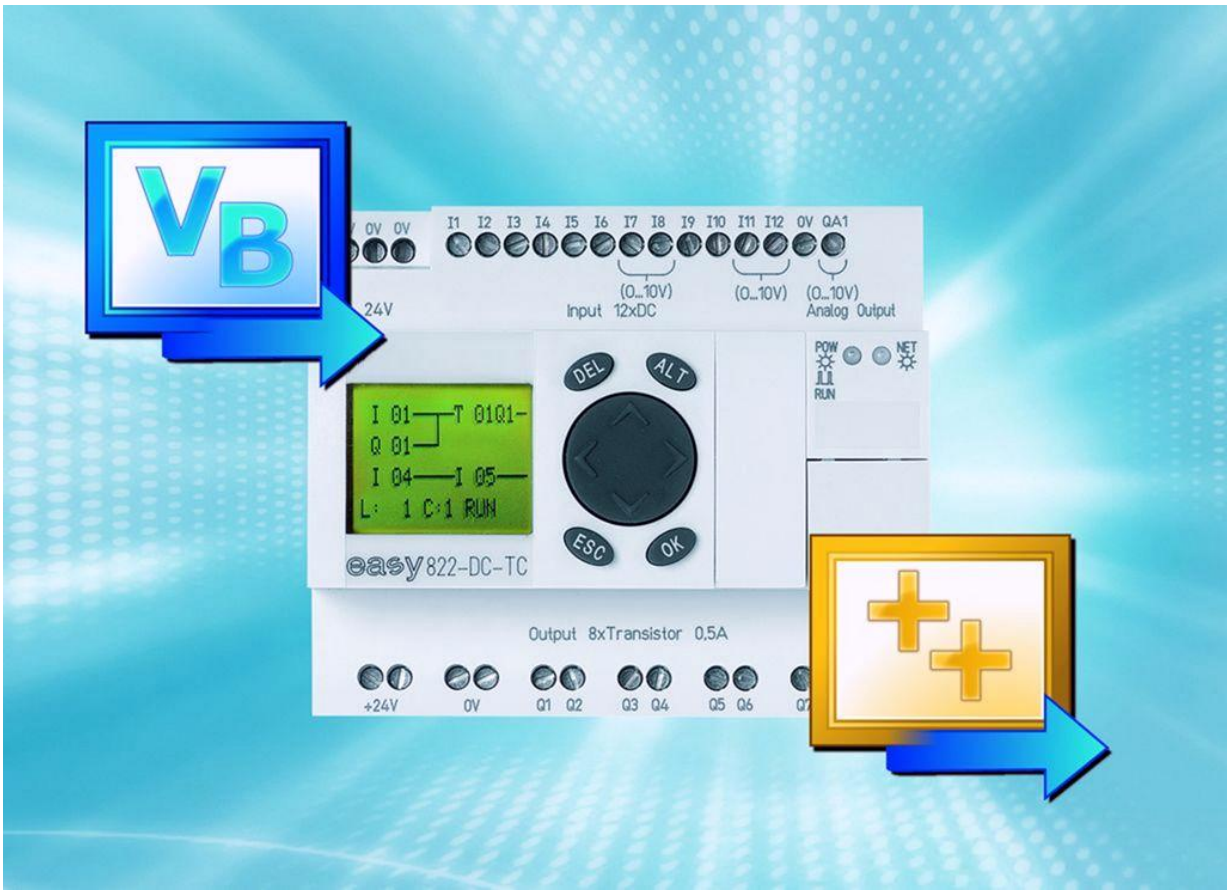
Application Notes

EASY_COM function library V2.5



11/11 AM_EASY_COM_G

Author: E. Kastner
easy@eaton.com

Table of Contents

1. Introduction

The function library "EASY_COM.dll" contains functions for the communication with easy500, easy700, easy800 and MFD-Titan.

The function library is intended for programmers who wish to create their own Windows-based visualization solutions without too much effort. Therefore it deliberately does not cover the complete function range that easySoft or easy-OPC-Server offers.

Functions overview

- Connection via a COM port (serial interface or USB converter)
  with automatic test of the suitable Baud rates
- Connection via Ethernet or TCP/IP
- Simultaneous operation of several open connections.
- Communication with all easyNet stations via an open connection (Routing)
- Lock and unlock a device with system password
- Start and stop the program process
- Reading and setting of the device clock
- Reading of the process image
- Writing in the marker range
- Reading and writing from year time and 7-day time switch channels

In order to simplify the use of the function library, communication functions are provided in two different function groups:

In "**single connection mode**", a maximum of only one open connection is supported within an application or process. Opening another connection automatically closes the previous one.

In "**multiple connection mode**", several connections can be opened and used simultaneously, provided that they do not address the same interface or the same device. When a new connection is opened, a handle is returned. This handle must be declared when other communication functions are called. The connection is selected with the corresponding handle. Communication functions remain the same otherwise when compared to those of the single connection mode.

Note: Only one single connection is required for simultaneous access to all the nodes on an easyNet line.

2. License conditions

Eaton Industries GmbH grants you an unlimited, non exclusive, royalty-free license to use the Software EASY_COM.dll for the development of your own software, to duplicate the Software EASY_COM.dll and to distribute it in connection with your developed software. Further, we grant you the right to sublicense your rights to any third party to the extent granted to you under these conditions

You agree,

1. to distribute the Software EASY_COM.dll only in connection with and as part of your own developed software,
2. not to use the name, the logo or trademark of Eaton Industries GmbH to market you own developed software,
3. not to remove or garble any indication to copyright, trademarks or patents which appear on the Software EASY_COM.dll as it was delivered to you as well as
4. to indemnify us from any claims made by third parties against Eaton Industries GmbH, arising from the use or the distribution of your developed software.

Eaton Industries's liability – based on whatever legal reason – is excluded. This exclusion shall not apply in case of mandatory liability according to the German Product Liability Act ("Produkthaftungsgesetz"), in case of intent, gross negligence, fraudulently conciliation of a defect, non-compliance with guaranteed characteristics, loss of life, bodily injury or damage to health or in case of a breach of a stipulation which goes to the root of the contract. However, claims for damages arising from a breach of a stipulation which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on liability for loss of life, bodily injury or damage to health.

3. Using the Function Library

3.1 System requirements

As part of a 32-bit application the EASY_COM.dll can be used under following Windows versions:

- Windows 2000 from SP4
- Windows XP from SP2
- Windows Vista (32-Bit)
- Windows 7 (32-Bit and 64-Bit)

The EASY_COM.dll supports all device families that have appeared up to now

- easy500
- easy700
- easy800
- MFD-Titan

For the access to the easyNet station the easyNet must already be in operation.

Also for Ethernet or TCP/IP connection via the EASY209-SE or an onboard Ethernet interface the connection parameters must already be correctly entered into the device. The device will not be configured from the EASY_COM.dll.

3.2 Use with Microsoft Visual C++

The function library can be used for Microsoft Visual C++ V6.0 SP5 and also for Microsoft Visual Studio 2005/2008/2010.

For generating your own projects the source code header file “easyComApi.h” and also the import library file “EASY_COM.lib“ are necessary and are included in the packet. These files should be filed locally in the project.

For execution of the generated application the EASY_COM.dll is necessary, as well as, “Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)”. Possibly additional redistributable packages are required, if the application was created with a different version of Visual C++.

