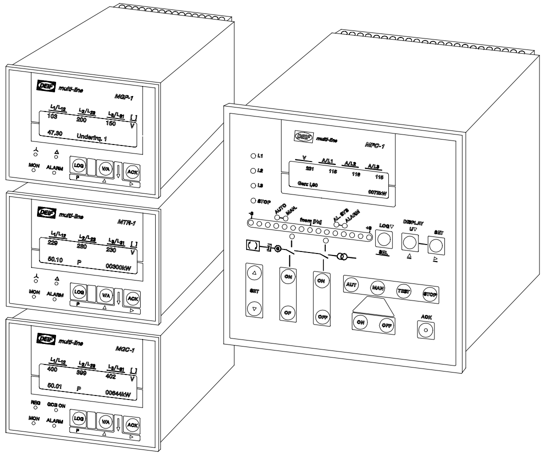
User's manual

DEIF

Serial Interface, ALL multi-line TYPES

multi-line

4189340028J



- RS 485: Modbus RTU
- RS 232: Siemens 3964 with RK-512 interpreter
- Fixed telegrams

DEIF A/S
Frisenborgvej 33, DK-7800 Skive
Denmark

Tel: (+45) 9614 9614
Fax: (+45) 9614 9615
E-mail: deif@deif.com

DEIF A/S

List of contents

This manual describes the standard serial interface to the *multi-line* products. It is however possible to adapt the serial interface to other protocols. Contact DEIF A/S for price and availability.

1. Serial interfaces for *multi-line*

1.1 Application and functional summary

Most *multi-line* products are equipped with a serial interface as a standard or an option. This serial interface can be either uni-directional or bi-directional. Uni-directional means that measured values can be sent from the *multi-line* product to a supervising system. Bi-directional means that control and set-up commands can be sent from the supervising system to the *multi-line* product together with the passing of measured values from the *multi-line* product to the supervising system.

At this moment two different protocols are implemented as a standard in multi-line: Modbus RTU for RS-485 multidrop communication and Siemens 3964 for RS-232 point to point communication. This manual describes how to interface equipment to these standard protocols.

The RS 232 serial interface is based on the Siemens 3964 protocol standard with RK512 interpreter. This is a very widely used protocol that is easy to code into the software of the receiving product. The Siemens 3964 protocol gives an easy and fault tolerant interface for the transmission speeds required.

The Siemens 3964 interface is described in hardware level, protocol level and telegram level to present the information needed when coding a receiving interface for *multi-line* serial interface. Telegram, protocol and hardware level is illustrated in the figure below where a flow of data is described for a unidirectional interface.

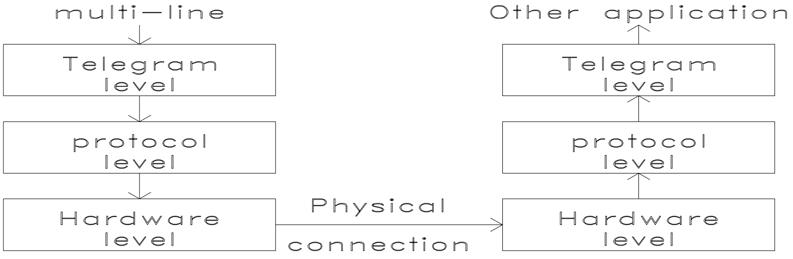


Fig. 1 Flow of data for a unidirectional interface

When a set of data is ready to send from *multi-line* they are packed into the right format in a telegram together with a telegram header. After this the telegram is sent to the procedure level where a connection to the procedure level in the other end is established through the hardware. The procedure level handles failures, timeouts, transmission retry etc. When a transmission is finished with no errors the telegram is handled to the telegram level in the receiving end for translating and presentation to the receiving system.

The RS 485 is based on the Modbus RTU protocol standard, using the *multi-line* units as slave nodes in the 2-wire multidrop network. The slave address can be set via the display of the *multi-line* unit to any number in the range 1 to 99.

The Modbus protocol works straight on the principle that the master sends a request. After this the slave must send an answer. Therefore the Modbus protocol contains no protocol level. The telegrams are sent directly to the hardware and a protocol level is therefore not described for the Modbus protocol.

Only parts of the Siemens 3964 or the Modbus protocol are used in the *multi-line* serial interface. The full description of the protocols can be found in the relevant Siemens or Modbus literature. The information in this document is also taken from this literature. All information about option disposal, order information and hardware connections to the serial connection on the individual products can be found in the specific product user manuals.

