BINDER Interface Technical Specifications

BINDER GmbH

Address	Post office box 102 78502 Tuttlingen, Germany
Internet	http://www.binder-world.com
E-mail	info@binder-world.com
Service e-mail	service@binder-world.com

Issue 10/2017 Art. No. 7001-0242

Content

1. Common description

The BINDER interface uses the MOD Bus protocol (Chapter 2), even when TCP/IP transmission is used. The MOD Bus protocol is tunneled through the TCP/IP (Chapter 3). The result is that a programmer has to implement the MOD Bus protocol and the TCP/IP tunnel.

This document describes both actions. Understanding and implementing the MOD Bus is the harder of both jobs.

1.1 Developer Support

This document is available free of charge. It enables you to develop your own software. This is the complete information regarding interfaces.

Please keep in mind that BINDER GmbH does not offer any support further than this document.

The information is provided as it is, without warranty of any kind. It could contain wrong information.

1.2 Good tools

On http://www.microsoft.com/technet/sysinternals/default.mspx you can find some really good freeware tools for software development.

Portmon 3.x is the one you need, if you want to see the data flow between BINDER controllers and the RS 232 (RS422). Don’t forget to switch to hex mode for viewing the data.

On http://www.binder-world.com/ you can download the APT-COM 3 Demo Version for free. The Demo is restricted in saving data and configuration but not in time. If you activate Portmon 3.x while running APT-COM 3 Demo you can see the data being sent and received. The data are not coded in any kind.

With APT-COM 3 Demo you can easily check the hardware.

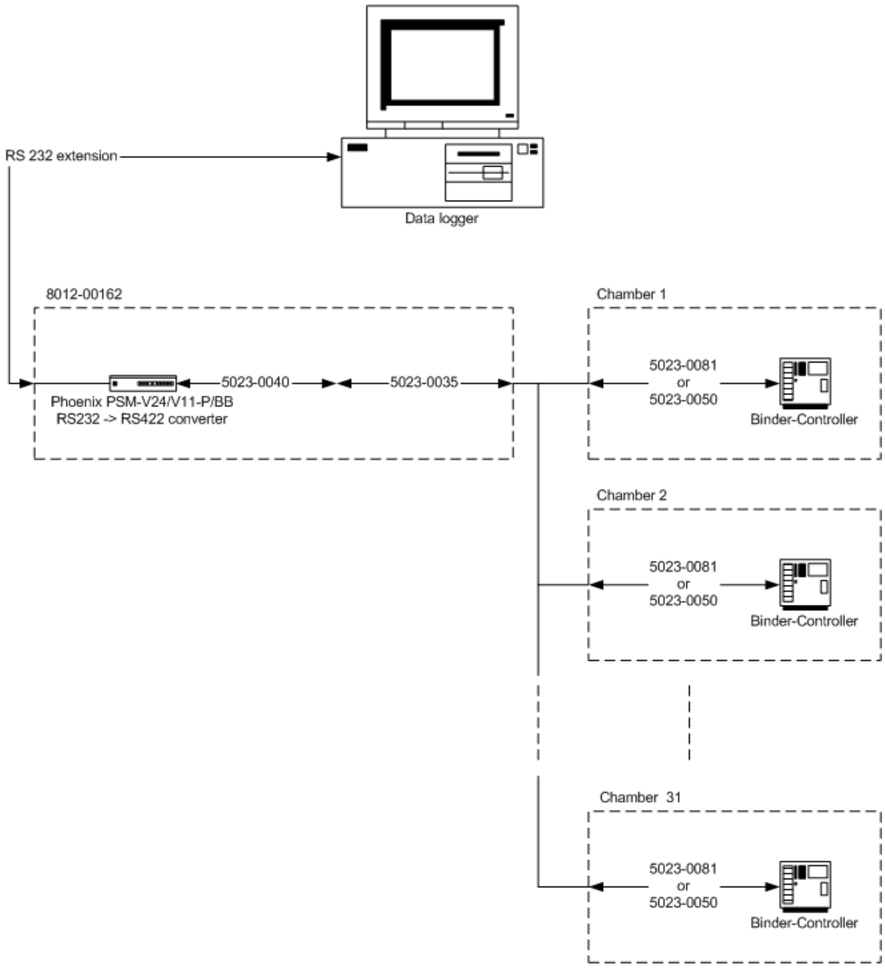
On http://www.wireshark.org/ you can download a very good Freeware tool. Wireshark is used by network professionals around the world for troubleshooting, analysis, software and protocol development, and education. It has all of the standard features you would expect in a protocol analyzer, and several features not seen in any other product.

2. BINDER MOD Bus common description

This description applies to the controllers R3, R3.1, DIC1000, DIC1001, RD3, dTron and MB1. All controllers come with a 4-wire RS 422 interface. The protocol is an adapted MOD Bus.

2.1 Physical

The RS422 of the controller(s) and the data logger (PC) is galvanically isolated from signals and supply and supports up to 31 Slaves (Chambers) and one Master. The system uses a 4 wire full duplex bus which is used only in half duplex mode. This means there is only a request <u>or</u> a response at one time.



2.2 Master-slave principle

The communication between a PC (master) and an instrument (slave) using MOD Bus takes place according to the master-slave principle in the form of data request/instruction - response. The master controls the data exchange; the slaves only have a response function. They are identified by their instrument addresses. A maximum of 255 addresses can be accessed.

2.3 Transmission mode (RTU)

The transmission mode used is the RTU mode (Remote Terminal Unit). Data is transmitted in binary form (hexadecimal) with 8 bits, 16 bits for integers, and 32 bits for floating values. The LSB (least significant bit) is transmitted first. The ASCII operating mode is not supported.

Data format The data format describes the arrangement of a byte transmitted.

The data format is as follows:

Baud rate:	9600 bits/sec
Data word:	8 bit
Parity bit:	none
Stop bit:	1

2.4 Instrument address

The address of the slaves can be set between 1 and 255. Address 0 is reserved.

Possibilities of data exchange

Query Data request/instruction from the master to a slave via the appropriate address. The slave accessed then responds.

Broadcast Instruction from the master to all slaves via address 0. The slaves in the system do not respond. A data request is not appropriate in this case.

For example, a certain set point can be transmitted to all slaves. The correct acceptance of the value by the slaves should in that case be checked by a subsequent read-out of the set point.

2.5 Timing of the communication

Start and end of a data block are identified by transmission pauses. The maximum permitted interval between two consecutive characters is three times the time for transmitting a character.

The character transmission time (time for transmitting a character) depends on the baud rate and on the data format used.

Character transmission time [us] = 937,5

A maximum of 31 slaves can be accessed via the RS422 interface.

Data request from master

Transmission time = n characters * 1000 * x bits/baud rate

Marker for data request end

3 characters * 1000 * x bits/baud rate = 2.813 ms

Processing the data request by the slave (200 ms max.)

