SERIAL COMMUNICATION PROTOCOL

Displays WAY Series

For further information please see the data sheet and manual at
https://www.waycon.de/produkte/anzeigen-displays/

GENERAL INFORMATION

The units use the LECOM protocol (DIN ISO 1745) which is a common standard with drive applications. Setting of the desired Baud rate and the data format as well as the pinout of the data connector are clearly described in the general operating instructions of the unit in use.

THE DEVICE ADDRESSES

The protocol supports unit addresses between 11 and 99 which can be set either by a keypad or a DIL switch or by serial setup. The unit address will be stored in the EEProm of the unit.
Adresses containing a „0“ must not be used, because they will be interpreted as a collective address for several units. The general address „00“ will talk to all units connected at this time. Addresses like „10“ or „20“ will talk to all units from 11 to 19 respectively from 21 to 29 etc.

Where the serial address of a unit should be unknown, the SCAN function of the correspondink motrona operator software will help you to find out. Ex factory, all motrona units are set to default address No.„11“

Please note that any response of a unit will be suppressed after access by the general address „00“ or a collective address like „20“.

SERIAL ACCESS CODES (REGISTER CODES)

For serial access to the registers within one unit the protocol uses either “Standard Addressing” or “Extended Addressing”, depending on the total number of registers to be accessed.
Please see the operating instructions of the corresponding unit to find out the code assignments and the mode of register addressing.
For reasons of clear differentiation, Extended register codes are always leaded by an exclamation mark.
Sub codes S1 and S2 must always be “0”, except other values are explicitly specified in
the manual of the unit.

REQUEST FOR DATA

To read out data from the operational registers (RAM), the subsequent request string must be used:

Standard register code

EOT	....	AD1	AD2	C1	C1	ENQ

EOT = Control character Ctrl D (Hex 04)

AD1 = Unit address, high byte **AD2** = Unit address, low byte

C1 = Register code, high byte **C2** = Register code, low byte

ENQ = Control character Ctrl E (Hex 05)

Extended register code

EOT	....	AD1	AD2	!	C1	C1	C3	C4	S1	S2	ENQ

EOT = Control character Ctrl D (Hex 04)

AD1 = Unit address, high byte **AD2** = Unit address, low byte

! = Exclamation mark (Hex 21)

C1 = Register code, high byte **C2** = Register code, low byte
S1 = Sub code, high byte **S2** = Sub code, low byte

ENQ = Control character Ctrl E (Hex 05)

For the valid register codes C1 and C2, please refer to the parameter list of the unit in use.

Example 1 (Standard addressing):
Read the integration speed register „Int-Time“ (register code 03) from a BY 150 synchro controller with device address „31“

	EOT	....	3	1	0	3	ENQ
hex:	04		33	31	30	33	05

Example 2 (Extended addressing):
Read the actual line speed (Code ! 081A) from a MC700 motion controller with Winding Firmware WR701 and the unit address “11”

	EOT	....	1	1	!	0	8	1	A	0	0	ENQ
hex:	04		31	31	21	30	38	31	41	30	30	05

REQUEST FOR DATA

Please note that figures 0-9 and characters A-F may be used for register addressing, where A-F are expressed by hexadecimalcodes 4.1 to 4.6.

With a correct unit address and a valid register code, the unit responds with one of the subsequent strings (depending on mode of addressing):

STX = Control character CRTL B (Hex 02)

ETX = Control character CTRL C (Hex 03)

XXX = Register data

BCC = Block check character

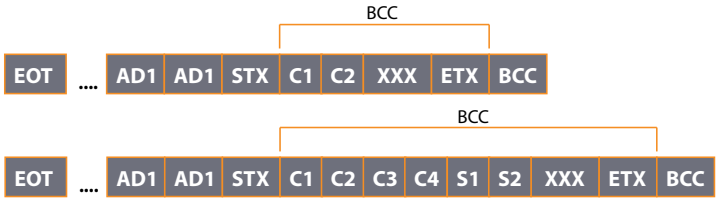
The total number of data characters „XXX" depends on the actual numeric value of the selected data register and may also be leaded by a minus sign with negative values. Leading zeros are always supressed and will not appear in the telegram. The block check character „BCC" is generated by an Exclusive-OR over all characters between "C1" resp. "!" and "ETX" (both included)