2. Telegram level, Siemens 3964

The telegrams are defined by the Siemens RK-512 interpretation. Only 2 basic telegram types exist: Command and reply telegrams. In the communication with *multi-line* the *multi-line* unit is always sending a command telegram, after this the connected unit sends a reply telegram. Command telegrams from *multi-line* can either be a send telegram with data where the receiving unit sends back an empty reply telegram, or the command telegram can be a fetch telegram (only bi-directional communication) where the reply telegram contains the requested data.

The two types of communication are shown below:

Send telegram:

=== Send telegram (Telegram header + data) ===>

<=== Reply telegram without data ===

Fetch telegram:

=== Fetch telegram (Telegram header) ===>

<=== Reply telegram with data ===

The *multi-line* will send a Send and a Fetch telegram each 1.5 second. In case the communication is uni-directional the *multi-line* will send only a Send telegram each 1.5 seconds.

2.1 The Command telegram

A command telegram consists of a 10-byte header and a data field with up to 128 bytes of data. The structure of the telegram header (Send or Fetch) is as follows:

Byte No.	Content (hex)	Remarks
1	00	Telegram identifier
2	00	Telegram identifier
3	41 for Send 45 for Fetch	Command
4	44	Type (data blocks)
5 6	0A00	Destination or source address (fixed) high byte send first
7 8	XXXX	Length of data block in words. Length is defined for each of the *multi-line* telegrams. High byte send first
9 10	FFFF	Co-ordination flag and CPU number (No co-ordination flag, no CPU number)

After the telegram header follows the data bytes in case on a Send telegram. When transferring word information as 2 bytes the high byte will be transferred first. When transferring double word information as two words, the high word will be transferred first. All data are in two's complement to transfer polarity information.

If the command telegram is a Fetch telegram there are no data bytes attached to the telegram header.

After a command telegram is sent, *multi-line* expects a reply telegram within 5 seconds.

2.2 The Reply telegram

A reply telegram consists of a 4-byte header and a datafield with up to 128 bytes of data. The structure of the telegram header contains information about the processing of the command and is built as follows:

Byte No.	Content (hex)	Remarks
1	00	Telegram identifier
2	00	Telegram identifier
3	00	
4	00	Error number. M*ulti-line* will NOT recognize different errors signalled in this field. All errors from the communication partner must be signalled on the protocol level with a "NAK" character

After the telegram header follows the data bytes in case it was a Fetch telegram that initiated the reply telegram. When transferring word information as 2 bytes the high byte will be transferred first. When transferring double word information as two words, the high word will be transferred first. All data are in two's complement to transfer polarity information.

2.3 Send telegrams for the products MTR-1, MGP-1 and MGC-1

Various telegrams are used in the products MTR-1, MGP-1 and MGC-1 to transfer measurement, alarm and state data from the *multi-line* product to another system. Telegram details see paragraph 6.

The different telegram versions can be recognised by the unit identifier in the beginning of the telegram.

- Telegrams from products with software version 1.0x have unit identifier 0300_D with a telegram length 14_H words.
- Telegrams from products with software version 1.1x have unit identifier 0301_D with a telegram length $1A_H$ words.
- Telegrams from products with software version 1.2x and up have unit identifier 0302_D with a telegram length $1B_H$ words.

2.4 Fetch telegram for the product MGC-1

The Fetch telegram is used in the product MGC-1 to receive set-point and control data from another system to MGC-1. See telegram details in paragraph 7. The data field length in the fetch telegram is set to $0C_H$ bytes.

2.5 Send telegram for the product MPC-1

The Send telegram is used in the product MPC-1 to transfer measurement data from the *multi-line* product to another system. Telegram details see paragraph 8.
The data field length is set to $5A_H$ bytes.

2.6 Fetch telegram for the MPC-1 (Reply on fetch)

The Fetch telegram is used in the product MPC-1 to receive set-point and control data from another system to MPC-1. Telegram details see paragraph 9.
The data field length in the fetch telegram is set to 14_H bytes.

3. Protocol level

The protocol level takes the defined telegrams and transmits them to the receiver. The protocol level will handle all software handshaking procedures and error handling as timeout errors, parity errors, and data failure errors.