Response of the slave

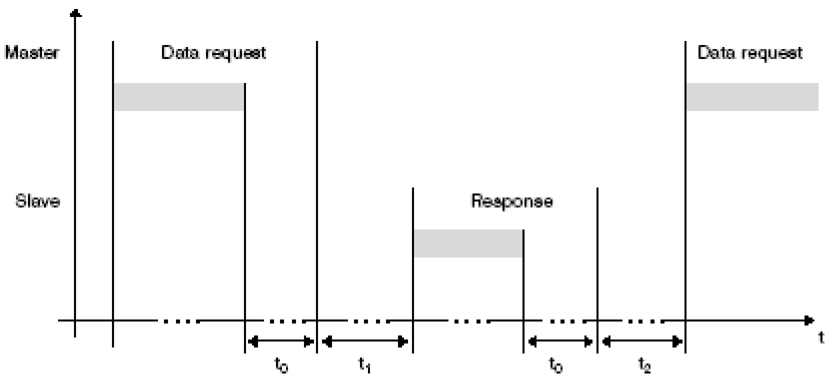
Transmission time = n characters * 1000 * x bits/baud rate

Marker for response end

3 characters * 1000 * x bits/baud rate = 2.813 ms

2.5.1 Timing scheme

A data request runs according to the following timing scheme:



t0 = end marker = 3 characters = 2.813 ms

t1 = 5 ms … 250 ms

t2 >= 10 ms

2.5.2 Communication during the internal processing time of the slave

The master must not make any data requests during the internal processing time of the slave. Any data requests during this period are ignored by the slave.

2.5.3 Communication during the response time of the slave

The master must not make any data requests during the response time of the slave. Any data requests during this period cause all data currently on the bus to become invalid.

2.6 Arrangement of the data blocks

All data blocks have the same structure:

Data structure

slave address	function code	data field	checksum CRC16
1 byte	1 byte	x byte(s)	2 bytes

Each data block consists of four fields:

Slave address instrument address of a particular slave

Function code function selection (read, write, bit, word)

Data field

-contains the information:

-bit address (word address)

-bit number (word number)

-bit value (word value)

Checksum recognition of transmission errors

2.7 Error treatment

Error codes There are five error codes:

1. invalid function
2. invalid parameter address
3. parameter value outside range of values
4. slave not ready
5. write access to parameter denied

Response in case of error

slave address	function XX OR 80h	error code	checksum CRC16
1 byte	1 byte	1 byte	2 bytes

The function code is linked by OR with 0x80, i.e. the MSB (most significant bit) is set to 1.

Example

Data request:

01	02	00	00	00	00	CRC16

Response:

01	82	01	CRC16

Special cases The slave does not respond to the following errors:

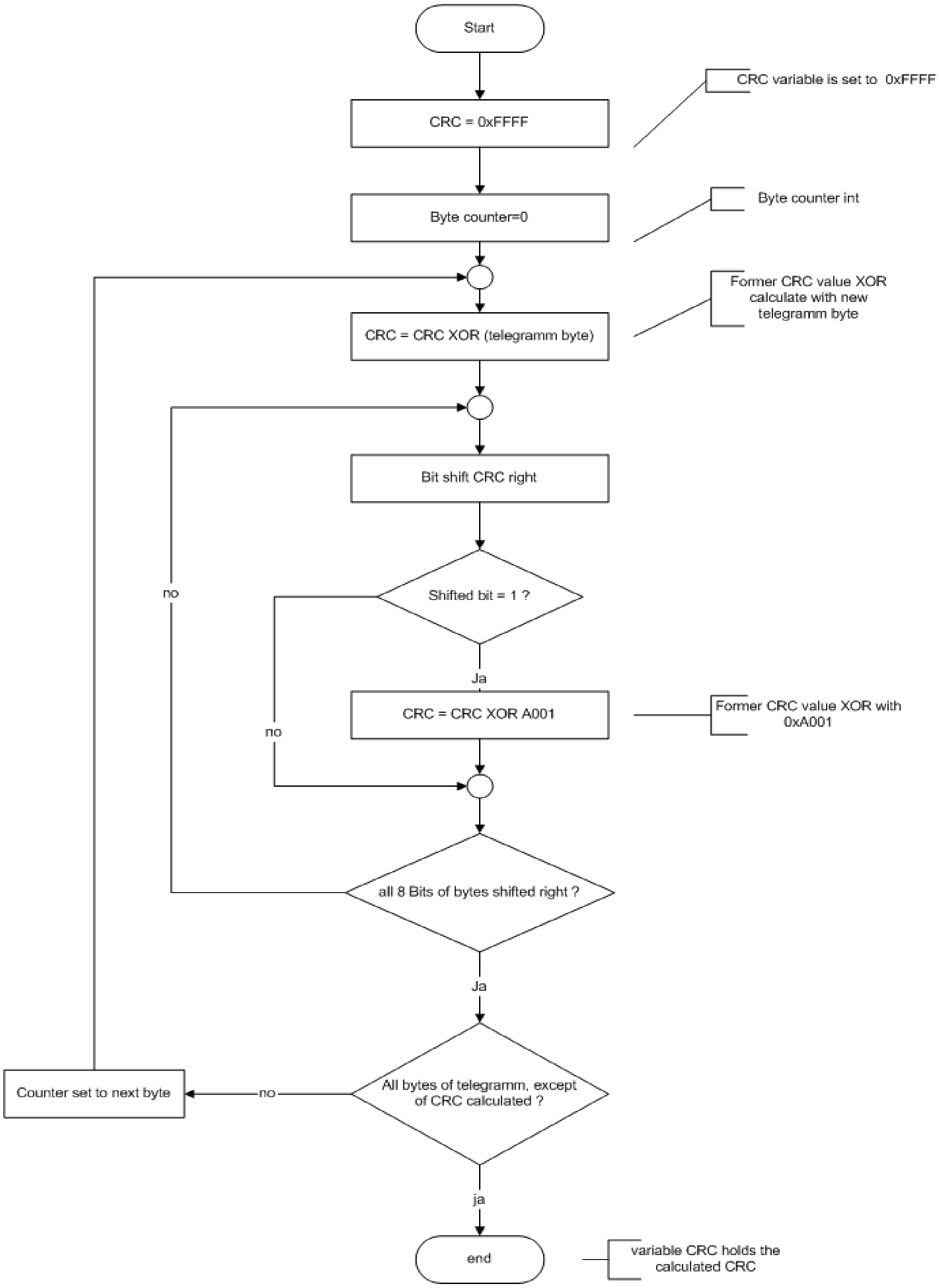
- The checksum (CRC16) is incorrect
- The instruction of the master is incomplete or over-defined
- The number of the words or bits to be read is zero

Error code 4 (slave not ready) is not implemented in the controller since the controller always responds within 250 ms to a valid data request.

2.8 Checksum (CRC16)

The checksum (CRC16) serves to recognize transmission errors. If an error is identified during processing, the appropriate instrument does not respond.

Calculation scheme



Example 1

Data request: reading two words from address 1 (CRC16 = 0x0E97)

14	03	00	01	00	02	97	0E

Response: (CRC16 = 0x953E)

14	03	04	03	E8	01	F4	3E	95
			Word 1		Word 2			

Example 2

Instruction: Set bit on address 24 (CRC16 = 0xF80E)

14	05	00	18	FF	00	0E	F8

Response (as instruction):

14	05	00	18	FF	00	0E	F8

2.9 Functions

The following functions are available to the controller:

Function number	**Function**
0x03 or 0x04	Reading n words (max. 80 words)
0x06	Writing one word
0x10	Writing n words (max. 80 words)

2.9.1 Reading more than one words

This function reads *n* words from a defined address.

