supply to device
Write command rejected	– Command not permissible – EASY display not on the Status display	– Change command – Show Status display
Actual value is zero	No actual value present	Function relay does not have an actual value or not triggered

Overview of commands

easy600

The commands are sorted in ascending order:

Command value hex	
01	Write T1 timing relay setpoint
02	Write T2 timing relay setpoint
03	Write T3 timing relay setpoint
04	Write T4 timing relay setpoint
05	Write T5 timing relay setpoint
06	Write T6 timing relay setpoint
07	Write T7 timing relay setpoint
08	Write T8 timing relay setpoint
09	Write C1 counter relay setpoint
0A	Write C2 counter relay setpoint
0B	Write C3 counter relay setpoint
0C	Write C4 counter relay setpoint
0D	Write C5 counter relay setpoint
0E	Write C6 counter relay setpoint
0F	Write C7 counter relay setpoint
10	Write C8 counter relay setpoint
12	Write time switch 1 channel A
13	Write time switch 1 channel B
14	Write time switch 1 channel C
15	Write time switch 1 channel D
16	Write time switch 2 channel A
17	Write time switch 2 channel B
18	Write time switch 2 channel C
19	Write time switch 2 channel D
1A	Write time switch 3 channel A

Command value hex	
1B	Write time switch 3 channel B
1C	Write time switch 3 channel C
1D	Write time switch 3 channel D
1E	Write time switch 4 channel A
1F	Write time switch 4 channel B
20	Write time switch 4 channel C
21	Write time switch 4 channel D
22	Write analog value comparator A1
23	Write analog value comparator A2
24	Write analog value comparator A3
25	Write analog value comparator A4
26	Write analog value comparator A5
27	Write analog value comparator A6
28	Write analog value comparator A7
29	Write analog value comparator A8
2A	Write time
2B	Read time switch 1 channel A
2C	Read time switch 1 channel B
2D	Read time switch 1 channel C
2E	Read time switch 1 channel D
2F	Read time switch 2 channel A
30	Read time switch 2 channel B
31	Read time switch 2 channel C
32	Read time switch 2 channel D
33	Read time switch 3 channel A
34	Read time switch 3 channel B
35	Read time switch 3 channel C
36	Read time switch 3 channel D

Command value hex	
37	Read time switch 4 channel A
38	Read time switch 4 channel B
39	Read time switch 4 channel C
3A	Read time switch 4 channel D
3C	Read time
3D	Read status of analog and digital inputs
3E	Read status of P buttons and operator buttons
3F	Read status of timing relays, counter relays, time switches and analog value comparators
40	Read status of markers, digital outputs and text display markers
41	Read T1 actual value
42	Read T2 actual value
43	Read T3 actual value
44	Read T4 actual value
45	Read T5 actual value
46	Read T6 actual value
47	Read T7 actual value
48	Read T8 actual value
49	Read C1 counter relay actual value
4A	Read C2 counter relay actual value
4B	Read C3 counter relay actual value
4C	Read C4 counter relay actual value
4D	Read C5 counter relay actual value
4E	Read C6 counter relay actual value
4F	Read C7 counter relay actual value
50	Read C8 counter relay actual value

easy800/MFD

Date and time	Byte 1 Command (hex)	Byte 2 Len[1] (hex)	Byte 3 Index (hex)
Read/write date and time	93/B3	05	00
Winter/summer time, DST			01

