PMA Prozeß- und Maschinen-Automation GmbH

Multifunctionunit KS 98-1

KS 98-1 MODBUS

KS98-1 Modbus

Description of functionality on the basis of examples.

9499 040 88711

Valid from: 02/2008

Explanation of symbols:

ⓘ **General information**

⚠ **General warning**

Caution: ESD-sensitive components

This is a publication of
PMA Prozeß- und Maschinen Automation
P.O. Box 310229
D-34058 Kassel
Germany

Content

1 General

This document describes the features and the use of the new modbus interface of the KS98-1 on the basis of examples. Only those features will be described, that are necessary after downloading a complete engineering with the engineering tool ET/KS98plus successfully.

At first the allocation of data in the available modbus address range is described. Afterwards the directly addressable data are characterized and the read and write access is illustrated with examples.

Finally the access to data of a function block of an engineering is described, which are not single addressable. In this case a special block transfer of the modbus protocol is used to transfer the so called ‚B' key messages of the ISO protocol.

In the examples a communication via the field interface with address = 1 is assumed. On use of the front interface the address = 0 has to be uased (at the front interface address = 0 is not a broadcast address!).

2 Bus protocol

2.1 Composition of a transmission byte

Originally, the MODBUS protocol was defined for the communication between a supervisory system and the Modicon® PLC. It used a master/slave structure, in which only one device (master) is able to initiate data transactions (queries). The query message from the master is answered (response) by other devices (slaves), which supply the requested data.
Moreover, the master can address a specific slave via its MODBUS address, or address all connected slaves by means of a general message (broadcast).
The MODBUS protocol determines the transmission formats for the query and the response. Function codes define the actions to be executed by the slaves.
Within the device, the MODBUS protocol uses the RTU (remote terminal unit) mode, i.e. every transmitted byte of a message contains two hexadecimal characters (0...9, A...F).

The composition of a byte in the RTU-protocol is as follows:

Start bit	8 data bits	Parity/Stop bit	Stop bit

2.2 General message frame

The message is read into a data buffer with a defined maximum length. Longer messages are not accepted, i.e. the device does not answer.

The message consists of the following elements:

Device address	Function code	Data field	CRC	End of frame detection
1 byte	1 byte	N * 1 bytes	2 bytes	

- Device address (Addr)
  The device address is used for identification. Device addresses can be assigned in the range of 1...127.
  The device address ‘0’ is reserved for ‘Broadcast’ messages to all slaves. A broadcast message can be transmitted e.g. with a write instruction that is then executed by all the slaves on the bus. Because all the slaves execute the instruction, no response messages are generated.
- Function code
  The function code defines the transaction type in a message. The MODBUS specification defines more than 17 different function codes. Supported codes are described in Section 3.6. „Function codes“.
- Data field
  The data field contains the detailed specifications of the transaction defined by the function code. The length of the data field depends on the function code.
- CRC
  As a further means of fault detection (in addition to parity bit detection) a 16-bit cyclical redundancy check (CRC) is performed. The CRC code ensures that communication errors are detected. For additional information, see Section 3.2.1. "CRC".
- End of frame detection
  The end of a message is defined by a period of 3,5 characters, during which no data transfer occurs. For additional information, see Section 3.2.2. „End of frame detection“

Further information is given in the documents named in **[1]** or under http://www.modbus.org.

2.2.1 CRC

The CRC is a 16-bit value that is attached to the message. It serves to determine whether a transmitted message has been received without errors. Together with the parity check, this should detect all possible communication errors.

If a parity fault is detected during reading, no response message will be generated.

The algorithm for generating a CRC is as follows:

① Load CRC register with FFFFhex.
② Exclusive OR the first transmit/receive byte with the low-order byte of the CRC register, putting the result into the CRC register, zero-filling the MSB.
③ Shift the CRC register one bit to the right.
④ If the expelled bit is a '0' repeat step 3.
If the expelled bit is a ‘1’, exclusive OR the CRC register with value A001hex.
⑤ Repeat steps 3 and 4 for the other 7 data bits.
⑥ Repeat steps 2 to 5 for all further transmit/receive bytes.
⑦ Attach the result of the CRC register to the message (low-order byte first, then the high-order byte).
When checking a received message, the CRC register will return ‘0’, when the message including the CRC is processed.

2.2.2 End of frame detection

The end of a message (frame) is defined as a silence period of 3.5 characters on the MODBUS.
A slave may not start its response, and a master may not start a new transmission before this time has elapsed.

However, the evaluation of a message may begin, if a silence period of more than 1.5 characters occurs on the MODBUS. But the response may not start before 3,5 characters of silence.

2.3 Transmission principles

Two transmission modes are used with MODBUS:

- **Unicast mode**
- **Broadcast mode**

In the Unicast mode, the master addresses an individual device, which processes the received message and generates a response. The device address can be 1...247. Messages always consist of a query (request) and an answer (response). If no response is read within a defined time, a timeout error is generated.

In the Broadcast mode, the master sends a write instruction (request) to all participants on the bus, but no responses are generated. The address ‘0’ is reserved for broadcast messages.

2.4 Function codes

Function codes serve to execute instructions. The device supports the following function codes:

Function code		Description	Explanation
hex	dez		
0x03	3	Read Holding (Output) Register	Reading of process data, parameters, and configuration data
0x04	4	Read Input Register	Reading of process data, parameters, and configuration data
0x06	6	Preset Single Register (Output)	Wordwise writing of a value (process value, parameter, or configuration data)
0x10	16	Preset Multiple Register (Output)	Wordwise writing of several values (process data, parameter or configuration data)
0x17	23	Read/Write Multiple Register	Lesen und Schreiben von Daten im Blockformat

The behaviour of function codes 3 and 4 is identical.
The following sections show various examples of message composition.