Data request

Slave address	Function 0x03 or 0x04	Address first word	Number of words	Checksum CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

Response

Slave address	Function 0x03 or 0x04	Number of bytes read	Word value(s)	Checksum CRC16
1 byte	1 byte	1 byte	x byte(s)	2 bytes

Example:

Reading the four floats form base address 0x0037

Word address= 0x0037

Data request:

14	03	00	37	00	08	CRC16

Response:

14	03	10	1999	4348	4CCC	4348	2666	4396	F333	43CA	CRC16
			Signal 1 200.1		Signal 2 200.3		Signal 3 300.3		Signal 4 405.9		

2.9.2 Writing one word

In the "writing word" function the data blocks for instruction and response are identical.

Instruction

Slave address	Function 0x06	Word address	Word value	Checksum CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

Response

Slave address	Function 0x06	Word address	Word value	Checksum CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

Example Write base address 0x0168 value 100

Instruction:

14	06	01	68	00	64	CRC16

Response (as instruction):

14	06	01	68	00	64	CRC16

2.9.3 Writing more than one words

Instruction

Slave address	Function 0x10	Address first word	Number of words	Number of bytes	Word value(s)	Checksum CRC16
1 byte	1 byte	2 bytes	2 bytes	1 byte	x byte(s)	2 bytes

Response

Slave address	Function 0x10	Address first word	Number of words	Checksum CRC16
1 byte	1 byte	2 bytes	2 bytes	2 bytes

Example:

Write value = 0.66 to base address 0x010F

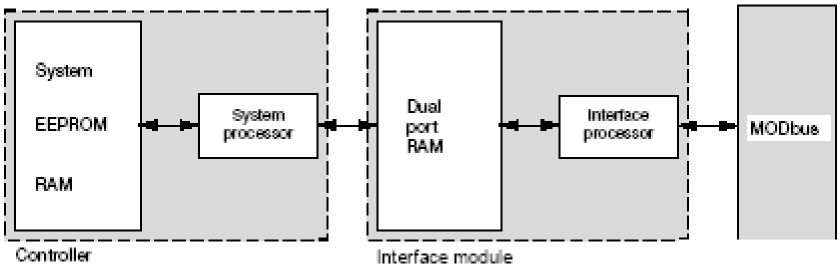
Instruction:

14	10	01	0F	00	02	04	F5	C2	3F	28	CRC16

Response:

14	10	01	0F	00	02	CRC16

2.10 Data flow (DIC1000 / DIC 1001 only)



For data transmission to the MODBus, the system processor places the process values in a dual port RAM. Not all the system variables present in the controller are updated cyclically in the dual port RAM. The dual port RAM is divided into two areas:

System variables

These variables can be read and written directly by the MODBus driver (cyclic data). These data are updated cyclically in the dual port RAM (within the sampling time).

Data after data request

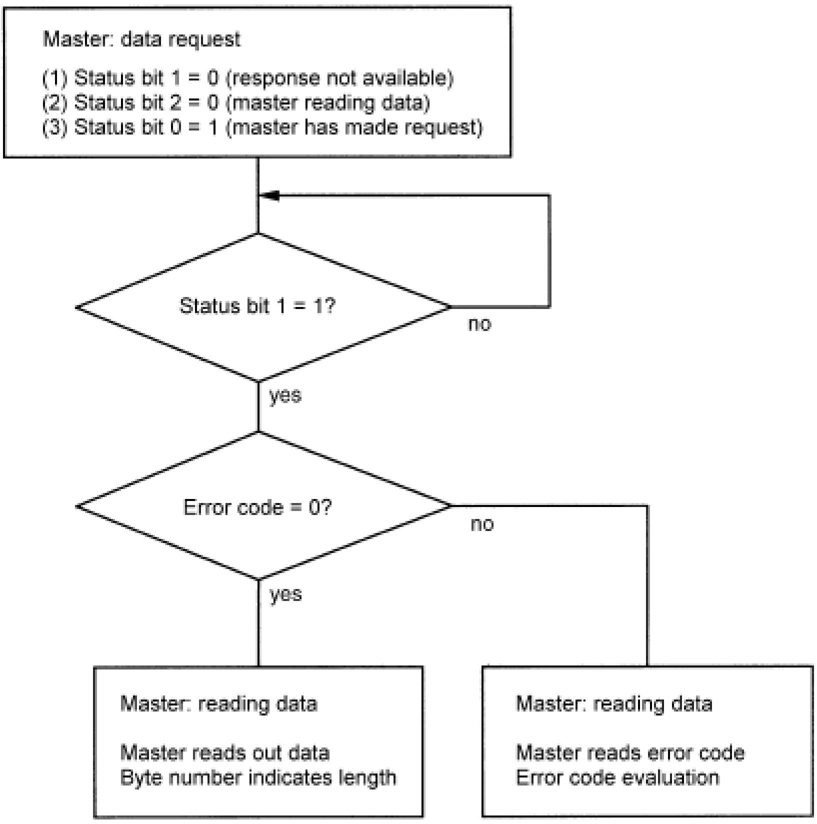
This area is not updated cyclically by the system processor (non-cyclic data). Variables in this data area must be requested by the MODBus driver. They are available only after processing by the system processor.

The dual port RAM is erased after a reset. This is followed by the system processor updating the cyclic data. The area of non-cyclic data has to be updated by the user.

The length information for char data types as given below always covers the character chain length including the chain end character /0.

Every alteration of a process value which is stored in EEPROM results in updating of the data in the EEPROM. Please note that the EEPROM can be re-written about 10000 times.

2.10.1 Receiving data from the controller (non-cyclic data only)



Example

Reading EEPROM set point 1

Step 1:

The data structure parameter set 1 of controller 1 is requested.

Set status bit 0 = 1, status bit 1 = 0, and status bit 2 = 0

MODBus instruction: write 1 word

01	06	09	F6	00	01	CRC16

Response:

01	06	09	F6	00	01	CRC16

Step 2:

Cyclic polling whether the corresponding data structure is available.