The EASY_COM.dll should be filed in the application directory.

The notification of the interface function is carried out by the #include instruction on the header file (if necessary the include search path in the preprocessor settings must be modified). In the linker settings "Object/library module" or "additional dependencies" for Visual Studio 2005/2008/2010 the entry "EASY_COM.lib" must be added.

The EASY_COM.dll is generated with the compiler settings

Platform Win32,
Multithreaded DLL,
no Unicode,
no C++ Exceptions

3.3 Use with .NET

The function library can be used with Microsoft Visual Studio 2005, 2008, 2010.

The source code "easyComApi.cs" (single connection mode) or "easyComApi_MC.cs" (multiple connection mode) that is delivered with the package is required for generating your own .NET projects. These files must be entered in the solution project as an "available element". Configure the target platform option as "x86", so the application runs on both 32-bit as on 64-bit systems.

For execution of the generated application the EASY_COM.dll is necessary, as well as, "Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)". Possibly additional redistributable packages are required, if the application was created with a different version of Visual Studio.

The EASY_COM.dll should be filed in the application directory.

The source code files define the "easy_COM_API" and "easy_COM_API_MC" classes, which were written in C#, with static methods. The names and parameters of these methods correspond to the functions from EASY_COM.dll.
Since EASY_COM.dll is an unmanaged code DLL, the entry points of the DLL must be specified with "DllImport" statements. The delivered classes do not allow for the entry points to be called directly, but instead encapsulate them in additional methods in order to be able to trap possible exception errors, e.g. `System.DllNotFoundException` or `System.EntryPointNotFoundException`.

With regard to the new methods the conventions are valid as described in the chapter "Description of Interface Functions".

3.3.1 Use with Microsoft Visual Basic 2005/2008/2010 (VB.NET)

For generating your own projects the source code header file "easyComApi.vb" is necessary and is included in the packet. This file must be entered in the solution project as an "available element". Configure the target platform option as "x86", so the application runs on both 32-bit as on 64-bit systems.

For execution of the generated application the EASY_COM.dll is necessary, as well as, "Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)". Possibly additional redistributable packages are required, if the application was created with a different version of Visual Basic.

The EASY_COM.dll should be filed in the application directory.

The source code file defines a class named "easyCOM" with static methods. The names and parameters of these methods correspond to the functions from EASY_COM.dll.

3.4 Use with Microsoft Visual Basic 6.0

For generating your own projects the source code header file "easyComApi.bas" is necessary and is included in the packet. This file must be entered in the Visual Basic Project as an "available module".

For execution of the generated application the EASY_COM.dll is necessary, as well as, "Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)".

The EASY_COM.dll should be filed in the application directory or in the Windows system directory.

3.5 Use with Java SE Development Kit 6

For generating your own projects the source code files "easyCom_API.java" or "easyCom_API_MC.java" is necessary and is included in the packet. Both files use the library "Java Native Access" (JNA) to simplify the declaration of the easyCOM interface (s. http://jna.java.net).

The path of the JNA library files must be added in both development and execution environment.

Example:
Compilation: javac.exe -classpath jna.jar;platform.jar;. easyCOM_API.java
Execution: java.exe -classpath jna.jar;platform.jar;. easyCOM_API

For execution of the generated application the EASY_COM.dll is necessary, as well as, "Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)".

The EASY_COM.dll should be filed in the application directory.

3.6 Equipment supplied

The function library is supplied as a Zip file. The Zip file contains the following files:

EASY_COM.dll	The executable function library as Dynamic Link Library
EASY_COM.lib	Import library file for MS Visual C++ V6.0 or MS Visual C++ 2005/2008/2010
Doku\easyComApi.h	Header file in C/C++ with the declaration of the library interface
Doku\EasyComApi.vb	Visual Basic 2005/2008/2010 source code file with declaration of the library interface for integration in .NET
Doku\easyComApi.bas	Visual Basic 6 source code file with the declaration of the library interface
Doku\easyComApi.cs Doku\easyComApi_MC.cs	C# source code files with class definitions for integration of the library interface in .NET
Doku\easyCOM_API.java Doku\easyCOM_API_MC.java	Java source code files with class definitions for integration of the library interface using Java Native Access
Doku\AM_EASY_COM_D.pdf	Application note to library, German version
Doku\AM_EASY_COM_G.pdf	Application note to library, engl. Version (this document)
Demo.cpp	The directory contains an MS Visual Studio 2008 project for a dialog-based MFC application in C++ that demonstrates the use of the function library.
DemoVB.NET	The directory contains an MS Visual Basic 2008 project for a dialog-based .NET application that demonstrates the use of the function library. It can also be used for MS Visual Basic 2010 (Express).

4. Description of Interface Functions

The function description starts with a declaration in the programming language C.
This begins with the data type of the return value followed by the function name and the parameter list in parentheses (data type of the parameter followed by the parameter name).
A star * between data type and parameter name indicates that the parameter is a pointer of a single data element or an array of data elements.
In this case, a data structure in sufficient size must be allocated (note parameter description); its memory address or reference must be passed to the function.

The following table of data types provides assistance in porting the interface to other programming languages:

Data Type in C	Meaning
unsigned char	8-Bit, unsigned
unsigned char*	Pointer of 8-Bit Item or memory block
unsigned short	16-Bit, unsigned
long	32-Bit, signed
long*	pointer of 32-Bit item
unsigned long	32-Bit, unsigned
bool	Boolean (true, false), 8-Bit size
const char*	ASCII-string with null termination
tEasyComHandle	untyped pointer
tEasyComHandle*	pointer of pointer

Data type in C / C++	Data type in C#	Data type in VB.NET	Data type in Visual Basic 6	Data type in Java Native Access
unsigned char	byte	ByVal Byte	ByVal Byte	byte[2]
unsigned char*	ref byte[1]	ByRef Byte	ByRef Byte	ByteByReference[3]
unsigned short	ushort	ByVal UShort	ByVal Integer	short[2]
long	int	ByVal Integer	ByVal Long	int
long*	out int	ByRef Integer	ByRef Long	IntByReference
unsigned long	uint	ByVal Integer[2]	ByVal Long[2]	int[2]
bool	bool	ByVal Boolean	ByVal Boolean	byte
const char*	String[4]	ByVal String	ByVal String	String
tEasyComHandle	IntPtr	ByVal Integer	ByVal Integer	Pointer
tEasyComHandle*	out IntPtr	ByRef Integer	ByRef Integer	PointerByReference

[1] if memory block: [MarshalAs(UnmanagedType.LPArray)] byte[]
[2] signed data type, since no exact correlation exists
[3] if memory block: byte[]
[4] [MarshalAs(UnmanagedType.LPStr)] String

4.1 Functions for single connection mode

Open_ComPort

long Open_ComPort (unsigned char com_port_no, long baud rate)

Make a connection to the device via the stated COM port.

The function opens the requested serial interface and configures it for the easyCOM protocol. Instead of a serial interface it could also be a virtual COM port of an interface converter especially an EASY(800)-USB-CAB programming cable.

If no connection to the device can be made a connection is attempted with all possible Baud rates. If the connection can be made to the device the device will be exclusively reconfigured to the required Baud rate. Which Baud rate is then set depends upon the ability of the device and the programming cable used. With a successful connection the actual set Baud rate can be determined with the function GetCurrent_Baudrate.