If the request string contains an invalid or unknown register code (C1 – C4 or S1-S2),the response of the unit will just be :

In case of other errors in the request string, the unit just responds by „NAK" (Hex 15)

WRITING TO REGISTERS

For modification of operational or other register data by a computer, one of the subsequent strings must be used, depending on addressing mode:



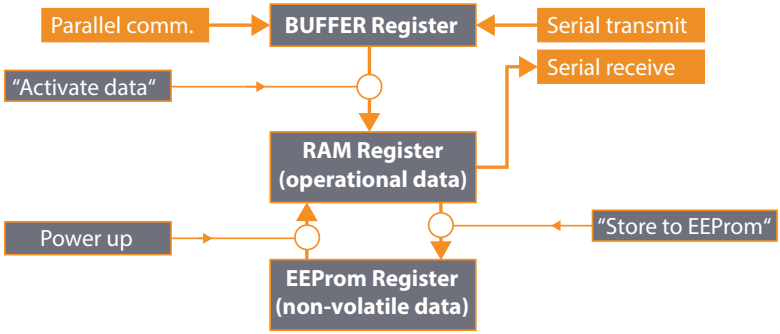
EOT = Control character Ctrl D (Hex 04)
AD1, AD2 = Unit address, High Byte first
! = Exclamation mark (Hex 21)
STX = Control character Ctrl B (Hex 02)
C1, C2, C3, C4 = Register code, High byte first
S1, S2 = Subcode, High byte first
XXX = New register data (ASCII Code)
ETX = Control character Ctrl (Hex 03)
BCC = Block check character

WRITING TO REGISTERS

The data string „XXX“ can have any number of characters and may also contain leading zeros or a negative sign. The block check character „BCC“ represents again the Exclusive-OR function of all characters between „C1“ resp. “!” and „ETX“. Upon a correct transmission of above protocol, the unit will respond by „ACK“ and in all other cases by „NAK“. With the units, all data from a serial transmission will first be latched in a buffer register and will not affect the current operation, until the command „Activate Data“ has been transmitted. This procedure enables the user to prepare a complete new set of parameters in the background of the unit, with no effect to the production in progress, and to activate all at the same time by one single command.

Remarks to the Register Organisation:
Any readout of parameters will read the current operational data from the RAM. Parameters, that have been written to the buffer memory only, but have not been activated yet, are not readable by serial request.

Upon power up, the unit automatically transfers the EEProm data to the RAM register. Previous serial register modifications will be lost, unless these data have been activated first and then stored to the EEProm.



TRANSMISSION OF CONTROL COMMANDS

All control commands are using the same transmission protocol like shown in section 4. However, the data string „XXXXX“ uses only one character which is „**1**“ to switch the function on and which is „**0**“ to switch the function off again. Some of the serial commands will automatically reset to zero upon execution of the corresponding commands (i.e. „Activate data“ or „Restore to EEProm“). Others will need to be set and also to be reset by serial command (i.e. Reset, Start/Stop, Trim etc.).

For applicable command codes, please see the operating instructions of the unit you use.

All serial transmission of a control command will have the same result as when you set the appropriate hardware input to „High“.

TRANSMISSION OF CONTROL COMMANDS

Important remark:
There is a logical „OR" condition between all hardware control inputs and their corresponding serial command flags. For this reason, it is essential to have the hardware input and the serial command set to OFF at the same time in order to switch a command OFF !

If, i.e. by serial communication, the Reset flag has been set to „1", the unit will be in it's Reset state, independant of what the logical level at the hardware reset input is. Only with the serial reset flag at „0" and the reset input at „Low", the unit will be in it's normal operative state ! Upon power up, all serial command flags will automatically be reset to „0" .