Status bit 1 read

MODBus instruction: read 1 word

01	03	09	F6	00	01	CRC16

Response:

01	03	02	00	00	CRC16

Status bit 1 = 0 (data structure is not yet available)

01	03	02	00	02	CRC16

Status bit 1 = 1 (data structure is available)

Step 3:

Read error code of the structure requested

MODBus instruction: read 1 word

01	03	09	F6	00	01	CRC16

Response:

01	03	02	00	00	CRC16

Step 4:

Read out set point value

MODBus instruction: (float → 2 words)

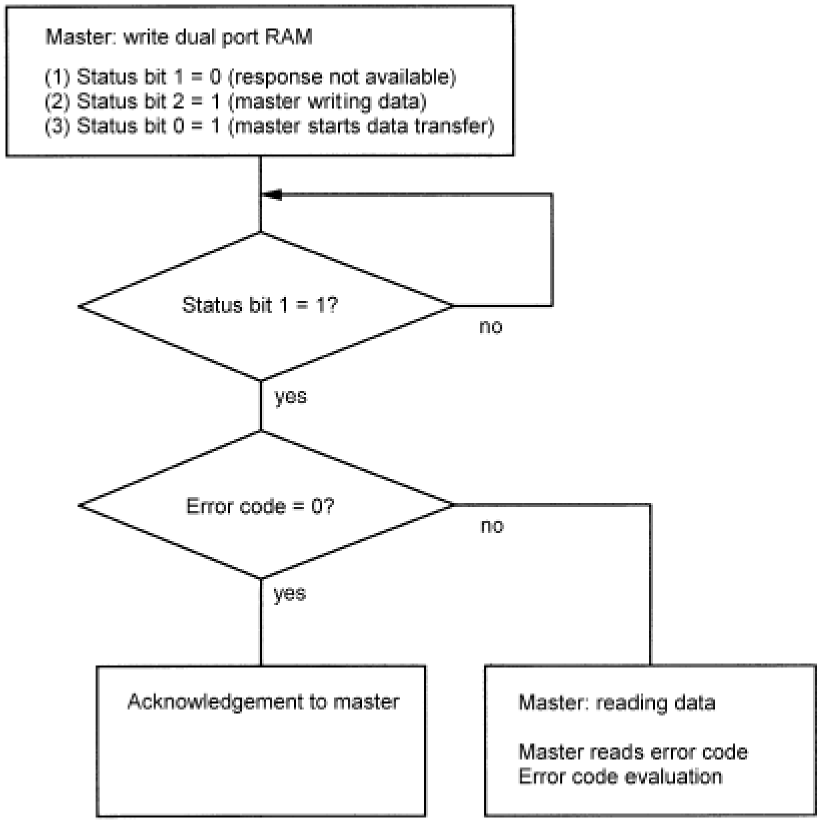
01	03	09	F8	00	02	CRC16

Response:

01	03	04	00	00	41	F0	CRC16

→ Value = 30

2.10.2 Transmitting data to the controller (non-cyclic data only)



Example:

Writing EEPROM set point 1

Step 1: The data structure set points is requested.

Set status: bit 0=1, bit 1 = 0, bit 2 = 0

MODBus instruction: write 1 word

01 06 09 F6 00 01 CRC16

Response:

01 06 09 F6 00 01 CRC16

Step 2: Cyclic polling whether the appropriate data structure is available.

Status bit 1 read

MODBus instruction: read 1 word

01 03 09 F6 00 01 CRC16

Response:

01 03 02 00 00 CRC16

Status bit 1 = 0 (data structure is not yet available)

01 03 02 00 02 CRC16

Status bit 1 = 1 (data structure is available)

Step 3: Read error code of the structure requested

MODBus instruction: read 1 word

01 03 09 F7 00 01 CRC16

Response:

01 03 02 00 00 CRC16

Step 4: Write set point 2 of controller 1

(Set point 20.32 is IEEE format 41 A2 8F 5C)

MODBus instruction: write 2 words

01	10	09	F8	00	02	04	8F	5C	41	A2	CRC16

Response:

01	06	09	F8	00	02	CRC16

Step 5: The data structure set points is transmitted

Set status: bit 0 = 1, bit 1 = 0, bit 2 = 1

MODBus instruction: write 1 word

01	06	09	F6	00	05	CRC16

Response:

01	06	09	F6	00	05	CRC16

Step 6: Cyclic polling whether the corresponding data structure has been transmitted.

Status bit 1 read

MODBus instruction: read 1 word

01	03	09	F6	00	01	CRC16

Response:

01	03	02	00	00	CRC16

Status bit 1 = 0 (data structure not yet transmitted)

01	03	02	00	02	CRC16

Status bit 1 = 1 (data structure has been transmitted)

Step 7: Read error code of transmitted structure

MODBus instruction: read 1 word

01	03	09	F7	00	01	CRC16

Response:

01	03	02	00	00	CRC16

2.11 Base address tables

2.11.1 How to read address tables

All process values (variables) together with their addresses, the data type and the access mode are described below.

References are as follows:

R/O	access reading only
R/W	access reading and writing
char	ASCII character (8 bits)
byte	byte (8 bits)
int	integer (16 bits)
char xx	character chain of length xx;
	xx = length including chain and character /0
bit x	bit No. x
long	long integer (4 byte)
int10	int10 Value / 10 = Value

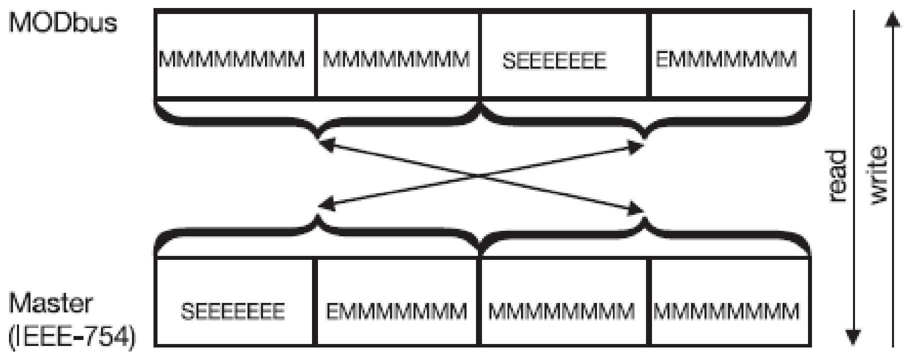
float **floating value (4 bytes)**

The explanations below apply on condition that the master operates in the IEEE-754 format. Before transmitting a value the byte sequence has to be changed so that it corresponds to the presentation for MODBus (see diagram).

M- 23 bit normalized mantissa

E - exponent (complement base 2)

S - sign bit; 1 = negative; 0 = positive



Example:

Transmitting the decimal value 550:

MODBus: 0x80, 0x00, 0x44, 0x09

2.11.2 Base address table for R3

Value name	Base Address (hex)	Data type	Access
Process value	0000	float	R/O
Current Set Point	0002	float	R/O
Set point	0004	float	R/W
Timer (hhmm)	0006	float	R/W

2.11.3 Base address table for R3.1 / R3.2

Value name	Base Address (hex)	Data type	Access
Process value	0000	float	R/O
Current Set point	0002	float	R/O
Set point	0004	float	R/W
Process value / set point fan	0006	float	R/W
Process value / set point timer	0008	float	R/W

2.11.4 Base address table for DIC1000

Value name	Base Address (hex)	Data type	Access	Non-cyclic data *
Process value 1	0047	float	R/O	no
Set point 1	0048	float	R/O	no
Process value 2	0058	float	R/O	no
Set point 2	005C	float	R/O	no
Set point status of data structure	0484	int	R/W	yes
Set point error code	0485	int	R/W	yes
Set point 1	0486	float	R/W	yes
Set point 2	0488	float	R/W	yes