2.4.1 Reading several values

Messages with function codes 3 or 4 are used for (wordwise) reading of process data, parameters or configuration data. For reading ‘Float’ type data, 2 values must be requested for each datum.

The composition of a read message is as follows:
Request:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	03 or 04	Reading process data, parameters or configuration data
Start address High Start address Low	02 8A	Starting address 650
No. of values	00 02	2 datums (2 words)
CRC	CRC-Byte1 CRC-Byte2	

Response:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	03 oder 04	Reading process data, parameters or configuration data
No. of bytes	04	4 data bytes are transmitted
Word 1	00 DE	Process data, parameters or configuration data. Address 650= 222
Word 2	01 4D	Process data, parameters or configuration data. Address 651= 333
CRC	CRC-byte1 CRC-byte2	

A broadcast message is not possible for function codes 3 and 4.

If the first addressed value is not defined, an error message ”ILLEGAL DATA ADDRESS” is generated.
If no further data are defined in the areas to be read following the first value, these areas will be entered with the value ”NOT DEFINED VALUE”. This enables areas with gaps to be to be read in a message.

2.4.2 Writing a single value

Messages with function code 6 are used for (wordwise) writing of process data, parameters or configuration data as integers. This function is not suitable for writing ‘Float’ type data.

The composition of a write message is as follows:
Request:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	06	Writing a single value (process data, parameter or configuration)
Write address High Write address Low	02 8A	Write address 650
Value	00 7B	Preset value = 123
CRC	CRC-byte1 CRC-byte2	

Response:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	06	Writing a single datum (process data, parameter or configuration)
Write address High Write address Low	02 8A	Write address 650
Value	00 7B	Preset value = 123
CRC	CRC-Byte1 CRC-Byte2	

If everything is correct, the response message corresponds exactly to the default.

ℹ **The devices can also receive this message as a broadcast with the address ‘0’.**

ℹ **A default value in the ‘Real’ data format is not possible, as only 2 bytes can be transmitted as value.**

ℹ **If a value is outside the adjustable range, the error message ”ILLEGAL DATA VALUE” is generated. The datum remains unchanged. Also if the datum cannot be written (e.g. configuration data, and the device is online), an error message ”ILLEGAL DATA VALUE” is generated.**

2.4.3 Writing several values

Messages with function code 16 are used for (wordwise) writing of process data, parameters or configuration data. For writing ‘Float’ type data, 2 values must be transmitted for each datum.

The composition of a write message is as follows:
Request:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	10	Writing several process values, parameters or configuration data
Start address High Start address Low	02 8A	Write address 650
No. of values	00 02	2 values
No. of bytes	04	4 data bytes are transmitted
Word 1	00 DE	Process value, parameters or configuration data. Address 650 = 222
Word 2	01 4D	Process value, parameters or configuration data. Address 651 = 333
CRC	CRC byte1 CRC byte2	

Response:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	10	Writing several process values, parameters or configuration data
Start address High Start address Low	02 8A	Write address 650
No. of values	00 02	2 process values, parameters or configuration data
CRC	CRC byte1 CRC byte2	

The devices can also receive this message as a broadcast with the address ‘0’.

If the first value is not defined, an error message ”ILLEGAL DATA ADDRESS” is generated.
If the first value cannot be written (e.g. configuration data, and the device is online), an error message ”ILLEGAL DATA VALUE” is generated.

If no further data are defined or cannot be written in the specified areas following the first value, these areas will be skipped. The data in these locations remains unchanged. This enables areas with gaps, or that are currently not writable, to be changed with a message. No error message is generated.

If a value is outside the adjustable range, the error message ”ILLEGAL DATA VALUE” is generated. Subsequent data are not evaluated. Previously accepted correct data are active.

2.4.4 Reading and writing data in blockformat

Messages with function code 17 are used for reading and writing data in blockformat. The data content of the KS 98-1 always consists of ASCII-data.

Request:

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	17	Reading and writing data in blockformat
Start address High Start address Low	0 0	Bei KS 98-1 ohne Bedeutung
No. of read data	0 1	=1. Bei KS 98-1 ohne Bedeutung, da sich die Anzahl der Lesedaten aus dem Inhalt der Schreibdaten ableitet
Schreibadresse High Schreibadresse Low	0 0	Bei KS 98-1 ohne Bedeutung
No. of write data	0 n	Anzahl 'n' der Datenworte in den Schreibdaten
No. of write data	2*n	Anzahl '2*n' der Datenworte in den Schreibdaten
Write data 1...n	x x ...	Datenblock der Schreibdaten
CRC	CRC-byte1 CRC-byte2	

Response :

Field name	Value (hex)	Explanation
Address	11	Address 17
Function	17	Reading and writing data in blockformat
No. of response data	2*n	Anzahl '2*n' der Datenworte in den Schreibdaten
Response data 1...n	x x ...	Datenblock der Antwortdaten
CRC	CRC-Byte1 CRC-Byte2	

A broadcast-message for functioncode 0x17 is not possible.

2.5 Error record

An error record is generated, if a message is received correctly, but message interpretation or the modification of a datum is not possible.

If a transmission error is detected, no response is generated. The master must retransmit the message.