The easy500 and easy700 device families only support 4800 Baud.

If a connection is already open this will firstly be closed as several open connections cannot be supported inside an application.

If the device's interface is protected with a system password, the function returns error code 4. The connection with the device has been established, but the Unlock_Device function must be called afterwards in order to be able to keep communicating with the device.

Parameter:

com_port_nr Number of the COM port, that is to be opened e.g. 3 for COM3 { 1…255 }
baudrate required baud rate { 4800, 9600, 19200, 38400, 57600 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.
2 Device sends an error message.
3 Device does not respond.
4 The device's interface is protected with a password. Call up Unlock_Device!
5 Device type is unknown.
6 Error message from Windows. Interrogate error code with GetLastSystemError.
8 The COM port is not present or not registered in the system at the moment (connection break)
9 The COM port is blocked by another process or is not registered in the system (connection break).
10 General communication error occurred (possible hardware failure)
11 Internal error of EASY_COM.dll

Open_EthernetPort

long Open_EthernetPort (const char * szIpAddress, long IpPort, long baudrate, bool no_baudrate_scan)

Make a connection to the device via Ethernet or Internet.

The function opens the requested TCP/IP connection and configures it for the easyCOM protocol. The TCP/IP connection must have been configured beforehand (see System requirements).

If no connection to the device can be made a connection is attempted with all possible Baud rates. If the connection can be made to the device the device will be exclusively reconfigured to the required Baud rate. Which Baud rate is then set depends upon the ability of the device and the programming cable used. With a successful connection the actual set Baud rate can be determined with the function GetCurrent_Baudrate.

Because of the long TCP/IP waiting time the baud rate test may take up to a minute. Because of this, the "no_baudrate_scan" parameter should be set to "true" for easy500 and easy700 in order to disable the baud rate test (generally, the devices support 4800 baud only).
This also applies if the connection is not to be established via the EASY209-SE Ethernet gateway. In this case the device and converter must already be configured to the given baud rate or the device has an Ethernet interface so that the baud rate is unimportant for the connection.

The TCP/IP port numbers are device dependent: For easy and MFD-Titan that are connected via a EASY209-SE gateway, the value range 10001 to 10999 is possible.

If a connection is already open this will firstly be closed as several open connections cannot be supported inside an application.

Parameter:

szIpAddress IPv4 address as 0 terminated string. As a rule 4 digit e.g. "10.1.41.31"
IpPort - IP port number { 1200, 10001...10999 }
baudrate Baud rate set in device { 4800, 9600, 19200, 38400, 57600 }
no_baudrate_scan
Switches the baud rate test off. "true" => only the given baud rate will be tested

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.
2 Device sends an error message.
3 Device does not respond.
4 The device's interface is protected with a password. Call up Unlock_Device!
5 Device type is unknown.
6 Error message from Windows. Interrogate error code with GetLastSystemError.
10 General communication error occurred (possible hardware failure)
11 Internal error of EASY_COM.dll
14 TCP/IP station doesn't respond or the TCP/IP port is already occupied by another opened connection
15 Baud rate test is not possible

See also:

GetCurrent_Baudrate
GetLastSystemError

Close_ComPort

long Close_ComPort ( )

Close the connection to the device.

The function closes the connection to the device and makes the serial interface free.

Return Values:

0 Function call up successful.

Close_EthernetPort

long Close_EthernetPort ()

Close the connection to the device.

The function closes the connection to the device and makes the Ethernet connection free.

Return Values:

0 Function call up successful.

GetCurrent_Baudrate

long GetCurrent_Baudrate (long * baudrate)

Returns the set baud rate of the current connection

After calling up Open_ComPort or Open_EthernetPort the function gives the actual set baud rate. Which baud rate is set at connection set-up depends upon the capability of the device, the programming cable used, the type of connection and possibly the capability of the converter in between.

After calling up Open_EthernetPort without baud rate test the function's return value is undefined.

Parameter:

baudrate
Pointer on a word variable to save the current baud rate { 4800, 9600, 19200, 38400, 57600 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.
7 No connection open.

Set_UserWaitingTime

long Set_UserWaitingTime (long timeout_delay)

Sets an additional timeout delay

The function sets an additional delay time which must elapse with every device request before a connection abort is reported.

Longer wait times must be set on the PC side as wireless or modem connections have a longer transmission route. The function should be called before the function call for opening a connection.

A value higher than 0 also causes a compulsory delay until the occurrence of an actual connection breakdown is detected. The value should therefore not be greater than is absolutely necessary.

Different delays occur depending on the transmission route so that a general guideline value for the delay time cannot be given. For example, an initial test run can be started with 800ms. If connection aborts still occur, the value should be increased gradually.

0 is set as an initial value for direct serial connections between the PC and the device.

Parameter:

timeout_delay delay time in milliseconds { 0…90000 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.

Get_UserWaitingTime

long Get_UserWaitingTime (long* timeout_delay)

Returns the last selected additional timeout delay

Parameter:

timeout_delay Pointer on a variable to saving the delay time { 0…90000 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.

GetLastSystemError

unsigned long GetLastSystemError ()

Delivers the last error code transmitted from Windows

If one of the communication functions returns an error code 6 the precise error code from Windows can be determined. This error code corresponds to the return value of Windows system function GetLastError. A description of the System Error Codes can be found for example in the MSDN:

http://msdn2.microsoft.com/en-us/library/ms681381.aspx

Return Values:

Windows System Error Code

Start_Program

long Start_Program (unsigned char net_id, unsigned char * errorcode)

Switch the device to "RUN".

The function switches the device into the RUN mode. If the display on the device is not in the normal position this will be automatically reset (user settings not yet taken over will be lost)

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Caution:

For the change to RUN, the device can take several seconds!
If the device is an easyNet Master (Station NT1), using the "Remote RUN" - option further devices can be switched into "RUN".

Parameter:

net_id	Target device {0...8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Stop_Program

long Stop_Program (unsigned char net_id, unsigned char * errorcode)

Switch the device to "STOP".

The function switches the device into the STOP mode. If the display on the device is not in the normal position this will be automatically reset (user settings not yet taken over will be lost)

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Caution:

If the device is an easyNet Master (Station NT1), using the "Remote RUN" - option further devices can be switched into "STOP".

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Read_Clock

```
long Read_Clock (
  unsigned char net_id, unsigned char* year, unsigned char* month, unsigned char* day,
  unsigned char* hour, unsigned char* min)
```

Reading the device clock

The function reads the date and time from the device.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "year" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
year	Pointer on the Byte variable to save the year number {0…99}, 0 corresponds to year 2000
month	Pointer on the Byte variable to save the month {1…12}
day	Pointer on a Byte variable to save the calendar day {1…31}
hour	Pointer on a Byte variable to save the hours {0…23}, 24h format
min	Pointer on a Byte variable to save the minutes {0…59}

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Write_Clock

```
long Write_Clock (
  unsigned char net_id, unsigned char* year, unsigned char* month, unsigned char* day,
  unsigned char* hour, unsigned char* min)
```

Set the device clock.