* For non-cyclic data please read chapter 2.10

2.11.5 Base address table for DIC1001

Value name	Base address (hex)	Data type	Access	Non-cyclic data *	Note
Process value 1	0047	float	R/O	no	
Set point 1	0048	float	R/O	no	
Process value 2	0058	float	R/O	no	
Set point 2	005C	float	R/O	no	
Track	007C	int	R/O	no	Bit 0–7 = Track 0-7
Controller mode	0039	int	R/O	no	Bit 3 = 1 → Basic Bit 4 = 1 → Manual Bit 13 = 1 → Auto
Prog mode status of data structure	01B5	int	R/W		
Prog mode error code	01B6	int	R/W		
Prog mode on	01B7	int	R/W	yes	Write value = 0x0001
Prog number	01B8	int	R/W	yes	Write value = 0 - 19
Prog start	01BB	long	R/W	yes	Write value = 0xFFFF FFFF
Set point status of data structure	0960	int	R/W		
Set point error code	0961	int	R/W		
Basic mode on	0962	int	W	yes	Write value = 0x0000
Manual mode on	0962	int	W	yes	- Write value = 0x0001 - after changing in manual mode, switch on controller and comparator
Set point 1 manual mode	0963	float	W	yes	
Set point 2 manual mode	0965	float	W	yes	
Track write manual mode	0967	int	W	yes	Bit 0–7 = Track 0-7
Start controller manual mode	096A	int	W	yes	Write value = 0x0003
Start limit comparator	096B	int	W		Write value = 0x00FF
Set point status of data structure	0980	int	R/W		
Set point error code	0981	int	R/W		
Set point 1 basic mode	0985	float	W	yes	
Set point 2 basic mode	0987	float	W	yes	

* For non-cyclic data please read chapter 4

2.11.6 Base address table for MB1 (without program option)

Value name	Base address (hex)	Data type	Access	Note
Process value 1	11A9	float	R/O	Temperature
Process value 1 (720)	1017	float	R/O	For KB(W)F E2 720 (prog) only
Process value 4	11CD	float	R/O	Humidity
Set point 1	1077	float	R/O	Temperature
Set point 2	1079	float	R/O	Humidity
Set point 1	1A69	float	R/W	Temperature
Set point 2	1A6D	float	R/W	Humidity

2.11.7 Base address table for MB1 (with program option)

Value name	Base address (hex)	Data type	Access	Note
Process value 1	11A9	float	R/O	Temperature
Process value 1 (720)	1017	float	R/O	For KB(W)F E2 720 (prog) only
Process value 4	11CD	float	R/O	Humidity
Set point 1	1077	float	R/O	Temperature
Set point 2	1079	float	R/O	Humidity
Set point 1 basic	156F	float	R/W	Temperature
Set point 2 basic	1571	float	R/W	Humidity
Set point 1 manual	1581	float	R/W	Temperature
Set point 2 manual	1583	float	R/W	Humidity
Track read	1081	int	R/O	Bit 0–7 = Track 0-7
Track manual	158B	int	R/W	Bit 0–7 = Track 0-7
Mode read	1A22	int	R/W	Bit 10=1 → auto Bit 11=1 → manual Bit 12=1 → basic
Mode basic on	1A22	int	R/W	Write value = 0x1000*
Mode manual on	1A22	int	R/W	Write value = 0x0800*
Select prog no.	1A23	int	R/W	Write value = 0 –19
Mode auto on	1A22	int	R/W	Write value = 0x0400*

* Note:

Steps to change mode

1. Read mode value
2. Set mode bits of the read value to zero
3. Set mode bit of the mode you want to One and write it to the controller

2.11.8 Base address table for MB1 (CB and CB O2 only)

Value name	Base address (hex)	Data type	Access	Note
Process value 1	11A9	float	R/O	Temperature
Process value 2	1045	float	R/O	CO_2 %
Process value 3	105C	float	R/O	O_2 %
Set point 1	1077	float	R/O	Temperature
Set point 2	1079	float	R/O	CO_2 %
Set point 3	1A1B	float	R/W	O_2 %
Set point 1	156F	float	R/W	Temperature
Set point 2	1571	float	R/W	CO_2 %

2.11.9 Base address table for RD3

Value name	Base address (hex)	Data type	Access	Note
Process value 1	0040	float	R/O	Temperature
Process value 2	0042	float	R/O	Light
Process value 3	0050	float	R/O	Fan speed
Set point 1	0072	float	R/O	Temperature
Set point 2	004E	float	R/O	Light
Set point 3	0050	float	R/W	Fan speed
Track read	0052	int	R/O	Bit 0–7 = Track 0-7
Set point 1 manual	0343	float	R/W	Temperature
Set point 2 manual	0345	float	R/W	Light
Set point 3 manual	0347	float	R/W	Fan speed
Track write	034B	int	R/W	Bit 0–7 = Track 0-7
Mode read	0046	int	R/W	0x0020 → auto (Bit 9) 0x0008 → manual (Bit 7)
Mode manual on	0046	int	R/W	Write value = 0x0008*
Select prog no.	0065	int	R/W	Write value = 0 or 1
Mode auto on	0046	int	R/W	Write value = 0x0020*

* Note:

Steps to change mode

1. Read mode value
2. Set mode bits of the read value to zero
3. Set mode bit of the mode you want to One and write it to the controller

2.11.10 Base address table for dTron8/16 (AB01 and TM01 only)

Value name	Base address (hex)	Data type	Access	Note
AB01 Alarm On/ Off	0026	float	R/W	100 = Alarm On 0 = Alarm Off
Process value TM01	0000	float	R/O	

Note: The slave address of AB01 is factory fixed to 30.

2.11.11 Base address table for dTron 308 (AB01 E2 and TM01 E2 only)

Value name	Base address (hex)	Data type	Access	Note
AB01 Alarm On/ Off	3100	float	R/W	100 = Alarm On 0 = Alarm Off
Process value TM01	0043	float	R/O	

Note: The slave address of AB01 is factory fixed to 30.

2.11.12 Base address table for RP1

Value name	Base address (hex)	Data type	Access	Note
Process value 1	0245	int10	R/O	Temperature
Process value 2	0247	int10	R/O	CO_2 %
Set point 1	0192	int10	R/W	Temperature
Set point 2	0191	int10	R/W	CO_2 %

2.11.13 Base address table for R4

Value name	Base address (hex)	Data type	Access	Note
Process value 1	8962	Float	RO	Temperature
Process value 2	89C6	Float	RO	Object temperature (optional)
Set point 1	8A2E	Float	RW	Temperature
Process value fan	8BDE	Float	RO	Fan speed *
Set point fan	8BE0	Float	RW	Fan speed *

* Note: Units with adjustable fan speed only

2.11.14 Base address table for MB2

Value name	Base address (hex)	Data type	Access	Note
Process value 1	1004	float	R/O	Temperature
Process value 4	100A	float	R/O	Humidity
Set point 1	10B2	float	R/O	Temperature
Set point 2	10B4	float	R/O	Humidity
Set point 3	10B6	float	R/O	Fan speed
Set point 1 manual	114C	float	R/W	Temperature
Set point 2 manual	114E	float	R/W	Humidity
Set point 3 manual	1150	float	R/W	Fan speed
Track read	1292	int	R/O	Bit 0-15 = Track 0-15
Track manual	1158	int	R/W	Bit 0-15 = Track 0-15
Program number	1147	int	R/W	0..24 = Time program 1..25 25..29 = Week program 26..30
Start section number	1148	int	R/W	
Start program	1149	bool	R/W	Write value = 1
Stop program	114A	bool	R/W	Write value = 1
Pause program	114B	bool	R/W	Write value = 1

3. BINDER TCP/IP common description (Lantronix XPort)

This description applies to the controllers including the Ethernet interface based on the XPort-03 Module of Lantronix Inc.