Detected transmission errors are:

- Parity fault
- Framing error (no stop bit received)
- Overrun error (receiving buffer has overflowed or data could not be retrieved quickly enough from the UART)
- CRC error

The composition of the error record is as follows:

Field name	Value	Explanation
Address	11	Address 17
Function	90	Error record for the message 'Writing several parameters or configuration data'. Composition: 80hex + function code
Error code	02	ILLEGAL DATA ADDRESS
CRC	CRC byte1 CRC byte2	

In the 'Function' field, the most significant bit is set.
The error code is transmitted in the subsequent byte.

2.5.1 Error codes

The following error codes are defined:

Code	Name	Explanation
01	ILLEGAL FUNCTION	The received function code is not defined in the device.
02	ILLEGAL DATA ADDRESS	The received address is not defined in the device, or the value may not be written (read only). If several data are read simultaneously (function codes 01, 03, 04) or written simultaneously (function codes 0F, 10), this error is only generated if the first datum is not defined.
03	ILLEGAL DATA VALUE	The received value is outside the adjusted limits or it cannot be written at present (device is not in the configuration mode). If several data are written simultaneously (function codes 0F, 10), this error is only generated if the first datum cannot be written.
04	SLAVE DEVICE FAILURE	More values are requested than permitted by the transmission buffer.

Other error codes specified in the MODBUS protocol are not supported.

3 Modbus definitions

3.1 Implemented modbus addresses

The modbus address range includes the addresses for access to data in integer and floating point format. The range 0x0001...0x3FFF is available for data in integer format and the range 0x8000...0xFFFF for data in floating point format. The address of data in floating point format is calculated by: address for integer format * 2 + 0x8000.

The following generically allocation of addresses has been fixed:

0x0001...0x004F	General process data of instrument and instrument parameter
0x0050...0x0103	process data of L1READ / L1WRIT blocks with block numbers 1...20 (9 addresses each)
0x0110...0x015F	5 function blocks MBDATA (new) with 16 addresses each)
0x0160...0x0687	30 ranges for process data of controller function blocks (CONTR / CONTR+ / PIDMA). (44 addresses each)
0x0688...0x09F7	40 ranges for process data of programmer function blocks (APROG / DPROG). (22 addresses each)
0x8000...0xFFF	data from range 0x0001...0x3FFF in floating point format.

The modbus address given in a message has to be defined in all cases. Following addresses used in messages with more data do not have to be active. While reading the switch off value (-32000 / -1.5e37) is transferred. While writing the not active addresses will be ignored.

3.2 Implemented modbus-function codes

With the standard messages single or multiple data are transferred, to which a modbus address is directly assigned. These are the above described device and level-1 data whose modbus addresses are defined via the basic modbus address and an offset address.

For this purpose the modbus function codes are used that are implemented in the other PMA devices as well. The layout is already described there.

3 or 4	:	Read single or multiple data
6	:	Write single data
16	:	Write multiple data

For the transmission of the codes B1...B4 that are transmitted in ISO1745 mode via function bloc protocol, the modbus function code 23 (0x17) is used. This mode offers a combined write/read message and is used in general only by the engineering tool.

4 Data structures

4.1 Instrument data

4.1.1 Definitions

The instrument data use the modbus addresses 0x0001...0x004F

Address	Data	Access	Range
0x0001	Status 1	R	0 ... 63
0x0002	Status 2	R	0 ... 63

Status 1:

Bit 15...6	:	0		
Bit 5	:	Parameter update	[0] - no	[1] - yes
Bit 4	:	Power-fail check	[0] - not active	[1] - active
Bit 3	:	E²PROM error	[0] - no	[1] - yes
Bit 2	:	Safety status	[0] - not active	[1] - active
Bit 1	:	Instrument status	[0] - online	[1] - configuration
Bit 0 (LSB)	:	Sensor failure (common message)	[0] - no	[1] - yes

Status 2:

Bit 15...6	:	0		
Bit 5	:	Field interface	[0] - read/write	[1] - read
Bit 4	:	Main menu display by operation	[0] - possible	[1] - blocked
Bit 3	:	Configuration menu by operation	[0] - possible	[1] - blocked
Bit 2	:	Parameter display by operation	[0] - possible	[1] - blocked
Bit 1	:	Wiring finished	[0] - no	[1] - yes
Bit 0 (LSB)	:	Engineering existing.	[0] - no	[1] - yes

Address	Data	Access	Range
0x0005	Operation mode	R/W	0 ... 1 / 0 ... 2
0x0006	Safety status	R/W	0 ... 1
0x0007	Reset of local data change	R/W	0 ... 1 / 0
0x0008	Delete engineering	R/W	0 ... 1 / 1
0x0009	Finish wiring	R/W	0 ... 1 / 1
0x000A	Debug mode	R/W	0 ... 127
0x000B	Activate power-fail check	R/W	0 ... 1 / 1
0x000C	Write permission for field interface	R/W	0 ... 1

Operation mode:	0	-	Online
	1	-	Configuration (Offline)
	2	-	Cancel configuration (Esc) (Write only)
Safety status:	0	-	not active
	1	-	active
Reset of local data change:			
	0	-	Parameter not changed / Reset Flag
	1	-	Parameter changed (read only)
Delete engineering:	0	-	Engineering not deleted (read only)
	1	-	Engineering deleted / delete
Finish wiring:	0	-	Wiring not finished (read only)
	1	-	Wiring finished / finish

Debug-Mode:	2^0	-	AINP1 (0 - off / 1 - on)
	2^2	-	AINP3
	2^3	-	AINP4
	2^4	-	AINP5
	2^5	-	AINP6
	2^6	-	DINPUT

Activate power-fail check:

0	-	not active (read only)
1	-	active / activate

Write permission for field interface:

0	-	Read and write permission
1	-	Only read permission

Address	Data	Access	Range
0x0010	Address field interface	R/W	1 ... 247
0x0011	Flag for address changing disabled	R/W	0 ... 1

Address field interface:

Flag for address changing disabled:
A single write access to this modbus address disables further write accesses to the modbus address 0x0010 . A new write access is possible only, if a write access to modbus address 0x0042 was made, if the address was changed via the instrument front panel, or if disabling was removed by deleting the flag.