The function writes the date and time into the device.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "year" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
year	Pointer on the Byte variable to save the year number {0…99}, 0 corresponds to year 2000
month	Pointer on the Byte variable to save the month {1…12}
day	Pointer on a Byte variable to save the calendar day {1…31}
hour	Pointer on a Byte variable to save the hours {0…23}, 24h format
min	Pointer on a Byte variable to save the minutes {0…59}

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Read_Object_Value

long Read_Object_Value(
unsigned char net_id, unsigned char object, unsigned short index, unsigned char * data)

Reading of the process image.

The function reads a part of the process image from the device.

The parameter value "object" determines the range of the process image and therefore amount of data read:

0	Digital inputs, basic device	I1…I16	2 Byte
1	Digital outputs, basic device	Q1…Q8	2 Byte
2	Digital inputs, expansion module	R1…R16	2 Byte
3	Digital outputs, expansion module	S1…S8	2 Byte
4	Bit marker for easy800 and MFD-Titan	M1…M96	12 Byte
4	Bit marker for easy500 and easy700	M1…M16	2 Byte
4	Bit marker for easy800-DC-SWD	M1…M128	16 Byte
5	P buttons	P1…P4	1 Byte
6	easyNet inputs	xI1… xI16 and xR1…xR16	4 Byte
7	easyNet outputs	xQ1…xQ8 and xS1…xS8	4 Byte
8	Analog inputs basic unit	IA1…IA4	8 Byte
9	Analog output, basic device	QA1	2 Byte
10	8 Marker double word	MDx…MDx+7	32 Byte
11	easyNet Receive data	xRN1…xRN32	32 Byte
12	easyNet Send data	xSN1…xSN32	32 Byte
13	Diagnostic bits	ID1…ID16	2 Byte
14	Expanded diagnostic bits for MFD-Titan	ID17…ID32	2 Byte
15	Programmable LEDs for MFD-Titan	LE1…LE3	1 Byte
16	Additional Bit marker for easy500/700	N1…N16	2 Byte
17	32 marker bytes	MBx…MBx+31	32 Byte
18	Digital inputs, basic device and assigned SWD inputs	I1…I128	16 Byte
19	Digital outputs, basic device and assigned SWD outputs	Q1…Q128	16 Byte

For further information take a look in the operating manual of the device.

The status value is returned in Little-Endian-Format (Intel format). The status value of the operand with the smallest operand number is in the Bit with the least value (LSB).

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code transmitted from the device is in the first Byte of the parameter value "data" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
object	Number of the object, therefore the process range
index	Node number for objects 6,7,11,12 {1…8} Operand number for objects 10 {1…96} or {1…128} Byte number for object 17 {1…384} or {1…512} No function otherwise.
data	Pointer on a memory area to save the status values

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Caution:

The size of the data block that is to be transmitted with "data" must be sufficiently dimensioned that the memory limits are not exceeded. With object 10 generally 8 double-words are read, except the memory limits are exceeded (then correspondingly less Byte are registered).
Object 17 allows byte-wise access to the entire marker range, even if the device only knows byte access for the lower area.

The MC_Read_Object_Value makes it possible for objects 10 and 17 to read a selectable amount of data.

Write_Object_Value

```
long Write_Object_Value (
  unsigned char net_id, unsigned char object, unsigned short index, unsigned char length, unsigned char * data)
```

Write in the marker range.

The function writes values into marker range of the device.

The parameter "object" determines the type of the marker range:

4	individual Bit marker Mx	1 Byte
10	1 to 80 bytes, starting with the MDx double word marker	1…80 Byte
16	Individual Bit marker Nx for easy500 and easy700	1 Byte
17	1 to 80 marker bytes	1…80 Byte

The status value is interpreted in Little-Endian-Format (Intel format). The status value of the Bit marker to be set must be entered in the least value Bit (LSB).

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code transmitted from the device is in the first Byte of the parameter value "data" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
object	Number of the object, therefore the process range
index	Operand number for objects 4, 10, 16 {1…16} or {1…96} or {1…128} Byte number for object 17 {1…384} or {1…512} No function otherwise.
length	For objects 10 and 17, specifies the number of bytes to be written {1…80}. No function otherwise.
data	Pointer on a memory area to save the status values

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or not registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Caution:

The size of the data block that is to be transmitted with the parameter "data" must be sufficiently dimensioned that the memory limits are not exceeded.
The status of a written marker only remains retained when the device does not process a program that has writing access on this marker.
Object 17 allows byte-wise access to the entire marker range, even if the device only knows byte access for the lower area.

Read_Channel_YearTimeSwitch

```
long Read_Channel_YearTimeSwitch(
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * on_year, unsigned char * on_month, unsigned char * on_day,
  unsigned char * off_year, unsigned char * off_month, unsigned char * off_day)
```

Read a parameter of the year time switch channel.

The function reads the parameter set of the selected channel from the given year time switch element. The parameter set consists of "calendar day", "month" and "year" each separated according to switch-on time-point and switch-off time-point.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "on_year" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
on_year	Pointer on a Byte variable to save the *switch-on point* - year number {0, 1…99}, 0 corresponds to "not occupied"
on_month	Pointer on a Byte variable to save the *switch-on point* - months {0, 1…12}, 0 corresponds to "not occupied"
on_day	Pointer on a Byte variable to save the *switch-on point* - calendar day {0, 1…31}, 0 corresponds to "not occupied"
off_year	Pointer on a Byte variable to save the *switch-off point* - year number {0, 1…99}, 0 corresponds to "not occupied"
off_month	Pointer on a Byte variable to save the *switch-off point* - months {0, 1…12}, 0 corresponds to "not occupied"
off_day	Pointer on a Byte variable to save the *switch-off point* – calendar day {0, 1…31}, 0 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No COM port is opened.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.

Caution:

The easy800 and MFD8 generally only give zero values when the device is in STOP.

Write_Channel_YearTimeSwitch

```
long Write_Channel_YearTimeSwitch(
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * on_year, unsigned char * on_month, unsigned char * on_day,
  unsigned char * off_year, unsigned char * off_month, unsigned char * off_day)
```

Overwrite the parameter of a year time switch channel.

The function overwrites the parameter set of the selected channel from the given year time switch with values. The parameter set consists of "calendar day", "month" and "year" each separated according to switch-on time-point and switch-off time-point.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "on_year" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
on_year	Pointer on a Byte variable to save the *switch-on point* - year number {0, 1…99}, 0 corresponds to "not occupied"
on_month	Pointer on a Byte variable to save the *switch-on point* - months {0, 1…12}, 0 corresponds to "not occupied"
on_day	Pointer on a Byte variable to save the *switch-on point* - calendar day {0, 1…31}, 0 corresponds to "not occupied"
off_year	Pointer on a Byte variable to save the *switch-off point* - year number {0, 1…99}, 0 corresponds to "not occupied"
off_month	Pointer on a Byte variable to save the *switch-off point* - months {0, 1…12}, 0 corresponds to "not occupied"
off_day	Pointer on a Byte variable to save the *switch-off point* – calendar day {0, 1…31}, 0 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No COM port is opened.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
13	The channel cannot be overwritten because there is no parameter there.

Caution:

With easy800 and MFD-Titan, at least a part of the given channel must be occupied or the channel will be seen as not used.