Image data	Byte 1 Command (hex)	Byte 2 Len[1] (hex)	Byte 3 Type (hex)	Byte 4 Index (dec)
Read/write image data	91/B1			
"Local inputs: I1 – I16"		2	01	0
"Read inputs of the stations IW1 to IW8"				1 – 8
"Read local analog inputs IA1 to IA4"			02	1 – 4
"Read local diagnostics ID1 to ID16"			03	0
"Read and write local QW0 outputs/outputs of the stations QW1 to QW8"			04	0/1 – 8
"Reading and writing local analog output QA1"			05	0
"Reading local P buttons"		1	06	0
"Reading RW.. inputs/SW.. outputs from EasyLink"		2	07/09	0
"Reading receive data network RN1 .. RN32/send data network SN1 .. SN32"				1 – 8
"Reading receive data network RN1 .. RN32/send data network SN1 .. SN32"		4	08/0A	1 – 8
Marker bit M1 .. M96		1	0B	1 – 96
Marker byte MB1 .. MB96			0C	1 – 96
Marker word MW1 .. MW96		2	0D	1 – 96
Marker double word MD1 .. MD96		4	0E	1 – 96

1) Len... stands for the number of data bytes to be sent.

Function blocks	Byte 1 Command (hex)	Byte 2 Type (hex)	Byte 3 Instance (hex)
Read/write function blocks	92/B2		
"Receive network data function blocks GT01 .. GT32"		0F	1 – 20
"Analog value comparators A01 .. A32"		11	1 – 20
"Arithmetic function blocks AR01 .. AR32"		12	1 – 20
"Boolean sequence function blocks BV01 .. BV32"		13	1 – 20
"Counters C01 .. C32"		14	1 – 20
"Frequency counters CF01 .. CF04"		15	1 – 20
"High-speed counters CH01 .. CH04"		16	1 – 4
"Incremental encoder counters CI01 .. CI02"		17	1 – 2
"Comparators CP01 .. CP32"		18	1 – 20
"Text output function blocks D01 ..D32"		19	1 – 20
"Data function blocks DB01 .. DB32"		1A	1 – 20
"Receive network data function blocks GT01 .. GT32"		1B	1 – 20
"7-day time switches HW01 .. HW32"		1C	1 – 20
"Year time switches HY01 .. HY32"		1D	1 – 20
"Operating hours counters OT01 .. OT04"		1E	1 – 4
"Send network data function blocks PT01 .. PT32"		1F	1 – 20
"Synchronize clock function block SC01"		20	1
"Set cycle time function block ST01"		21	1 – 20
"Block compare function blocks BC01 .. BC32"		25	1 – 20
"Block transfer function blocks BT01 .. BT32"		26	1 – 20
"PID controllers DC01 .. DC32"		27	1 – 20
"Signal smoothing filters FT01 .. FT32"		28	1 – 20
"Value scaling function blocks LS01 .. LS32"		29	1 – 20
"Numerical converters NC01 .. NC32"		2A	1 – 20
"Pulse width modulation function blocks PW01 .. PW02"		2B	1 – 2
"Set cycle time function block ST01"		2C	1
"Value limitation function blocks VC01 .. VC32"		2D	1 – 20

Technical Data

General

Standards and regulations	EN 55011, EN 55022, IEC/EN 61-4, IEC 60068-2-27, IEC 61158
Dimensions (W × H × D)	35.5 × 90 × 56.5
Weight	150 g
Mounting	Top-hat rail to DIN 50022, 35 mm Screw fixing with fixing brackets ZB4-101-GF1 (accessories)

Ambient temperatures

Ambient temperature Installed horizontally/vertically	Cold to IEC 60068-2-1 Heat to IEC 60068-2-2	–25 to 55 °C
Condensation		Prevent condensation with suitable measures
Storage/transport temperature		–40 to +70 °C
Relative air humidity	IEC 60068-2-30	5 to 95 %, non-condensing
Air pressure (operation)		795 to 1080 hPa
Corrosion resistance	IEC 60068-2-42 IEC 60068-2-43	SO_2 10 cm^3/m^3, 4 days H_2S 1 cm^3/m^3, 4 days