Address	Data	Access	Range
0x0014	Password mode	R/W	0 ... 3
0x0015	Password attempts	R/W	0 ... 99
0x0016	Password status	R	0 ... 2

Password mode:
The password mode determines the access possibilities to the KS98 data via the interface.

Password attempts:
Determines the number of permitted unsuccessful attempts during password transmission (log-in). When exceeding the number of permitted attempts, KS98 is switched to the OFFLINE mode and the password as well as the existing engineering are deleted.

Password status:

0	-	No password existing
1	-	Password existing, but not active (in logged-in condition)
2	-	Password existing and active (in logged-out condition)

Address	Data	Access	Range
0x0020	Basic HW options: Modul A, P	R	2101 ... 2999
0x0021	Ext. HW options: Modul B, C	R	0000 ... 9999
0x0022	SW options	R	0000 ... 9999
0x0023	SW code number (7.-10.digit)	R	7254
0x0024	SW version (11.+12. digit)	R	0000 ... 0099
0x0025	Operating version	R	0000 ... 0099
0x0026	E²PROM version	R	0000 ... 0099
0x0027	HW code number (6.-9. digit)	R	6300 ... 8939
0x0028	Modul 1, Modular options card C	R	0, 46-49, 76-78
0x0029	Modul 2, Modular options card C	R	0, 46-49, 76-78
0x002A	Modul 3, Modular options card C	R	0, 46-49, 76-78
0x002B	Modul 4, Modular options card C	R	0, 46-49, 76-78
0x002C	Modular options card C	R	0 ... 1
0x002E	Engineering length	R	0 ... 28399

Basic HW options: Value = 21xy
with : 21 = Instrument type KS98
and xy = 01: Relay OUT1, 2, 4, 5
e.g. = 21: Current OUT1, 2
Relay OUT 4, 5
e.g. = 99: Extension

Ext. HW options: Value = abcd
with ab = 00: no option card B

ab = 00:	no option card B
01:	option card B with TTL interface, di/do
02:	option card B with RS485/422 interface, di/do, clock
10:	option card B with Profibus DP interface, di/do
11:	option card B with Interbus S interface
99:	other option card B
cd = 00:	no option card C
07:	option card C with INP3/4, OUT3, di/do
08:	option card C, modular
99:	other option card C

SW options:
This value is currently not used in KS98.

SW code number:
SW version:
Contains 7.-10. and 11.+12. digits of SW code number
4012 157 254VR

Operationg version:
Operating version 1... The operating version is calculated unattached to SW code number.

Version number of E²PROM's:
This value is currently not used in KS98.

Engineering length:
Used number of bytes in RAM memory of engineering memory (readable while online mode).

Modulare options card C:

0	-	not connected
1	-	connected

Modul x, Modular options card C:

0	-	not equipped
46	-	equipped with thermocouple input modul
47	-	equipped with current output modul
48	-	equipped with voltage output modul
49	-	equipped with Digital-I/O modul
76	-	equipped with frequency input modul
77	-	equipped with resistance input modul
78	-	equipped with voltage input modul

Address	Data	Access	Range
0x0030	Time year	R/W	0 ... 99 / 1970...2069
0x0031	Time month	R/W	1 ... 12
0x0032	Time day	R/W	1 ... 31
0x0033	Time hours	R/W	0 ... 23
0x0034	Time minutes	R/W	0 ... 59
0x0035	Parameter display by operation	R/W	0 (possible), 1 (blocked)
0x0036	Configuration display by operation	R/W	0 (possible), 1 (blocked)
0x0037	Main menu display by operation	R/W	0 (possible), 1 (blocked)

Instrument parameter:
The modbus addresses ... are only active with option real time clock.
The range is alternative: 00...69, 70...99 = 2000...2069, 1970...1999 or 970...2069.

The instrument parameter with modbus addresses 0x0035...0x0037 affect the possiblity to change existing settings via the operation. Their status is connected via an OR function with the relevant digital inputs of function STATUS , type number 125, if it is used. The parameters are stored in EEPROM, i.e. they are available also after power-on. The priority of the OR function results is different.

- Blocking of main menu display blocks parameter anc configuration display too.
- Parameter display blocking blocks the configuration display too.
- Configuration display blocking includes no other blocking.

Configuration display blocking means that the instrument cannot leave the online mode by operator entry, but only by interface message and that the configuration display via operation is not possible.

Parameter display blocking means that the parameters cannot be displayed. This does not affect the change of process data on the operating pages.