Read_Channel_7DayTimeSwitch

```
long Read_Channel_7DayTimeSwitch(
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * DY1, unsigned char * DY2,
  unsigned char * on_hour, unsigned char * on_minute,
  unsigned char * off_hour, unsigned char * off_minute)
```

Read a parameter of the 7-day time switch channel.

The function reads the parameter of the selected channel from the given 7-day time switch element.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "DY1" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
DY1	Pointer on a Byte variable to save the weekday parameter DY1 {0=SO…6=SA}, 255 corresponds to "not occupied"
DY2	Pointer on a Byte variable to save the weekday parameter DY2 {0=SO…6=SA}, 255 corresponds to "not occupied"
on_hour	Pointer on a Byte variable to save the switch-on hour {0…23}, 255 correspond to "not occupied"
on_minute	Pointer on a Byte variable to save the switch-on minute {0…59}, 255 correspond to "not occupied"
off_hour	Pointer on a Byte variable to save the switch-off hour {0…23}, 255 corresponds to "not occupied"
off_minute	Pointer on a Byte variable to save the switch-off minute {0…59}, 255 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.

Caution:

The easy800 and MFD8 generally only give zero values when the device is in STOP.

Write_Channel_7DayTimeSwitch

```
long Write_Channel_7DayTimeSwitch(
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * DY1, unsigned char * DY2,
  unsigned char * on_hour, unsigned char * on_minute,
  unsigned char * off_hour, unsigned char * off_minute)
```

Overwrite a parameter of the 7-day time switch channel.

The function overwrites the parameter set of the selected channel from the given 7-day time switch with new values.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "DY1" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
DY1	Pointer on a Byte variable to save the weekday parameter DY1 {0=SO…6=SA}, 255 corresponds to "not occupied"
DY2	Pointer on a Byte variable to save the weekday parameter DY2 {0=SO…6=SA}, 255 corresponds to "not occupied"
on_hour	Pointer on a Byte variable to save the switch-on hour {0…23}, 255 correspond to "not occupied"
on_minute	Pointer on a Byte variable to save the switch-on minute {0…59}, 255 correspond to "not occupied"
off_hour	Pointer on a Byte variable to save the switch-off hour {0…23}, 255 corresponds to "not occupied"
off_minute	Pointer on a Byte variable to save the switch-off minute {0…59}, 255 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
13	The channel cannot be overwritten because there is no parameter there.

Caution:

With easy800 and MFD-Titan, at least a part of the given channel must be occupied or the channel will be seen as not used.

Unlock_Device

long Unlock_Device (unsigned char net_id, const char * szPassword, unsigned char * errorcode)

Unlocks a device that is protected by a system password.

If a password is active on the device, the device can be unlocked with this function. For this purpose, the system password stored in the device must be passed in the "szPassword" parameter as a decimal value string. In the case of easy500 and easy700, the password consists of four decimal digits, whereas the password for easy800 and MFD-Titan consists of six decimal digits. The password is transmitted to the device in encrypted fashion and verified there. If the password does not correspond to the password stored in the device, an additional delay pause is used in order to hinder automatic testing of all combinations.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID. .
If a password is active on all easyNet nodes and communications with all of them are necessary, Unlock_Device must be called separately for each node.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Parameter:

net_id	Target device {0…8}; 0 for local programming interface
szPassword	Device system password as 0 terminated chain {"0001"…"9999" resp. "000001"…"999999"}
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

Caution:

When the device is unlocked the previously locked menu items can called up. After approx. 10 minutes inactivity the device automatically reactivates the password protection.

Lock_Device

long Lock_Device (unsigned char net_id, unsigned char * errorcode)

Reactivates the system password of the device.

The password protection of a device can be reactivated when a device has a password and has been previously unlocked.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If a password is active on all easyNet nodes and all nodes have been unlocked, the function must be called separately for each node.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Parameter:

net_id	Target device {0...8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

4.2 Functions for multiple connection mode

In multiple connection mode, several connections can be opened and used simultaneously, provided that they do not address the same interface or the same easy device. When a new connection is opened, a handle is returned. This handle must be declared when other communication functions are called. The connection is selected with the corresponding handle.

Aside from the additional handle parameter, the functions remain the same when compared to those of the single connection mode. Because of this, the corresponding function names start with the "MC_" prefix.

```
typedef void* tEasyComHandle;
```

MC_CloseAll

```
long MC_CloseAll( )
```

Closes all easy-Com connections

The function ends all connections that are still open and releases the corresponding system resources. It can also be used to close the connection that was opened in single connection mode.

Return Values:

0 Function call up successful.

MC_Open_ComPort

long MC_Open_ComPort(tEasyComHandle *phandle, unsigned char com_port_nr, long baudrate)

Make a connection to the device via the stated COM port.

The function opens the requested serial interface and configures it for the easy-COM protocol. Instead of a serial interface it could also be a virtual COM port of an interface converter especially a EASY(800)-USB-CAB programming cable.

If no connection to the device can be made a connection is attempted with all possible Baud rates. If the connection can be made to the device the device will be exclusively reconfigured to the required Baud rate. Which Baud rate is then set depends upon the ability of the device and the programming cable used. With a successful connection the actual set Baud rate can be determined with the function MC_GetCurrent_Baudrate.

The easy500 and easy700 device families only support 4800 Baud.

If the connection was established successfully, the "p_hPort" parameter is used to return a handle value. This handle value is used to select the connection in the subsequent communication call.

If the device's interface is protected with a system password, the function returns error code 4. The connection with the device has been established, but the MC_Unlock_Device function must be called afterwards in order to be able to keep communicating with the device.

Parameter:

phandle	Pointer to a handle variable
com_port_nr	Number of the COM port, that is to be opened e.g. 3 for COM3 { 1...255 }
baudrate	required Baud rate { 4800, 9600, 19200, 38400, 57600 }

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
4	The device's interface is protected with a password.
5	Device type is unknown.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
8	The COM port is not present or not registered in the system at the moment (connection break)
9	The COM port is blocked by another process or not registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll

MC_Open_EthernetPort

```
long MC_Open_EthernetPort (
  tEasyComHandle *phandle, const char * szIpAddress, long IpPort, long baudrate, bool no_baudrate_scan)
```

Make a connection to the device via Ethernet or Internet.

The function opens the requested TCP/IP connection and configures it for the easy-COM protocol. The TCP/IP connection must have been configured beforehand (see System requirements).

If no connection to the device can be made a connection is attempted with all possible Baud rates. If the connection can be made to the device the device will be exclusively reconfigured to the required Baud rate. Which Baud rate is then set depends upon the ability of the device and the programming cable used. With a successful connection the actual set Baud rate can be determined with the function MC_GetCurrent_Baudrate.

Because of the long TCP/IP waiting time the baud rate test may take up to a minute. Because of this, the "no_baudrate_scan" parameter should be set to "true" for easy500 and easy700 in order to disable the baud rate test (generally, the devices support 4800 baud only).
This also applies if the connection is not to be established via the EASY209-SE Ethernet gateway. In this case the device and converter must already be configured to the given baud rate or the device has an Ethernet interface so that the baud rate is unimportant for the connection.

The TCP/IP port numbers are device dependent: For easy and MFD-Titan that are connected via a EASY209-SE gateway, the value range 10001 to 10999 is possible.

If the connection was established successfully, the "phandle" parameter is used to return a handle value. This handle value is used to select the connection in the subsequent communication call.