This document is only an overview, which describes how the chambers use the XPort services. Some tools and a complete documentation can be downloaded from www.lantronix.com

You can download the Lantronix Tool “Device Installer” as well from the Homepage of Lantronix.

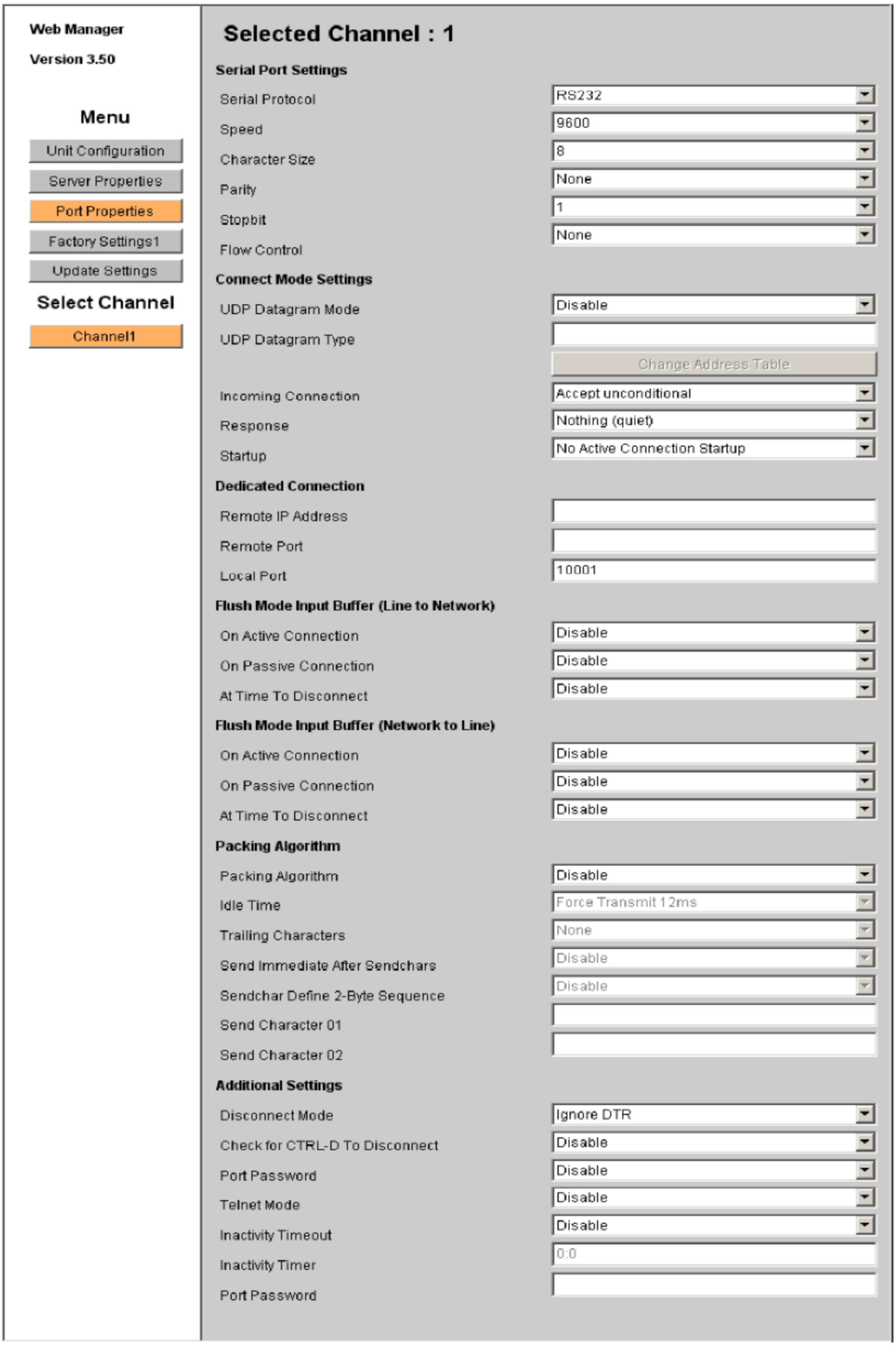
3.1 Protocol description

If nothing else is described in this document, the XPort Module is used with factory default settings. (see picture below)

Note:

The description of MODBus-Protocol (Chapter 2) is needed for a successful connection to a BINDER controller. The MODBus address at the BINDER controller should always be set to 1 (default value).

Pressing the button “Factory Settings 1” sets the XPort always back to the configuration needed to communicate internally with the BINDER controllers.



Internet Browser Java configuration

3.1.1 Network specification

Network Interface:

RJ45 Ethernet 10BASE-T or 100BASE-TX (auto-sensing)

Compatibility

Ethernet: Version 2.0/IEEE 802.3

3.1.2 Used TCP/UDP protocols and ports (factory settings)

Port	Protocol	Used as
TCP Port 80	http	webserver for configurations
TCP Port 9999	telnet	telnet for configurations not implemented in webserver
TCP Port 10001	Raw binary	Serial MODBus data tunnel used for communication with BINDER chambers
UDP port 30718		With IP address = 255.255.255.255 used for detecting XPort-03 devices in the network
UDP Port 68	dhcp	DHCP-Client

3.2 Detecting XPort-03 devices in the network (UDP)

For detecting all XPort-03 (BINDER Chamber) a UDP Broadcast (IP=255.255.255.255) on port 30718 has to be send. Every XPort available answers with 30 Byte of data.

The IP of the XPort can be seen in the returning frame under Internet Protocol → Source

The last 8 bytes of the data include the MAC address of the XPort.

Return data example (hex bytes):

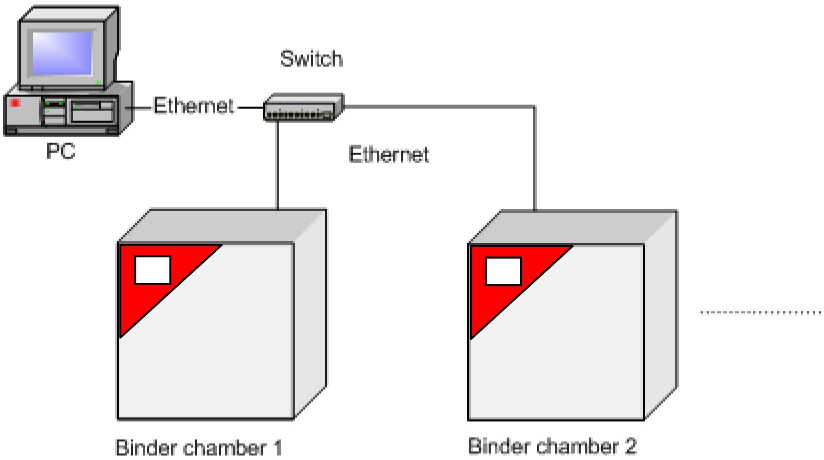
………………. 00 20 4A 8A A0 F7

MAC = 00-02-4A-8A-A0-F7

The MAC is labeled on the chamber near the Ethernet port. With the MAC it is possible to identify the answering chambers.