Ambient mechanical conditions

Pollution degree		2
Degree of protection	EN 50178 IEC 60529 VBG4	IP20
Oscillations	IEC 60068-2-6	10 to 57 Hz (constant amplitude 0.15 mm) 57 to 150 Hz (constant acceleration 2 g)
Shocks	IEC 60068-2-27	18 shocks (semi-sinusoidal 15 g/11 ms)
Drop	IEC 60068-2-31	Drop height 50 mm
Free fall, packaged	IEC 60068-2-32	1 m

Electromagnetic compatibility (EMC)

Electrostatic discharge	IEC/EN 61000-4-2, degree of severity 3	8 kV air discharge, 6 kV contact discharge
Electromagnetic fields	IEC/EN 61000-4-3	Field strength 10 V/m
Radio interference suppression	EN 55011, EN 55022	Limit class A
Burst	IEC/EN 61000-4-4, degree of severity 3	2 kV supply lines, 1 kV signal lines
High-energy pulses (surge)		
EASY...-DC...	IEC/EN 61000-4-5, degree of severity 2	0.5 kV power cable symmetrical
Line-conducted interference	IEC/EN 61000-4-6	10 V

Dielectric strength

Measurement of the clearance and creepage distance	EN 50178, UL 508, CSC C22.2 No 142
Dielectric strength	EN 50178

Tools and cable cross-sections

Solid	
min.	0.2 mm^2 , AWG 22
max.	4 mm^2, AWG 12
Flexible with ferrule	
min.	0.2 mm^2 , AWG 22
max.	2.5 mm^2, AWG 12
Slot-head screwdriver, width	3.5 × 0.8 mm
Tightening torque max.	0.5 Nm

Power supply

Rated voltage	
Rated value	24 V DC, –15 %, +20 %
Permissible range	20.4 to 28.8 V DC
Residual ripple	< 5 %
Input current at 24 V DC	Normally 200 mA
Voltage dips (IEC/EN 61131-2)	10 ms
Power dissipation at 24 V DC	Normally 4.8 W

LEDs

Power LED (POW)	green
PROFIBUS-DP LED (BUS)	green

PROFIBUS-DP

Device connection	SUB-D 9-pole, socket
Electrical isolation	Bus to power supply (simple) Bus and power supply to "easy" basic unit (safe isolation)
Function	PROFIBUS-DP slave
Interface	RS 485
Bus protocol	PROFIBUS-DP
Baud rates	Automatic search up to 12 MBd
Bus terminating resistors	Connectable via plug
Bus addresses	1 to 126 addressable via "easy" basic unit with display or EASY-SOFT
Services	
Inputs module	All data S1 to S8 (EASY6..)
Outputs module	All data R1 to R16 (EASY6..)
Control commands module	Read/Write Time, day, summer/winter time (DST) All parameters of the EASY function relays