Parameter:

phandle	Pointer to a handle variable
szIpAddress	IPv4 address as 0 terminated string. As a rule 4 digit e.g. "10.1.41.31"
IpPort -	IP port number { 1200, 10001...10999 }
baudrate	Baud rate set in device { 4800, 9600, 19200, 38400, 57600 }
no_baudrate_scan	Switches the baud rate test off. "true" => only the given baud rate will be tested

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
4	The device's interface is protected with a password. Call up MC_Unlock_Device!
5	Device type is unknown.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
14	TCP/IP station doesn't respond or the TCP/IP port is already occupied by another opened connection
15	Baud rate test is not possible

MC_Close_ComPort

long MC_Close_ComPort (tEasyComHandle handle)

Close the connection to the device.

The function closes the connection to the device and makes the serial interface free.

Parameter:

handle The connection handle returned when a connection is opened

Return Values:

0 Function call up successful.
16 The connection handle is not valid (anymore)

MC_Close_EthernetPort

long MC_Close_EthernetPort (tEasyComHandle handle)

Close the connection to the device.

The function closes the connection to the device and makes the Ethernet connection free.

Parameter:

handle The connection handle returned when a connection is opened

Return Values:

0 Function call up successful.
16 The connection handle is not valid (anymore)

MC_GetCurrent_Baudrate

long MC_GetCurrent_Baudrate (tEasyComHandle handle, long * baudrate)

Returns the set baud rate of the specified connection.

After calling up MC_Open_ComPort or MC_Open_EthernetPort the function gives the actual set baud rate. Which baud rate is set at connection set-up depends upon the capability of the device, the programming cable used, the type of connection and possibly the capability of the converter in between

After calling up MC_Open_EthernetPort without a baud rate test, the function's return value is undefined.

Parameter:

handle The connection handle returned when a connection is opened
baudrate Pointer on a word variable to save the current baud rate { 4800, 9600, 19200, 38400, 57600 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.
16 The connection handle is not valid (anymore)

MC_Set_UserWaitingTime

long Set_UserWaitingTime (long timeout_delay)

Sets an additional timeout delay

The function sets an additional delay time which must elapse with every device request before a connection abort is reported. This setting applies to all connections that are opened with the DLL.

Longer wait times must be set on the PC side as wireless or modem connections have a longer transmission route. The function should be called before the function call for opening a connection.

A value higher than 0 also causes a compulsory delay until the occurrence of an actual connection breakdown is detected. The value should therefore not be greater than is absolutely necessary.

Different delays occur depending on the transmission route so that a general guideline value for the delay time cannot be given. For example, an initial test run can be started with 800ms. If connection aborts still occur, the value should be increased gradually.

0 is set as an initial value for direct serial connections between the PC and the device.

Parameter:

timeout_delay delay time in milliseconds { 0…90000 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.

MC_Get_UserWaitingTime

long Get_UserWaitingTime (long* timeout_delay)

Returns the last selected additional timeout delay

Parameter:

timeout_delay Pointer on a variable to saving the delay time { 0…90000 }

Return Values:

0 Function call up successful.
1 A parameter contains an invalid value.

MC_Start_Program

long MC_Start_Program (tEasyComHandle handle, unsigned char net_id, unsigned char * errorcode)

Switch the device to "RUN".

The function switches the device into the RUN mode. If the display on the device is not in the normal position this will be automatically reset (user settings not yet taken over will be lost)

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Caution:

For the change to RUN, the device can take several seconds!
If the device is an easyNet Master (Station NT1), using the "Remote RUN" - option further devices can be switched into "RUN".

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

MC_Stop_Program

long MC_Stop_Program (tEasyComHandle handle, unsigned char net_id, unsigned char * errorcode)

Switch the device to "STOP".

The function switches the device into the STOP mode. If the display on the device is not in the normal position this will be automatically reset (user settings not yet taken over will be lost)

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Caution:

If the device is an easyNet Master (Station NT1), using the "Remote RUN" - option further devices can be switched into "STOP".

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

MC_Read_Clock

long MC_Read_Clock(tEasyComHandle handle, unsigned char net_id,
unsigned char* year, unsigned char* month, unsigned char* day, unsigned char* hour, unsigned char* min)

Reading the device clock.

The function reads the date and time from the device.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "year" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
year	Pointer on the Byte variable to save the year number {0…99}, 0 corresponds to year 2000
month	Pointer on the Byte variable to save the month {1…12}
day	Pointer on a Byte variable to save the calendar day {1…31}
hour	Pointer on a Byte variable to save the hours {0…23}, 24h format
min	Pointer on a Byte variable to save the minutes {0…59}

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

MC_Write_Clock

long MC_Write_Clock (tEasyComHandle handle, unsigned char net_id,
unsigned char* year, unsigned char* month, unsigned char* day, unsigned char* hour, unsigned char* min)

Set the device clock.

The function writes the date and time into the device.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "year" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
year	Pointer on the Byte variable to save the year number {0…99}, 0 corresponds to year 2000
month	Pointer on the Byte variable to save the month {1…12}
day	Pointer on a Byte variable to save the calendar day {1…31}
hour	Pointer on a Byte variable to save the hours {0…23}, 24h format
min	Pointer on a Byte variable to save the minutes {0…59}

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

MC_Read_Object_Value

long MC_Read_Object_Value (tEasyComHandle handle, unsigned char net_id,
unsigned char object, unsigned short index, unsigned char length, unsigned char * data)

Readout of process image or marker range

The function reads a range belonging to the process image or to the marker range from the device.
The "object" parameter value defines the range of the process image and, as a result, the amount of data read (exception: the „length" parameter is used to specify the data volume read in objects 10 and 17).

0	Digital inputs, basic device	I1…I16	2 Byte
1	Digital outputs, basic device	Q1…Q8	2 Byte
2	Digital inputs, expansion module	R1…R16	2 Byte
3	Digital outputs, expansion module	S1…S8	2 Byte
4	Bit marker for easy800 and MFD-Titan	M1…M96	12 Byte
4	Bit marker for easy500 and easy700	M1…M16	2 Byte
4	Bit marker for easy800-DC-SWD	M1…M128	16 Byte
5	P buttons	P1…P4	1 Byte
6	easyNet inputs	xI1… xI16 and xR1…xR16	4 Byte
7	easyNet outputs	xQ1…xQ8 and xS1…xS8	4 Byte
8	Analog inputs basic unit	IA1…IA4	8 Byte
9	Analog output, basic device	QA1	2 Byte
10	1 to 20 double word marker, MDx	MDx…MDx+n	4…80 Byte
11	easyNet Receive data	xRN1…xRN32	32 Byte
12	easyNet Send data	xSN1…xSN32	32 Byte
13	Diagnostic bits	ID1…ID16	2 Byte
14	Additional diagnostic bits for MFD-Titan	ID17…ID32	2 Byte
15	Programmable LEDs for MFD-Titan	LE1…LE3	1 Byte
16	Additional Bit marker for easy500/700	N1…N16	2 Byte
17	1 to 255 marker bytes	MBx…MBx+n	1…255 Byte
18	Digital inputs, basic device and assigned SWD inputs	I1…I128	16 Byte
19	Digital outputs, basic device and assigned SWD outputs	Q1…Q128	16 Byte

For further information take a look in the operating manual of the device.