3.1 Examples of Send and Fetch telegrams handled by the 3964 protocol

An example of a command (Send) telegram is shown on the next page. Please note the data marked with * are added by the protocol level as software handshaking. The example shows a successful Send telegram with the corresponding reply telegram.

	multi-line				**Other equipment**
*	-----	STX	02_H	--->	Start character
*	<---	DLE	10_H	-----	Pos. acknowledge
	-----	1st Byte	00_H	--->	Telegram header
	-----	2nd Byte	00_H	--->	Telegram header
	-----	3rd Byte	41_H	--->	Telegram header (Send)

•

•

•

	-----	9th Byte	FF$_H$	--->	Telegram header
	-----	10th Byte	FF$_H$	--->	Telegram header
	-----	11th Byte		--->	1st Data byte
	-----	12th Byte		--->	2nd Data byte
				●	
				●	
				●	
	-----	n+10th Byte		--->	nth data byte (last data byte)
*	<---	DLE	10$_H$	-----	End identifier
*	-----	ETX	03$_H$	--->	End identifier
*	-----	DLE	10$_H$	--->	Pos. acknowledge

Within 5 seconds a reply telegram must be started

*	<---	STX	02$_H$	-----	Start character
*	-----	DLE	10$_H$	--->	Pos. acknowledge
	<---	1st Byte	00$_H$	-----	Telegram header
	<---	2nd Byte	00$_H$	-----	Telegram header
	<---	3rd Byte	00$_H$	-----	Telegram header
	<---	4th Byte	00$_H$	-----	Telegram header (error no.)
*	<---	DLE	10$_H$	--->	End identifier
*	-----	ETX	03$_H$	--->	End identifier
*	-----	DLE	10$_H$	--->	Pos. Acknowledge

An example of a command (Fetch) telegram is shown below. Please note the data marked with * are added by the protocol level as software handshaking. The example shows a successful Fetch telegram with corresponding reply telegram that includes data.

	multi-line				Other equipment
*	-----	STX	02$_H$	--->	Start character
*	<---	DLE	10$_H$	-----	Pos. Acknowledge
	-----	1st Byte	00$_H$	--->	Telegram header
	-----	2nd Byte	00$_H$	--->	Telegram header
	-----	3rd Byte	45$_H$	--->	Telgram header (Fetch)
				●	
				●	
				●	
	-----	9th Byte	FF$_H$	--->	Telegram header
	-----	10th Byte	FF$_H$	--->	Telegram header
*	-----	DLE	10$_H$	--->	End identifier
*	-----	ETX	03$_H$	--->	End identifier
*	<---	DLE	10$_H$	-----	Pos. Acknowledge

Within 5 seconds a reply telegram must be started

*	<---	STX	02$_H$	-----	Start character
*	-----	DLE	10$_H$	--->	Pos. Acknowledge
	<---	1st Byte	00$_H$	-----	Telegram header
	<---	2nd Byte	00$_H$	-----	Telegram header
	<---	3rd Byte	00$_H$	-----	Telegram header
	<---	4th Byte	00$_H$	-----	Telegram header (error

					no.)
	<---	5th Byte		-----	1st data byte
	<---	6 th Byte		-----	2nd data byte
				● ● ●	
	<---	n+4th Byte		-----	nth data byte (last data byte)
*	<---	DLE	10_H	-----	End identifier
*	<---	ETX	03_H	-----	End identifier
*	-----	DLE	10_H	--->	Pos. Acknowledge

3.2 Building a send procedure for the 3964 protocol

In the following a description on how to build a 3964 protocol handler is given. The description can be used to program a 3964 protocol handler to a system that should communicate with *multi-line* products. Several libraries that support 3964 protocol with RK-512 interpreter exist. It is recommended to use an existing and tested communication software library to build an application that can communicate with *multi-line*.

To establish the link, the 3964 procedure sends the control character STX. If the partner replies within the time allowed of 550 msec. (i.e. no QVZ = timeout) the procedure starts the transmission. If the partner answers with NAK (15_H), any other character (except DLE) or if there is a timeout, then the attempt to establish a link has failed. After a total of 6 unsuccessful attempts, the procedure gives up, signals the telegram level that establishing the link has failed and sends the character NAK to the partner.

If the link is successfully established the information data in the current output buffer is sent to the partner. The partner monitors the time between the characters as it receives them. The time between two characters must not exceed the character delay time of 220 msec (ZVZ = character timeout).

Each DLE character found in the buffer is sent as two DLE characters (DLE doubling), i.e. the data (10_H) is sent twice.

After sending the contents of the buffer, the procedure adds the characters DLE and ETX as end identifier and waits for an acknowledgement. If the partner sends the character DLE within the time allowed (QVZ = 550 msec), the data field was received without errors. If the partner replies with NAK, any other character (except DLE), or if there is a timeout, the procedure starts again with the link establishment STX. After a total of 6 unsuccessful attempts to send the data field, the procedure breaks off the attempt, signals the error to the telegram level and sends NAK to the partner.