Address	Data	Access	Range
0x0040	Protocol mode	R/W	0 ... 3
0x0041	Baudrate	R/W	(0), 1 ... 4
0x0042	Instrument address	R/W	1 ... 247
0x0043	Main frequency	R/W	0, 1
0x0044	Language	R/W	0, 2
0x0045	CAN node-Id	R/W	1 ... 24
0x0046	CAN baudrate	R/W	0 ... 8
0x0047	Status of outputs while download	R/W	0, 1
0x0048	Switch on delay CAN	R/W	0 ... 10

Protocol mode:	0	-	ISO 1745	
	1	-	Profibus DP	
	2	-	Interbus S	
	3	-	Modbus	
Baudrate:	0	-	not adjustable	(No field interface / Profibus DP / Interbus S)
	1	-	2400 Baud	
	2	-	4800 Baud	
	3	-	9600 Baud	
	4	-	19200 Baud	

Instrument address:
Setting the address of the field interface. The range is 1 ... 247.

Main frequency:	0	-	50 Hz
	1	-	60 Hz
Language:	0	-	german
	1	-	english
	2	-	french

CAN node-Id:
Node number of KS98. Id=1 means, KS98 network master (NMT) .

CAN baudrate:	0	-	10 KBaud
	1	-	20 KBaud
	2	-	50 KBaud
	3	-	100 KBaud
	4	-	125 KBaud
	5	-	250 KBaud
	6	-	500 KBaud
	7	-	800 KBaud
	8	-	1000 KBaud

Status of outputs while download:

0 - all outputs switch off
after switching ONLINE -> OFFLINE set status of RAM as invalid

1 - Freeze status of outputs at the last value / status
Set status of RAM while switching ONLINE -> OFFLINE as valid

4.1.2 Examples

1. Reading of instrument status informationen 'Status 1' (0x0001), 'Status 2' (0x0002) and 'Instrument status' (0x0005):

   => Reading of 5 values from instrument with the address 1 starting with modbus address 0x0001

   Structure of request message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*CrcH*	*CrcL*
01	03	00	01	00	05	xx	xx

   Structure of response message (Hex representation):

Adr	*Mode*	*Anz*	*Status1*		*Status 2*		*32000*		*-32000*		*Instrument status*		*CrcH*	*CrcL*
01	03	0A	00	22	00	03	81	0C	81	0C	00	01	yy	yy

2. Switching to Offline 'Instrument status' (0x0005) = 1:

   => Writing of one value to instrument with address 1 at modbus address 0x0005

   Structure of the send message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*Anz*	*Offline*		*CrcH*	*CrcL*
01	10	00	05	00	01	02	00	01	xx	xx

   Structure of response message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*CrcH*	*CrcL*
01	10	00	05	00	01	yy	yy

4.2 L1WRIT / L1READ

The addresses of data of function types L1WRIT and L1READ are dependant of the block number of the function block.
Calculating of the start address = 0x0050 + (Block number –1) * 0x0009

4.2.1 Structure for L1WRIT

Offset	Data	Access	Range
0x0	Digital outputs z1...z15	R/W	0 ... 32767
0x1	Analogue output Y1	R/W	-29999 ... 200000
0x2	Analogue output Y2	R/W	-29999 ... 200000
0x3	Analogue output Y3	R/W	-29999 ... 200000
0x4	Analogue output Y4	R/W	-29999 ... 200000
0x5	Analogue output Y5	R/W	-29999 ... 200000
0x6	Analogue output Y6	R/W	-29999 ... 200000
0x7	Analogue output Y7	R/W	-29999 ... 200000
0x8	Analogue output Y8	R/W	-29999 ... 200000

Digital output z1 at bit 0 (LSB).

4.2.2 Structure for L1READ

Offset	Data	Access	Range
0x0	Status 1	R	0 ... 63
0x1	Status 2	R	0 ... 63
0x2	Analogue input X1	R	-29999 ... 200000
0x3	Analogue input X2	R	-29999 ... 200000
0x4	Analogue input X3	R	-29999 ... 200000
0x5	Analogue input X4	R	-29999 ... 200000
0x6	Analogue input X5	R	-29999 ... 200000
0x7	Analogue input X6	R	-29999 ... 200000
0x8	Analogue input X7	R	-29999 ... 200000

Status 1:

Bit 15...6 :	0			
Bit 5	:	Status d6	[0] - off	[1] - on
Bit 4	:	Status d5	[0] - off	[1] - on
Bit 3	:	Status d4	[0] - off	[1] - on
Bit 2	:	Status d3	[0] - off	[1] - on
Bit 1	:	Status d2	[0] - off	[1] - on
Bit 0 (LSB)	:	Status d1	[0] - off	[1] - on

Status 2:

Bit 15...6 :	0			
Bit 5	:	Status d12	[0] - off	[1] - on
Bit 4	:	Status d11	[0] - off	[1] - on
Bit 3	:	Status d10	[0] - off	[1] - on
Bit 2	:	Status d9	[0] - off	[1] - on
Bit 1	:	Status d8	[0] - off	[1] - on
Bit 0 (LSB)	:	Status d7	[0] - off	[1] - on

4.2.3 Example

Writing of second to fourth value (3, 4, 5) in floating point format at L1WRIT function at block number 8:

=> Writing of three values to instrument with address 1 at modbus address 0x8000 + 2*(0x50 + 7*0x9)

Structure of send message (Hex representation)::

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*Anz*	*Value = 3.0*				*Value = 4.0*				*Value*
01	10	81	1E	00	06	0C	40	40	00	00	40	80	00	00	40

= 5.0			*CrcH*	*CrcL*
A0	00	00	xx	xx

Structure of response message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*CrcH*	*CrcL*
01	10	81	1E	00	06	yy	yy

4.3 MBDATA-Structures

4.3.1 Definitions

The address range 0x0110...0x015F is used by 5 function blocks MBDATA. These functions are new and permit access each to 16 free configurable parameters of the engineering similar to VPARA. The parameters can be changed only via interface, not via inputs or any operation of the functions. They can be set in the engineering at block numbers 56...60. Block number 56 starts with modbus address . The other block numbers each 0x0010 addresses subsequent.