The status value is returned in Little-Endian-Format (Intel format). The status value of the operand with the smallest operand number is in the Bit with the least value (LSB). The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code transmitted from the device is in the first Byte of the parameter value "data" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
object	Number of the object, therefore the process range
index	Node numbers for objects 6, 7, 11, 12 {1…8} Operand number for object 10 {1…96} or {1…128} Byte number for object 17 {1…384} or {1…512} No function otherwise
length	Object 10: Number of bytes to be read, divisible by 4 {4…80}. Object 17: Number of bytes to be read {1…255} No function otherwise.
data	Pointer on a memory area to save the status values

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

Caution:

The size of the data block that is to be transmitted with "data" must be sufficiently dimensioned that the memory limits are not exceeded.
Generally, only multiples of 4 bytes can be read in the case of object 10. Object 17 allows byte-wise access to the entire marker range, even if the device only knows byte access for the lower area. Use – whenever possible – the more efficient access method with object 10.

MC_Write_Object_Value

long MC_Write_Object_Value (tEasyComHandle handle, unsigned char net_id,
unsigned char object, unsigned short index, unsigned char length, unsigned char * data)

Write in the marker range.

The function writes values into marker range of the device.

The parameter "object" determines the type of the marker range:

4	Individual Bit marker Mx	1 Byte
10	1 to 80 marker bytes starting with double word marker MDx	1…80 Byte
16	Individual Bit marker Nx for easy500 and easy700	1 Byte
17	1 to 80 marker bytes	1…80 Byte

The status value is interpreted in Little-Endian-Format (Intel format). The status value of the Bit marker to be set must be entered in the least value Bit (LSB).

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code transmitted from the device is in the first Byte of the parameter value "data" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
object	Number of the object, therefore the process range
index	Operand number for objects 4, 10, 16 {1…16} or {1…96} or {1…128} Byte number for object 17 {1…384} or {1…512} No function otherwise.
length	For object types 10 and 17, specifies the number of bytes to be written {1…80}. No function otherwise.
data	Pointer on a memory area to save the status values

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or not registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

Caution:

The size of the data block that is to be transmitted with the parameter "data" must be sufficiently dimensioned that the memory limits are not exceeded.
The status of a written marker only remains the same if the device does not execute a program that has write access to this marker.
Object 17 allows byte-wise access to the entire marker range even if the device only knows byte access for the lower area. Use – whenever possible – the more efficient access method with object 10.

MC_Read_Channel_YearTimeSwitch

```
long MC_Read_Channel_YearTimeSwitch(tEasyComHandle handle,
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * on_year, unsigned char * on_month, unsigned char * on_day,
  unsigned char * off_year, unsigned char * off_month, unsigned char * off_day)
```

Read a parameter of the year time switch channel.

The function reads the parameter set of the selected channel from the given year time switch element. The parameter set consists of "calendar day", "month" and "year" each separated according to switch-on time-point and switch-off time-point.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "on_year" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
on_year	Pointer on a Byte variable to save the switch-on point - year number {0, 1…99}, 0 corresponds to "not occupied"
on_month	Pointer on a Byte variable to save the *switch-on point* - months {0, 1…12}, 0 corresponds to "not occupied"
on_day	Pointer on a Byte variable to save the *switch-on point* - calendar day {0, 1…31}, 0 corresponds to "not occupied"
off_year	Pointer on a Byte variable to save the *switch-off point* - year number {0, 1…99}, 0 corresponds to "not occupied"
off_month	Pointer on a Byte variable to save the *switch-off point* - months {0, 1…12}, 0 corresponds to "not occupied"
off_day	Pointer on a Byte variable to save the *switch-off point* – calendar day {0, 1…31}, 0 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No COM port is opened.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
16	The connection handle is not valid (any more)

Caution:

The easy800 and MFD8 generally only give zero values when the device is in STOP.

MC_Write_Channel_YearTimeSwitch

```
long MC_Write_Channel_YearTimeSwitch(tEasyComHandle handle,
  unsigned char net_id, unsigned char index, unsigned char channel        ,
  unsigned char * on_year, unsigned char * on_month, unsigned char * on_day,
  unsigned char * off_year, unsigned char * off_month, unsigned char * off_day)
```

Overwrite the parameter of a year time switch channel.

The function overwrites the parameter set of the selected channel from the given year time switch with values. The parameter set consists of "calendar day", "month" and "year" each separated according to switch-on time-point and switch-off time-point. The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "on_year" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
on_year	Pointer on a Byte variable to save the *switch-on point* - year number {0, 1…99}, 0 corresponds to "not occupied"
on_month	Pointer on a Byte variable to save the *switch-on point* - months {0, 1…12}, 0 corresponds to "not occupied"
on_day	Pointer on a Byte variable to save the *switch-on point* - calendar day {0, 1…31}, 0 corresponds to "not occupied"
off_year	Pointer on a Byte variable to save the *switch-off point* - year number {0, 1…99}, 0 corresponds to "not occupied"
off_month	Pointer on a Byte variable to save the *switch-off point* - months {0, 1…12}, 0 corresponds to "not occupied"
off_day	Pointer on a Byte variable to save the *switch-off point* – calendar day {0, 1…31}, 0 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No COM port is opened.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
13	The channel cannot be overwritten because there is no parameter there.
16	The connection handle is not valid (anymore)

Caution:

With easy800 and MFD-Titan, at least a part of the given channel must be occupied or the channel will be seen as not used.

MC_Read_Channel_7DayTimeSwitch

```
long MC_Read_Channel_7DayTimeSwitch(tEasyComHandle handle,
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * DY1, unsigned char * DY2,
  unsigned char * on_hour, unsigned char * on_minute,
  unsigned char * off_hour, unsigned char * off_minute)
```

Read a parameter of the 7-day time switch channel.

The function reads the parameter of the selected channel from the given 7-day time switch element.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "DY1" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
DY1	Pointer on a Byte variable to save the weekday parameter DY1 {0=SO…6=SA}, 255 corresponds to "not occupied"
DY2	Pointer on a Byte variable to save the weekday parameter DY2 {0=SO…6=SA}, 255 corresponds to "not occupied"
on_hour	Pointer on a Byte variable to save the switch-on hour {0…23}, 255 correspond to "not occupied"
on_minute	Pointer on a Byte variable to save the switch-on minute {0…59}, 255 correspond to "not occupied"
off_hour	Pointer on a Byte variable to save the switch-off hour {0…23}, 255 corresponds to "not occupied"
off_minute	Pointer on a Byte variable to save the switch-off minute {0…59}, 255 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
16	The connection handle is not valid (anymore)

Caution:

The easy800 and MFD8 generally only give zero values when the device is in STOP.

MC_Write_Channel_7DayTimeSwitch

```
long MC_Write_Channel_7DayTimeSwitch(tEasyComHandle handle,
  unsigned char net_id, unsigned char index, unsigned char channel,
  unsigned char * DY1, unsigned char * DY2,
  unsigned char * on_hour, unsigned char * on_minute,
  unsigned char * off_hour, unsigned char * off_minute)
```

Overwrite a parameter of the 7-day time switch channel.