PRACTICAL EXAMPLE FOR COMMUNICATION

The following example describes how to transmit a new "Factor1" setting to a BY125 synchro controller. BY125 uses standard addressing for register access. Where you use a unit with extended addressing, the subsequent example is fully valid,too, except of the extended version of address codes and sub codes.

The example uses a BY125 controller with serial **unit address "11"**. According to the manual, register "Factor1" uses the serial **access code "00".** The example shows how to set the register to a **value of "0.9873"**

1. First, we transmit the data string consisting of totally 13 ASCII characters:

No.	Expression	ASC II	Hex	Binary code Hi------ -----Lo	Comment
01	EOT	EOT	0 4	0000 0100	Control character Initialisation
02	AD1	1	3 1	0011 0001	Address, high byte
03	AD2	1	3 1	0011 0001	Address, low byte
04	STX	STX	0 2	0000 0010	Control character
05	C1	0	3 0	0011 0000	Register code high byte
06	C2	0	3 0	0011 0000	Register code low byte
07	X (Data)	0	3 0	0011 0000	Factor, highest digit
08	X (Data)	9	3 9	0011 1001	
09	X (Data)	8	3 8	0011 1000	
10	X (Data)	7	3 7	0011 0111	
11	X (Data)	3	3 3	0011 0011	Factor, lowest digit
12	ETX	ETX	0 3	0000 0011	Control character
13	BCC	6	3 6	0011 0110	Block check character

PRACTICAL EXAMPLE FOR COMMUNICATION

Characters with gray background are used to form the block check character by an **exclusive- OR** function. Consider each of the **8 columns in the binary code field** only gray rows).In the high bit column we find all zeros, therefore the exclusive OR is 0 and the high bit of the block check character must be zero at this position.

In the low bit column we find 0-0-0-1-0-1-1-1 (from up to down) and the exclusive OR is also „0". Therefore the low bit of the block check character must be „0" again at this position. In the column left of the lowest bit, the X-OR function results in a „1" etc.

In general we can say: when in one column we find an even number of „1", the block check bit must be „0" at this position. When we find an odd number of „1", the block check bit must be „1" at this position.

So we find with our example, that the 8 Bits of the block check character, read as a row, are **0-0-1-1-0-1-1-0** which equals to the Hexadecimal code "36" or to the ASCII character of „6".

2. Wait for acknowledgement
After correct transmission, the unit will acknowledge by responding "ACK" (Hex "06" or binary 0000 0110). Where the unit instead should respond "NAK" (Hex "15"), the transmission was rejected due to a fault like wrong BCC or incorrect sequence of characters etc.

Where the unit does not respond at all, this indicates an uncomplete transmission string or wrong basic serial settings like Baud rate, Data format etc.

3. Transmit more parameters
We are free now to transmit any number of more parameters if necessary, without any effect to the function of the machine.

4. **Activate data**
After successful transmission of all parameters we must activate the new settings, to make them effective to the operation of the machine. With BY125 units, we need to write a "1" into the register with code "67" (C1 = 6 and C2 = 7).

No.	Expression	ASCII	Hex	Binary code Hi----------Lo	Comment
01	EOT	EOT	0 4	0000 0100	Control character Initialisation
02	AD1	1	3 1	0011 0001	Address, high byte
03	AD2	1	3 1	0011 0001	Address, low byte
04	STX	STX	0 2	0000 0010	Control character
05	C1	6	3 6	0011 0110	Register code high byte
06	C2	7	3 7	0011 0111	Register code low byte
07	X (Data)	1	3 1	0011 0001	Activate command „ON"
08	ETX	ETX	0 3	0000 0011	Control character
09	BCC	3	3 3	0011 0011	Block check character

PRACTICAL EXAMPLE FOR COMMUNICATION

5. Save data to EEProm (option only)
Without sending this command, the controller will use all data which have been transmitted and activated, until the unit will be powered down. At next power up, the unit however will restart with data loaded from it's EEprom. The serial register code for the „Store" command is "68" (C1 = 6, C2 = 8), and again the data string XXX must just be one digit with the value of "1".

Hint:
The life time of EEProm memory chips is limited to a total number of about 100 000 storage cycles. After this, stored data may get lost.