Dimensions

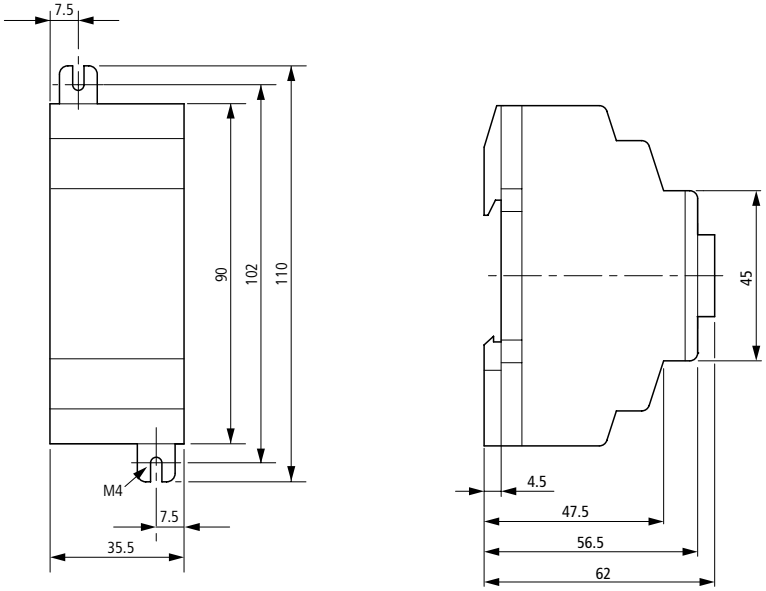


Figure 9: Dimensions EASY204-DP

GSD file

```
Moeller GmbH
; Device: EASY204-DP
; Version: V1.2
; Date: 21.02.2003
; Author: Dieter Bauerfeind/ Ronny Happ
; Description: GSD file for EASY-DP slave module
Modifications:
; 21.02.2003: EASY 800 Support
;
; Copyright (c) 2003 by Moeller GmbH
;=======================================================
#PROFIBUS_DP

GSD_Revision            = 2
Vendor_Name             = "Moeller GmbH"
Model_Name              = "EASY204-DP"
Revision                = "1.2"
Ident_Number            = 0x4D10
Protocol_Ident          = 0 ; PROFIBUS-DP
Station_Type            = 0 ; DP-Slave
FMS_supp                = 0
Hardware_Release        = "V1.0"
Software_Release        = "V1.2"
9.6_supp                = 1
19.2_supp               = 1
93.75_supp              = 1
187.5_supp              = 1
500_supp                = 1
1.5M_supp               = 1
3M_supp                 = 1
6M_supp                 = 1
12M_supp                = 1
```

```
MaxTsdr_9.6             = 60
MaxTsdr_19.2            = 60
MaxTsdr_93.75           = 60
MaxTsdr_187.5           = 60
MaxTsdr_500             = 100
MaxTsdr_1.5M            = 150
MaxTsdr_3M              = 250
MaxTsdr_6M              = 450
MaxTsdr_12M             = 800

Redundancy              = 0
Repeater_Ctrl_Sig       = 2
24V_Pins                = 0

; the following functions are tested for the certification

Freeze_Mode_supp        = 1
Sync_Mode_supp          = 1
Auto_Baud_supp          = 1
Set_Slave_Add_supp      = 1
Fail_safe               = 0
; Slave belongs to switchgear family
Slave_Family            = 2
; Slave implemented with PROFIBUS-DP-ASIC VPC3.
Implementation_Type     = "VPC3"
;
Min_Slave_Interval      = 2
Modular_Station         = 1
Max_Module              = 3
Modul_Offset            = 0
Max_Input_Len           = 12
Max_Output_Len          = 12
Max_Data_Len            = 24
User_Prm_Data_Len       = 0
Max_Diag_Data_Len       = 9
Unit_Diag_Bit(0000)     = "EASY-LINK disconnected"
```

```
Module                    = "1: Control commands 7 Byte" 0xB6
; General ID format
; 7 bytes I/O control commands; consistency via entire length
EndModule

Module                    = "2: Control commands 9 Byte" 0xB8
; General ID format
; 9 bytes I/O control commands; consistency over entire length
EndModule

Module                    = "3: Inputs 3 bytes" 0x92
; General ID format
; 3 byte inputs; consistency over the entire length
EndModule

Module                    = "4: Outputs 3 bytes" 0xA2
; General ID format
; 3 byte outputs; consistency over the entire length
EndModule

Module                    = "5: Inputs 1 byte" 0x90
; General ID format
; 1 byte inputs; consistency over the entire length
EndModule

Module                    = "6: Outputs 1 byte" 0xA0
; General ID format
; 1 byte outputs; consistency over the entire length
EndModule
```

Glossary

This glossary refers to subjects relating to PROFIBUS-DP.