The function overwrites the parameter set of the selected channel from the given 7-day time switch with new values. The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter "DY1" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
index	Gives the module number {1…4} for easy500/700 or {1…32} for easy800/MFD-Titan.
channel	Gives the channel {0…3}, 0 corresponds to channel A, 3 corresponds to channel D
DY1	Pointer on a Byte variable to save the weekday parameter DY1 {0=SU…6=SA}, 255 corresponds to "not occupied"
DY2	Pointer on a Byte variable to save the weekday parameter DY2 {0=SU…6=SA}, 255 corresponds to "not occupied"
on_hour	Pointer on a Byte variable to save the switch-on hour {0…23}, 255 correspond to "not occupied"
on_minute	Pointer on a Byte variable to save the switch-on minute {0…59}, 255 correspond to "not occupied"
off_hour	Pointer on a Byte variable to save the switch-off hour {0…23}, 255 corresponds to "not occupied"
off_minute	Pointer on a Byte variable to save the switch-off minute {0…59}, 255 corresponds to "not occupied"

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
12	The element is not available in the program.
13	The channel cannot be overwritten because there is no parameter there.
16	The connection handle is not valid (anymore)

Caution:

With easy800 and MFD-Titan, at least a part of the given channel must be occupied or the channel will be seen as not used.

MC_Unlock_Device

long MC_Unlock_Device (
*tEasyComHandle handle, unsigned char net_id, const char * szPassword, unsigned char * errorcode)*

Unlocks a device that is protected by a system password.

If a password is active on the device, the device can be unlocked with this function. For this purpose, the system password stored in the device must be passed in the "szPassword" parameter as a decimal value string. In the case of easy500 and easy700, the password consists of four decimal digits, whereas the password for easy800 and MFD-Titan consists of six decimal digits. The password is transmitted to the device in encrypted fashion and verified there. If the password does not correspond to the password stored in the device, an additional delay pause is used in order to hinder automatic testing of all combinations.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.
If a password is active on all easyNet nodes and communications with all of them are necessary, Unlock_Device must be called separately for each node.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
szPassword	Device system password as 0 terminated string {"0001"…"9999" resp. "000001"…"999999"}
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

Caution:

When the device is unlocked the previously locked menu items can called up.
After approx. 10 minutes inactivity the device automatically reactivates the password protection.

MC_Lock_Device

long MC_Lock_Device (tEasyComHandle handle, unsigned char net_id, unsigned char * errorcode)

Reactivates the system password of the device.

The password protection of a device can be reactivated when a device has a password and has been previously unlocked.

The setting parameter "net_id" identifies the target device. The value 0 indicates the device which is connected to the programming cable. The values 1 to 8 are selected by the station with the corresponding NET-ID.
If a password is active on all easyNet nodes and all nodes have been unlocked, the function must be called separately for each node.

If the function returns 2 the request from the device is rejected. The error code from the device is then in the parameter value "error code" (see Error codes of the devices)

Parameter:

handle	The connection handle returned when a connection is opened
net_id	Target device {0…8}; 0 for local programming interface
errorcode	Pointer on a Byte variable to the saving of error code sent from device.

Return Values:

0	Function call up successful.
1	A parameter contains an invalid value.
2	Device sends an error message.
3	Device does not respond.
6	Error message from Windows. Interrogate error code with GetLastSystemError.
7	No connection open.
8	The COM port is no longer present or no longer registered in the system (connection break)
9	The COM port is blocked by another process or is no longer registered in the system (connection break).
10	General communication error occurred (possible hardware failure)
11	Internal error of EASY_COM.dll
16	The connection handle is not valid (anymore)

5. Appendix

5.1 Error codes of the devices

If one of the interface functions returns 2 the request from the device is rejected. The error code transmitted from the device is in the first Byte (LSB) of the parameter "data" or "errorcode".

The meaning of the error codes returned from the device can be seen in the following tables. If the error code is not shown in the tables it could mean a faulty data transmission e.g. when other communication accesses take place parallel to the PC ("Terminal mode" with MFD-CP4)

Error codes from easy 500, easy700

Code	Meaning
2	Function/operation is not supported
3	Function/operation is not supported
4	Entity number is invalid
5	Parameter value is invalid
6	Write access on a element parameter that doesn't contain a constant
12	Function/operation only permissible in STOP
13	Display indication is not in normal position (input running)
16	Transmitted password does not agree with password in the device.
101	Password protection is active. Device must be unlocked.

Error codes from easy800, MFD-Titan

Code	Meaning
6	Function/operation is not supported
7	Function/operation is not supported
8	Invalid easyNet ID
9	easyNet ID larger than 0, but the device is not configured for easyNet
10	Entity number is invalid
18	Function/operation only permissible in STOP
20	Write access on a read only object
22	Given object size is incorrect
26	Requested element not available in the program
96	Password protection is active. Device must be unlocked.
97	Password protection is active. Device must be unlocked.
98	Password protection is active. Device must be unlocked.
99	Password protection is active. Device must be unlocked.
100	Password protection is active. Device must be unlocked.
101	Password protection is active. Device must be unlocked.
102	Password protection is active. Device must be unlocked.
103	Transmitted password does not agree with password in the device.
112	Device is not configured for easyNet operation. No communication possible via easyNet.
113	easyNet out of service
114	Communication defective. Device is communicating with another station.
115	easyNet out of service
116	Communication defective. Device is communicating with another station.
126	Communication defective. Device is communicating with another station.
133	Display indication is not in normal position (input running)
135	The action can not be performed because of a SWD configuration error or the wiring test is still active.
176	Setting of device clock or setting of summer time configuration rejected because of invalid parameter
225	Communication defective. Device is communicating with another station.
226	Communication defective. Device is communicating with another station.

5.2 Further information

User Manuals

MN05013003Z...	easy500/700
MN04902001Z...	easy800
MN05002001Z...	MFD-Titan
MN05013012Z...	EASY209-SE
MN05006004Z...	Data transfer between easy and IEC stations

Installation instructions

IL05013015Z...	easy500/700
IL05013012Z...	easy800
IL05013014Z...	MFD-Titan
IL05013021Z...	Programming cable EASY800-MO-CAB
IL05013019Z...	EASY209-SE

Application notes

USBDriverInst_x...	Programming cable EASY-USB-CAB / EASY800-USB-CAB
Online help of easySoft	
Online help of EASY209-SE configurator V2	
AN_EASYNET...	easyNet commissioning

Internet addresses

Support	http://www.moeller.net/easy
Product Catalog	http://ecat.moeller.net/
FTP server	ftp://ftp.moeller.net/EASY/
easy Forum	http://www.easy-forum.net/

5.3 Glossary

easyNet	A network for easy relays based on CAN. Up to 8 devices can be networked within a single line.
easyCOM	easy relay communication protocol for access via a programming device.
Station number NET-ID	Unique address of an easyNet station (1…8)
easyNet Master	The station with address 1. This station is always the 1st device on a line.
easyNet Slave	A station with an address between 2 and 8.
Option Remote-RUN	A slave operating mode. If this option is active, the slave follows the master's RUN/STOP state.
Option Remote-IO	A slave operating mode ("Remote I/O"). If this option is active, the slave does not execute any programs and allows the master to control the slave's outputs directly.
Process image	Data storage area that summarizes the states of all the inputs and outputs of a device.
Pointer	A variable that contains the address of a data memory