3.3 Connecting a chamber via TCP/IP

To read and write data, port 10001 is used as a TCP/IP tunnel. This means the MODBus protocol is tunneled in RAW binaries through the XPort device. Only the MODBus telegram is sent to the controller.



3.3.1 Assigning a fix IP Address (only if no DHCP is available)

The unit's IP address must be configured before it can work correctly on a network. You have several options for assigning an IP to your unit. We recommend that you manually assign the IP address via the network using Device Installer software, which is on the product CD.

Installing DeviceInstaller

1. Open DeviceInstaller on the CD-ROM.

   If the CD does not launch automatically:

   a. Click the Start button on the Task Bar and select Run.

   b. Enter your CD drive letter, colon, backslash, Launch.exe (e.g., D:\Launch.exe).

2. Respond to the installation wizard prompts.

Assigning an IP Address

The unit's IP address is normally set to 0.0.0.0 at the factory. The hardware address is printed on the chamber near the Ethernet port. The unit is DHCP enabled as the default.

To manually assign an IP address:

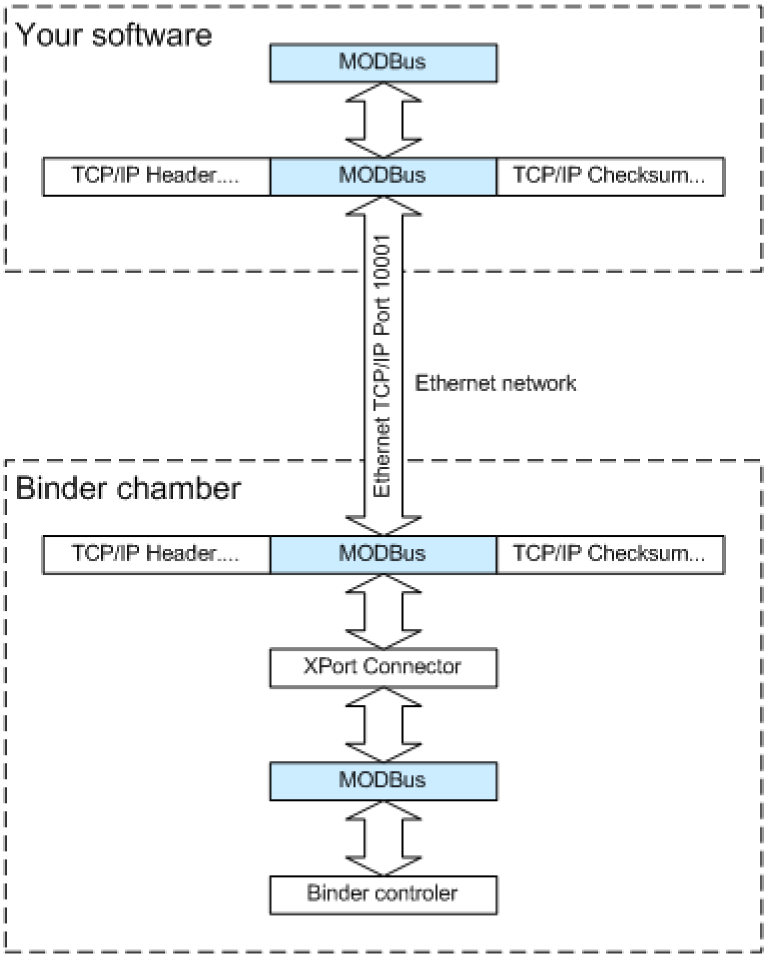
1. Click Start → Programs → Lantronix → DeviceInstaller → DeviceInstaller. If your PC has more than one network adapter, a message is displayed. Select an adapter and click OK.

   Note: If the unit already has an IP address (e.g., DHCP has assigned an IP address), click the Search icon and select the unit from the list of Lantronix device servers on the local network.

2. Click the Assign IP icon
3. If prompted, enter the hardware address and click “Next”. The hardware address is printed on the product label near the RJ45 connector at the rear or on the side of the chamber
4. Select “Assign a specific IP address” and click “Next”.
5. Enter the IP address. The Subnet mask displayed automatically based on the IP address; if desired, you may change it. On a local network, you can leave the Default gateway blank (all zero). Click Next.
6. Click the Assign pushbutton and wait several seconds until a confirmation message is displayed. Click Finish.

3.4 Communicating with the chamber

TCP/IP is used to tunnel the MOD Bus telegram through the Ethernet network connection. The default port is 10001.



4. BINDER TCP/IP communication with the BINDER Controller World (BCW T4.x)

4.1 UDP General

Always Port 9001

All read and write telegrams must end with CRLF (0x0D0A). Otherwise CRLF must not appear in a telegram.

4.1.1 Detecting T4.x Ethernet devices in the network (UDP)

Communication:

1. Chamber detection (UDP IP:255.255.255.255):
2. Computer sends "GetConfigDirect" (ASCII format)
3. T4.x answers with chamber information (Timeout:3000ms + max. 500 characters per answer):

"IP:" + strIP + " Port:" + strTCPPort +" MAC:" + strMAC + TAB +

"Chambertype:strChamberType" + TAB +

"Controllertype:strControllerType" + TAB +

"COMType:BCWDirect" + TAB +

"ProgramController:strProgramController";

Explanation of the C# Notation:

strIP = IP address of the chamber as string (xxx.xxx.xxx.xxx)

strTCPPort = UDP Port number decimal as string

strMAC = "xx-xx-xx-xx-xx-xx" as string

strChamberName = Chamber (default), otherwise customer's setting on the T4.x (max 40 characters)

strChamberType = Examples: "CB 160 (E6)", "KT 115 (E6)"... (max 250 characters, no special characters)

strControllerType = "T4.2", "T4.1" .... (max 250 characters and ".", no special characters)

strProgramController = "True" or "False"

TAB = ASCII for [TAB]

CRLF = ASCII for [CR][LF]

4.2 TCP General

Always Port 9000.

If writing the value, the value returns as data (corrected, if appropriate)

4.2.1 Connecting a chamber via TCP – general

All read and write telegrams must end with CRLF (0x0D0A), in either direction.

Connection established by computer through port 9000; 4 trials each with a 1000 ms timeout

1. Computer sends command to T4.x (e.g. communication test: "CANIDGetValue:10010010"+CRLF); timeout:1000 ms
2. T4.x transmits byte counts of data to the computer; timeout 1000ms
3. T4.x transmits data to the computer; timeout 1000ms

Before T4.x sends data, it transmits a telegram with the data byte counts (decimal , max 100 characters) to the computer. This telegram always must end with CRLR.