Acknowledge	Acknowledgement returned by the receiving station after having received a signal.
Active metal component	Conductor or conductive component that is live when in operation.
Address	Number, for example, for identifying a memory location, a system or a module within a network.
Addressing	Assignment or setting of an address such as for a module in a network.
Analog	Value, such as voltage, that is infinitely variable and proportional. Analog signals can acquire any value within specific limits.
Automation device	Control device with inputs and outputs that is connected to a technical process. Programmable controllers (PLCs) are a special group of automation devices.
Baud	Unit for the data transfer rate. One baud corresponds to the transmission of one bit per second (bit/s).
Baud rate	Unit of measure of the data transmission speed in bit/s.
Bidirectional	Operation in both directions.
Bus	Bus cable system for data exchange between CPU, memory and I/O level. A bus can consist of several parallel segments, such as the data bus, address bus, control bus and power supply bus.
Bus cycle time	Time interval in which a master will serve all slaves or stations in a bus system, i.e. writes their outputs and reads their inputs.
Bus line	Smallest unit connected to the bus. Consists of the PLC, a module and a bus interface for the module.
Bus system	The entirety of all units which communicate across a bus.

Bus terminating resistor	Resistor at the beginning and end of a bus line for preventing disturbance caused by signal reflections and for adapting bus cables. Bus terminating resistors must always be the last unit at the end of a bus segment.
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	Entirety of all interconnected inactive equipment parts that do not have any contact voltage, even in the event of a fault.
Coding element	Two-part element for the unambiguous allocation of electronic and basic module.
Command-capable modules	Command-capable modules are modules with an internal memory that are capable of executing particular commands (such as output substitute values).
Common potential	Electrical interconnection of the reference potentials of the control and load circuit of I/O modules.
Configuring	Systematic arrangement of the I/O modules of a station.
CPU	Abbreviation for "Central Processing Unit". Central unit for data processing, which represents the core element of a computer.
Digital	A value, for example voltage, that can only be represented by a certain number of states within a defined range, usually defined as 0 and 1.
DIN	Abbreviation for "Deutsches Institut für Normungen e.V." (German standards institute).
Earthing strip	Flexible conductor, mostly braided. Interconnects inactive parts of equipment, e.g. the doors of a control panel and the switch cabinet body.
Electrical equipment	All objects that are used for the generation, conversion, transfer, distribution and use of electric power, such as conductors, cables, machines, control devices.

EMC	Abbreviation for "Electromagnetic Compatibility". The ability of electrical equipment to function trouble-free within a particular environment without a negative effect on the environment concerned.
EN	Abbreviation for "European Norm" or European standard.
Equipotential bonding	Adaptation of the electrical level of the body of electrical equipment and auxiliary conductive bodies by means of an electrical connection.
ESD	Abbreviation for "Electrostatic Discharge".
Field supply	Voltage supply to field devices as well as signal voltage.
Fieldbus	Data network on the sensor/actuator level. The fieldbus interconnects the devices at field level. Characteristic feature of the fieldbus is their highly reliable transfer of signals and real-time response.
Galvanic coupling	A galvanic coupling occurs when two circuits use the same cable. Typical sources of interference are, for example, starting motors, static discharges, clocked devices, and a potential difference between the housing of components and the common power supply.
GND	Abbreviation for "GROUND" (0 potential).
Ground	In electrical engineering the name for conductive grounding with an electrical potential at any point equal to zero. In the environment of grounding devices, the electrical ground potential may not equal zero. This is called a "reference ground".
ground (verb)	Represents the connection of an electrically conductive component to the equipotential earth via a grounding device.
Ground connection	One or several components that have a direct and good contact with the ground.
GSD	The device master data files (GSD) contain standardized PROFIBUS station descriptions. They are used to simplify the configuration of the DP master and DP slaves.
hexadecimal	Number system with base 16. Counting from 0 to 9 and then with the letters A, B, C, D, E and F.