Specification:

Analoge inputs:	no
Digitale inputs:	no
Analoge outputs:	16, Values of the configured parameter or 0
Digitale outputs:	no
Float parameter:	no
Int parameter:	no
Float configurations:	no
Int configurations:	32, for each of the 16 data information of block number and parameter number (first counting the integer then the floating point parameter).
Texts:	1, Titel
Block number range:	56...60
Time group assignement:	all

4.3.2 Example

Reading the values of the 4th to 6th parameters, configured for access via MBDATA function at block number 59, in floating point format

=> Reading of 3 values in floating point format (6 words) from instrument with address 1 starting with modbus address 0x8000 + 2*(0x110 + 3*0x10 + 3) = 0x8286

Structure of request message (Hex representation):

Adr	Mode	ModH	ModL	AnzH	AnzL	CrcH	CrcL
01	03	82	86	00	06	xx	xx

Structure of response message (Hex representation):

Adr	Mode	Anz	Value = 3.0				Value = 4.0				Value = 5.0				CrcH	CrcL
01	03	0C	40	40	00	00	40	80	00	00	40	A0	00	00	yy	yy

4.4 Modbus addresses for controller

For process data of 30 controller function blocks (CONTR / CONTR+ / PIDMA) modbus addresses are reserved. The controller base modbus addresses 1...30 define the start addresses of 30 ranges with data structures for controllers. The total modbus adress range comprises 0x0160....0x0687 The modbus addresses of the single process data are calculated via the offsets within the structure that are given in the definition described below.
The sequence of the controller function blocks sets the range number.

Calculating of controller base address = 0x0160 + (range number – 1) * 0x2C

4.4.1 Structure for CONTR, CONTR+, PIDMA

Offset	Data	Access	Range
0x00	Status 1	R	0 ... 63
0x01	Status 2	R	0 ... 63
0x02	Status 3 (not PIDMA)	R	0 ... 63
0x03	Setpoint status	R	0 ... 63
0x04	Status Tuning 1	R	0 ... 63

Status 1:

Bit 15...6	:	0		
Bit 5	:	Sensor fail	[0] - no	[1] - yes
Bit 4	:	Controller switched off	[0] - no	[1] - yes
Bit 3	:	Y/Y2 switch over	[0] - Y	[1] - Y2
Bit 2	:	Auto/manual	[0] - Auto	[1] - manual
Bit 1	:	Switching output 2	[0] - off	[1] - on
Bit 0 (LSB)	:	Switching output 1	[0] - off	[1] - on

Status 2:

Bit 15...6	:	0		
Bit 5		Status controller	[0] - ok	[1] - not ok
Bit 4	:	Status PI/P	[0] - PI	[1] - P (not PIDMA)
Bit 3	:	0		
Bit 2	:	0		
Bit 1	:	0		
Bit 0 (LSB)	:	0		

Status 3: (not PIDMA)

Bit 15...3	:	0		
Bit 2	:	Override control+	[0] - off	[1] - on
Bit 1	:	Override control-	[0] - off	[1] - on
Bit 0 (LSB)	:	0		

Setpoint status:

Bit 15...5	:	0		
Bit 4	:	Tracking	[0] - off	[1] - on
Bit 3	:	Setpoint gradient suppressed	[0] - no	[1] - yes
Bit 2	:	Weff frozen	[0] - no	[1] - yes
Bit 1	:	Wext/Wint switch-over	[0] - Wext	[1] - Wint
Bit 0 (LSB)	:	w/W2 switch-over	[0] - w	[1] - W2

Status Tuning:

Bit 15...3	:	0		
Bit 2	:	Self-tuning result	[0] - Ok	[1] - error
Bit 1	:	Self-tuning operation	[0] - off	[1] - on
Bit 0 (LSB)	:	Process at rest	[0] - no	[1] - yes (not PIDMA)

Offset	Data	Access	Range
0x07	Additional correcting value on/off	R/W	0 / 1
0x08	PI/P switch-over (not PIDMA)	R/W	0 / 1
0x09	Auto/manual switch-over	R/W	0 / 1
0x0A	Self-tuning start	R/W	0 / 1
0x0B	Wext/Wint switch-over	R/W	0 / 1
0x0C	w/W2 switch-over	R/W	0 / 1
0x0D	Controller on/off	R/W	0 / 1

Offset	Data	Access	Range
0x10	Eff. set-point	R	-29999 ... 200000
0x11	Eff. process value	R	-29999 ... 200000
0x12	Effective correcting variable	R	-29999 ... 200000
0x13	Control deviation	R	-29999 ... 200000
0x14	Main variable 1	R	-29999 ... 200000
0x15	Auxiliary variable 2	R	-29999 ... 200000
0x16	Auxiliary variable 3	R	-29999 ... 200000
0x17	Position feedback	R	-29999 ... 200000
0x18	Override control + (not PIDMA)	R	-29999 ... 200000
0x19	Override control - (not PIDMA)	R	-29999 ... 200000
0x1A	Ext. set-point	R	-29999 ... 200000
0x1B	internal set-point, non volatile (EEPROM)	R/W	-29999 ... 200000
0x1C	internal set-point, volatile (RAM)	R/W	-29999 ... 200000
0x1D	Difference correcting variable	R/W	-210 ... 210
0x1E	Absolute correcting variable	R/W	-105 ... 105
0x1F	Effective parameter set number (not PIDMA)	R/W	1 ... 6
0x20	Parameter set for self-tuning	R	1 ... 6
0x21	Delay time heating (not PIDMA)	R	0 ... 200000
0x22	Rate of change heating (not PIDMA)	R	0 ... 9.999
0x23	Process gain heating (not PIDMA)	R	0 ... 9.999
0x24	Error code of self-tuning heating (not PIDMA)	R	0 ... 8
0x25	Delay time cooling (not PIDMA)	R	0 ... 200000
0x26	Rate of change cooling (not PIDMA)	R	0 ... 9.999
0x27	Process gain cooling (not PIDMA)	R	0 ... 9.999
0x28	Error code of self-tuning cooling (not PIDMA)	R	0 ... 8