Structure:

For transmission, the maximum byte count is100 characters (e.g. 23789 Bytes = 5 characters)

All telegrams have the structure "command:command value“. They are separated by the colon.

If a value is included in the telegram, it is also separated by a colon: "command:command value:value"

The answer is always "command:command value". If a value string is included, it is separated by a colon.

The T4.x answers in case of an error (any error) with the key word "Error" as command.

CANID information:

The computer can request information of CANIDs (ID0xYYYYYYYY, with “Y” meaning CANID in HEX)

<u>Example:</u> CANID ID0x11800080 (Actual temperature value of T4.x)

1. Computer sends to T4.x: "BCWDirectGetCANIDInfo:ID0x11800080"
2. T4.2 answers to computer:

   "BCWDirectGetCANIDInfo:ID0x11800080:"

   + "Befehl=ID0x11800080" + CRLF

   + "HexAdress=11800080" + CRLF

   + "VarType=float" + CRLF

   + "ReadWords=" + CRLF

   + "ConstValue=" + CRLF

   + "LimitLow=123.0" + CRLF

   + "LimitHigh=600.0" + CRLF

+"ReadWrite=RO" + CRLF

+ "Doku=P1_ChamberTemperature" + CRLF

+ "Unit=1"

Wireshark Raw Data of the answer:

```
42 65 66 65 68 6c 3d 49 44 30 Befehl=ID0
78 31 31 38 30 30 30 38 30 0d 0a 48 65 78 41 64 x11800080..HexAd
72 65 73 73 3d 31 31 38 30 30 30 38 30 0d 0a 56 ress=11800080..V
61 72 54 79 70 65 3d 66 6c 6f 61 74 0d 0a 52 65 arType=float..Re
61 64 57 6f 72 64 73 3d 0d 0a 43 6f 6e 73 74 56 adWords=..ConstV
61 6c 75 65 3d 0d 0a 4c 69 6d 69 74 4c 6f 77 3d alue=..LimitLow=
31 32 33 2e 30 0d 0a 4c 69 6d 69 74 48 69 67 68 123.0..LimitHigh
3d 36 30 30 2e 30 0d 0a 52 65 61 64 57 72 69 74 =600.0..ReadWrit
65 3d 52 4f 0d 0a 44 6f 6b 75 3d 50 31 5f 43 68 e=RO..Doku=P1_Ch
61 6d 62 65 72 54 65 6d 70 65 72 61 74 75 72 65 amberTemperature
0d 0a 55 6e 69 74 3d 31 0d 0a                   ..Unit=1..
```

4.2.2 TCP communication test

1. Computer sends to T4.x: "CANIDGetValue:10010010"
2. If OK, T4.x answers to computer:
   "CANIDGetValue:10010010:Communication OK".
   Otherwise (in case of any error) T4.x answers with the key word "Error".

4.2.3 Reading a CANID value via TCP

1. Computer sends to T4.x: "CANIDGetValue:CANIDin32BitHex"
2. If OK, T4.x answers to computer:
   “CANIDGetValue:CANID in 32Bit Hex:value in string format”.
   Otherwise (in case of any error) T4.x answers with the key word "Error".

4.2.4 Writing a CANID value via TCP

1. Computer sends to T4.x: "CANIDWriteValue:CANID in 32Bit Hex:value in internal format"
2. If OK, T4.x answers to computer:
   "CANIDWriteValue:CANID in 32Bit Hex:value in internal format "
   Otherwise (in case of any error) T4.x answers with the key word "Error".

4.2.5 Types of variables (internal format)

Data format	Representation (always 32bit)
unsignedbyte	only LSB evaluated AND mask =0x000000FF
signedbyte	only LSB evaluated AND mask =0x000000FF
float	acc. to IEEE 754
longsignedint	-2147483648 up to 2147483647
longunsignedint	0 up to 4294967295
binary	00000000 00000000 00000000 00000000 up to 11111111 11111111 11111111 11111111
binary16	00000000 00000000 00000000 00000000 up to 00000000 00000000 11111111 11111111
hex	0000 0000 up to FFFF FFFF (longunsignedint)
hex16	0000 0000 up to 0000 FFFF (unsignedint)
boolean	LSB is decisive (AND mask 0x00000001)
enum	see _unsignedint
string	Maximum 40 characters,
date	Date: always yyyymmdd in longsignedint as a decimal value (yyyymmdd) Example: 23.03.2007 = 2007 = year, 03 = month, 23 = day. Results in 20070323 in the telegram.
timeabs	Time: hhmmsscc 2 characters each for hours, minutes, seconds, and milliseconds as a decimal value (longunsignedint) 8 characters include milliseconds. When 6 characters, the milliseconds are set to zero. Internal format: always 24 hours. Example: 8 characters: 23:45:38,83 = 23453883 Example: 6 characters: 23:45:38,00 = 234538
version	xxxxyyzz one Hex value per Byte (longunsignedint) Version numbers for hardware and software xxxx = Byte<3><2>: Main version numbers 0x0001 up to 0xFFFF (0x0000 = reserved = no information) yy = Byte<1> subversion (0x00 = Beta version, for development only) zz = Byte<0> Bugfix counter

Data format	Representation (always 32bit)
seconds	Count of seconds (= longunsignedint)
signedint	AND mask = 0x0000 FFFF
unsignedint	AND mask = 0x0000 FFFF
partno	BINDER number xxxx-xxxx e.g. BINDER number = 5014-01A7 → Decimal value = 1343488423
CANID	Contains a CANID (29 bit) AND mask 1FFF FFFF
Int16Div10	16 Bit Integer (value divided by 10 results in real value)

4.2.6 Units

Parameter	unit	Unit:unit number
none	none	0 (default)
Temperature	Kelvin	1
Percent	none	2
Time relative	seconds	3
Pressure	mbar	4
Light	SI unit	5
Voltage	Volt	6
Current	Ampere	7
Power	Watt	8
Volume Percent	none	9
TemperatureDelta	Kelvin	10
PercentDelta		11

4.2.7 CANID Table

Value	CAN ID (hex)
All (if function is available)	
Door open, closed	0x10010100 (Read Only)
KT	
Temperature, actual value	0x11400080 (Read Only)
Temperature, set-point value	0x114000C0 (Read/Write)
Fan speed, set-point value	0x112000E1 (Read/Write)
CB	
Temperature, actual value	0x11400080 (Read Only)
Temperature, set-point value	0x114000C0 (Read/Write)
CO_2, actual value	0x11600B07 (Read Only)
CO_2, set-point value	0x116000C9 (Read/Write)
O_2, actual value	0x11600082 (Read Only)
O_2, set-point value	0x116000CC (Read/Write)
Humidity, actual value	0x100C1066 (Read Only)
Sterilization Active (No = 0, Yes = 1)	0x10010102 (Read Only)
AI	
Temperature, actual value	0x11400080 (Read Only)
Temperature, set-point value	0x114000C0 (Read/Write)
Humidity, actual value	0x11600083 (Read Only)
Humidity, set-point value	0x116000C6 (Read/Write)