I/O	Abbreviation for "Input/Output".
Impedance	Apparent resistance that a component or circuit of several components has for an alternating current at a particular frequency.
Inactive metal parts	Conductive parts that cannot be touched and which are insulated from active metal parts. They can, however, carry voltage in the event of a fault.
Inductive coupling	Inductive (magnetic) coupling occurs between two current carrying conductors. The magnetism produced by the currents induces an interference voltage. Typical interference sources are, for example transformers, motors, mains cables installed parallel and RF signal cables.
Lightning protection	Represents all measures for preventing system damage due to overvoltage caused by lightning strike.
Low impedance connection	Connection with low alternating-current resistance.
LSB	Abbreviation for "Least Significant Bit". Bit with the least significant value.
Master	Station or node in a bus system that controls communication between the other stations of the bus system.
Master-slave mode	Operating mode in which a station or node of the system acts as master that controls communication on the bus.
Mode	Operating mode.
Module bus	Represents the internal bus of an XI/ON station. Used by the XI/ON modules for communication with the gateway. Independent of the fieldbus.
MSB	Abbreviation for "Most Significant Bit". Bit with the most significant value.
Multimaster Mode	Operating mode in which all stations or nodes of a system have equal rights for communicating on the bus.

Namur	Abbreviation for "Normen-Arbeitsgemeinschaft für Mess- und Regeltechnik" (Standards Committee for Measurement and Control Technology). 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.
Overhead	System management time required in the system in each transmission cycle.
Parameter assignment	Assignment of parameters in the configuration software of the DP master for the individual stations on the bus and their modules.
PLC	Abbreviation for Programmable Logic Controller.
Potential-free	Galvanic isolation between the reference potentials of the control and load circuit of I/O modules.
PROFIBUS-DP	PROFIBUS bus system with the DP protocol. DP stands for "decentralized periphery". PROFIBUS-DP is based on DIN 19245 Part 1+4, and was integrated in the European fieldbus standard EN 50170. It is used for high-speed data exchange between the central DP master and the decentralized peripheral devices, the DP slaves. The comprehensive use is implemented by means of a multi-master concept.
PROFIBUS-DP address	Each PROFIBUS-DP station is assigned an unambiguous PROFIBUS-DP address by means of which it can be addressed by the master.
PROFIBUS-DP master	The PROFIBUS-DP master is the central station and controls the PROFIBUS access of all PROFIBUS-DP slaves.
PROFIBUS-DP slave	PROFIBUS-DP slaves are addressed by the PROFIBUS-DP master and exchange data with it at its request.
Protected against short-circuit	Property of electrical equipment. Short-circuit-proof equipment has the ability to withstand the thermal and dynamic loads that may occur at the location of installation on account of a short-circuit.

Protective conductor	A conductor required for the protection against dangerous currents, designated by the letters PE (abbreviation of "Protective Earth").
Radiated coupling	Radiated coupling occurs when an electromagnetic wave makes contact with a conductor structure. The impact of the wave induces currents and voltages. Typical interference sources are, for example ignition circuits (spark plugs, commutators of electrical motors) and transmitters (e.g. radio-operated devices), which are operated near the corresponding conductor structure.
Reference ground	Ground potential in the area of grounding devices. Unlike "ground", which always has zero potential, it may have any potential except zero.
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 the time interval between the sending of a read job and the receipt of the response. Within an input module, it represents the time interval between the signal change at an input and its output to the bus system.
RS 485	Serial interface in accordance with the EIA standard for high-speed data transmission via several transmitters.
Serial	Describes an information transfer technique. Data is transferred in a bit-stream across the cables.
Shield	Term that describes the conductive covering of cables, cubicles and cabinets.
Shielding	All measures and equipment used for connecting system parts with the shield.
Slave	Station in a bus system that is subordinate to the master.
Station	Function unit or module, consisting of several elements.
SUB-D plug	9-pole plug for connecting the fieldbus.
Topology	Geometric structure of a network or circuit arrangement.

UART	Abbreviation for "Universal Asynchronous Receiver/ Transmitter". A "UART" is a logic circuit used for converting an asynchronous serial data sequence into a bit-parallel data sequence or vice versa.
Unidirectional	Working in one direction.

Index