4.4.2 Example

Writing of internal setpoint = 100 at 2nd controller function block in floating point format

=> Writing of 1 value in floating point format (2 words) at instrument with address 1 starting with modbus address 0x8000 + 2*(0x160 + 0x2C + 0x1C) = 0x8350

Structure of send message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*Anz*	*Value = 1 0 0 . 0*				*CrcH*	*CrcL*
01	10	83	50	00	02	04	42	C8	00	00	xx	xx

Structure of response message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*CrcH*	*CrcL*
01	10	81	1E	00	02	yy	yy

4.5 Modbus addresses for programmer

For process data of 40 programmer function blocks (APROG / DPROG) modbus addresses are reserved.
The programmer base modbus addresses 1...40 define the start addresses of 40 ranges with data structures for programmers. The total modbus adress range comprises 0x0688....0x09F7. The modbus addresses of the single process data are calculated via the offsets within the structure that are given in the definition described below.
The sequence of the programmer function blocks sets the range number.

Calculating of programmer base address = 0x0688 + (range number – 1) * 0x16

4.5.1 Structure for APROG and DPROG

Offset	Data	Access	Range
0x00	Status 1	R	0 ... 63
0x01	Status 2	R	0 ... 63
0x02	Status 3 (only DPROG)	R	0 ... 63
0x03	eff. program number	R	1 ... 99
0x04	Program time net	R	0 ... 959999
0x05	Program time gross	R	0 ... 959999
0x06	Programmer set-point (only APROG)	R	-29999 ... 999999
0x07	Rest time programmer	R	0 ... 959999
0x08	End value active segment (only APROG)	R	-29999 ... 999999
0x09	Segment number	R	1 ... 999
0x0A	Rest time segment	R	0 ... 959999

Status 1:

Bit 15...5	:	0		
Bit 4	:	Err2	[0] - no.	[1] - yes
Bit 3	:	Err1	[0] - no	[1] - yes
Bit 2	:	Program reset	[0] - off	[1] - on
Bit 1	:	Program end	[0] - no	[1] - yes
Bit 0 (LSB)	:	Program run	[0] - Stop	[1] - running

Err1: Faulty parameter block_
Err2: Infinite loop with parameter blocks

Status 2: Actual status of control outputs (only DPROG)

Bit 15...6	:	0		
Bit 5	:	Control output 6	0] - off	[1] - on
Bit 4	:	Control output 5	[0] - off	[1] - on
Bit 3	:	Control output 4	[0] - off	[1] - on
Bit 2	:	Control output 3	[0] - off	[1] - on
Bit 1	:	Control output 2	[0] - off	[1] - on
Bit 0 (LSB)	:	Control output 1	[0] - off	[1] - on

Status 2: Actual Status (only APROG)
Status 3: Actual Status (only DPROG)

Bit 15...2	:	0		
Bit 1	:	Prog-Manual	[0] - Auto	[1] - Manual
Bit 0 (LSB)	:	Prog-Halt	[0] - no Stop	[1] - Stop

Offset	Data	Access	Range
0x0C	Program stop / run	R/W	0 / 1
0x0D	Program continue / reset	R/W	0 / 1
0x0E	Start program search run (only APROG)	R/W	0 / 1
0x0F	F-key function (A/M swich over)	R/W	0 / 1
0x10	Program Auto / Manual	R/W	0 / 1

Offset	Data	Access	Range
0x12	Program number effective	R/W	1 ... 99
0x13	Program preset value Pmode = Seg Pmode = time	R/W	1...999 0...59999
0x14	Setpoint (in manual operation)	R/W	-29999 ... 999999 / 000000 ... 111111

4.5.2 Example

Reading of program time net (0x04), program time gross (0x05), programmer setpoint (0x06) and rest time programmer (0x07) of 15th programmer function block in floating point format.

=> Reading of 4 values in floating point format (8 words) from instrument with address 1 starting at modbus address 0x8000 + 2*(0x0688 + 14*0x16 + 0x04) = 0x87C0

Structure of request message (Hex representation):

Adr	*Mode*	*ModH*	*ModL*	*AnzH*	*AnzL*	*CrcH*	*CrcL*
01	03	87	C0	00	08	xx	xx

Structure of response message (Hex representation):

Adr	*Mode*	*Anz*	*Value = 20.0*				*Value = 20.0*				*Value = 100.0*				*Value*	
01	03	10	41	A0	00	00	41	A0	00	00	42	C8	00	00	42	20

= 40.0		*CrcH*	*CrcL*
00	00	yy	yy

5 'B'-key messages

To transmit the codes B1...B4 from the function block protocol of the ISO1745 mode, the modbus function code 23 (0x17) is used. It offers a combined write / read message. This message type should be used in general only from the engineering tool ET/KS98plus.
The structure of those message is:

Sending:

Address	Function code	Read address		Number of read data		Write address		Number of write data in words		in bytes	Write data 1...n				CRC	
	0x17	0x0	0x0			0x0	0x0									

Answer:

Address	Function Code	Number of bytes	Read data 1 .... n					CRC	
	0x17								

This message type is only used to transmit the consisting ISO1745 function block protocols within the write or read data. Therefore the 'read address' and the 'write address' are without interest and are ignored. The ranges 'write data' and 'read data' contain always ASCII values in contrast to other ranges..

While writing data the 'Number of read data' = 1. The 'Write data' contain the function block protocol according to the following definitions, that is the information, where to transmit which data. The 'Number of write data' indicates how much words / bytes of data are included in range 'Write data'. The answer contains as 'Number of bytes' the value 2 and 2 'Read data' with the value 0.

While reading data the 'Number of read data' ≠ 0. The exact value is don't care, because the function block protocol included in the 'Write data' contains the information, which data shall be read and from where. The 'Number of write data' indicates how much words / bytes of data are included in range 'Write data'. The answer contains as 'Number of bytes' the length of 'Read data'. 'Read data' contains the answer structure according to the following definitions.

5.1 Function block protocol for I/O- and VTREND-Data: Code B1

I/O data of a function block and data of VTREND can be read via B1 access.

Reading of data:

Structure of 'Write data' for request:

B	1	,	fb nr	,	fkt nr

Structure of 'Read data' for answer:

B	1	,	fb nr	,	fkt nr	=	typ nr	,	no real	,

real 1	,	...	,	real n	,	no int	,	int 1	,

max. number of data:	Reals:	25	0	Integers
	Reals	0	38	Integers

with VTREND always 25 real values

fb_nr:	1 ... 450	Function block number
fkt_nr:	0	Input data
	1	Output data
	80-84	Trend data range 1 to 5
typ_nr:	1...127	Number of function type
no_real:	0	no real values
	1 ... 25	Number of transmitted real values
no_int:	0	no integer values
	1 ... 38	Number of transmitted integer values

5.2 Function block protocol for parameter: Code B2

This access permits reading and writing of parameters in groups. Parameter are permanent stored data (online).

Reading of data:

Structure of 'Write data' for request:

B	2	,	fb_nr	,	fkt_nr

Structure of 'Read data' for answer::

B	2	,	fb nr	,	fkt nr	=	typ nr	,	no real	,

real 1	,	...	,	real n	,	no int	,	int 1	,

...	,	int n

Writing of data:

Structure of 'Write data' for request::

B	2	,	fb nr	,	fkt nr	=	typ nr	,	no real	,

real 1	,	...	,	real n	,	no int	,	int 1	,

...	,	int n

Structure of 'Read data' for answer:
No Read data!

max. number of data:	Reals:	25	0	Integers
	Reals	0	38	Integers

fb_nr:	0 1 ... 450	Instrument Function block number
fkt_nr:	0 1-9 1-10	General other Functions, if necessary
typ_nr:	1...127	Number of function types
no_real:	0 1 ... 25	no real values Number of transmitted real values
no_int:	0 1 ... 38	no integer values Number of transmitted integer values

5.3 Function block protocol for display texts: Code B2

This access permits reading and writing of display texts in groups. Parameter are permanent stored data (online). A text contains always 16 characters. The data type CHAR[n] is a special function and is handeled as integer value.

Reading of data:

Structure of 'Write data' for request::

B	2	,	fb_nr	,	8	0

Structure of 'Read data' for answer::

B	2	,	fb_nr	,	8	0	=	typ_nr	,	0	,

Writing of data:

Structure of 'Write data' for request::

B	2	,	fb_nr	,	8	0	=	typ_nr	,	0	,

no_text	,	text 1	,	...	,	text n

...	,	int n

Structure of 'Read data' for answer:
No Read data!

max. number of texts:	13 Texts	
fb_nr:	1 ... 450	Function block
typ_nr:	1...127	Number of function type
no_text:	0 1 ... 13	no texts Number of transmitted texts

5.4 Function block protocol for configuration data: Code B3

This access permits reading and writing of configurations in groups. Configurations can be stored permanently only in offline mode.

Reading of data:

Structure of 'Write data' for request::

B	2	,	fb nr	,	8	0

Structure of 'Read data' for answer:

B	2	,	fb_nr	,	8	0	=	typ_nr	,	0	,

Writing of data:

Structure of 'Write data' for request:

B	2	,	fb nr	,	8	0	=	typ nr	,	0	,

no text	,	text 1	,	...	,	text n

...	,	int n

Structure of 'Read data' for answer:
No Read data!

max. number of data:	Reals:	25	0	Integers
	Reals	0	38	Integers

fb_nr:	0	Instrument
	1 ... 450	Function block number
fkt_nr:	0	General
	1-9	othe functions, if necessary
	1-10	
typ_nr:	1...127	Number of function type
no_real:	0	no real values
	1 ... 25	Number of transmitted real values
no_int:	0	no integer values
	1 ... 38	Number of transmitted integer values

To write data via B3 key, the instrument has to be set in configuration mode before. The new configuration and parameter data become active after switching back to online mode.

9499-040-88711

P.O.B. 310 229, D-34058 Kassel, Germany
Printed in Germany 9499-040-88711 (02/2008)

A4, unibind, SW-Druck, 80g weiß