If the partner sends the character NAK while data is being transmitted, the procedure breaks off the transmission and repeats the transmission as previously described. If any other character is received, the procedure waits for the character delay time (ZVZ = 220 msec) and then sends NAK to reset the partner. The procedure then repeats the transmission as previously described.

3.3 Building a receive procedure for the 3964 protocol

In the idle state, i.e. when there is no send job to be processed, the 3964 procedure waits for the link to be established by the partner.

If the procedure receives any character (except STX) while in the idle state, it waits until the character delay time (ZVZ = 220 msec) and then sends NAK to reset the partner. The error is signalled to the telegram level.

If the procedure receives the character STX and if it has an empty input buffer available, it answers with DLE. The received characters are then entered in the input buffer. If two DLE characters are received in succession, only one character is entered in the input buffer. If the input buffer is full before the partner has started to terminate the link, the full buffer is transferred to the telegram level and the next characters arriving are entered in the second input buffer.

After each character received, the procedure waits for the next character to arrive within the character delay time. If there is a character timeout, a NAK character must be sent to reset the partner and the error is signalled to the telegram level.

If the procedure recognises the character sequence DLE ETX, it stops receiving and sends DLE to the partner to indicate that the field was received without errors. A NAK character is sent instead of DLE if errors are detected. The procedure transfers the content of the input buffer to the telegram level and returns to idle state.

If there is no empty input buffer available when the link is established with STX, a waiting time of 400 msec is started. If there is still, after this time has elapsed, no empty input buffer, an error is signalled to telegram level and a NAK character is sent, after this the procedure returns to idle state. If an input buffer becomes available in the 400 msec waiting time, the procedure sends the DLE character and receives the data as described above.

If transmission errors occur during the reception (characters lost, frame error, parity error) reception is continued until the link is terminated and the NAK is then sent to the partner as acknowledge. A repetition is to be expected. If the field cannot be received without errors after six attempts, or the repetition do not take place within 4 sec. the 3964 procedure stops receiving and signals the error to telegram level.

4. Telegram level, Modbus RTU

The telegrams are defined by the Modbus RTU protocol. Only 3 different types of commands are implemented in the *multi-line*:

- Read up to 10 words, Code 03_H or 04_H
- write 1 word, Code 06
- write up to 4 words, Code 10_H

As shown every command works on a block of data, so each command must be followed with an address to the data block of interest. In the Modbus protocol this address is always the word identifier shown in the telegram data list.

The following failure codes are implemented:

- Unknown command code, Failure Code 01_H
- Address failure, Failure Code 02_H
- Data failure, Failure Code 03_H

Multi-line cannot respond with other failure codes.

The data blocks in the Modbus RTU protocol are always built in the same way as it is shown in the explanation of the various commands:

- Slave number	1 Byte
- Function code	1 Byte
- Telegram text	N Byte
- CRC16 check	2 Bytes

The Modbus protocol always works on the principle that the master sends a request after this the slave must send an answer. The *multi-line* units are always slaves in the communication, and the command read N words therefore means reading N words from the *multi-line* product.

4.1 Reading N words (0<N≤10)

Function code: 03 or 04

Send:

SLAVE	FCT	ADH ADL	NB OF WORDS	CRCL CRCH

Receive:

SLAVE	FCT	NBYTE	…DATA…	CRCL CRCH

Data: |__|__| |__|__| ….

first second word

Example:

Reading 10 words at the address 0060_H from slave 1.

Send:
01 03 00 60 00 0A C5 D3

Receive:
01 03 14 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 A367

With 14H = 20 Bytes.

4.2 Writing 1 word

Function code: 06

Send:

SLAVE	FCT	ADH ADL	DATA DATA	CRCL CRCH

Receive:

SLAVE	FCT	ADH ADL	DATA DATA	CRCL CRCH

Data:

HI(Word)	LO(Word)

Example: Writing 1234_H at address 2000_H to slave 1.

Send:
01 06 20 00 12 34 8F 7D

Receive:
01 06 20 00 12 34 8F 7D

4.3 Writing N words (1<N≤4)

Function code: 10

Send:

SLAVE	FCT	ADH ADL	NB OF WORDS	NBYTE	..DATA..	CRCL CRCH

Data: | | | | | | ….

first   second   word

Receive:

SLAVE	FCT	ADH ADL	NBYTES*2	CRCL CRCH

Example:

Write: 1H at address 2000_H
2H at address 2001_H
3H at address 2002_H

Send:
01 10 20 00 0003 06 0001 0002 0003 91 41

Receive:
01 10 20 00 00 0C CB CC

4.4 Fail code

Send:

SLAVE	FCT v 80H	ERR	CRCL CRCH

ERR code 01 = Unknown function code
ERR code 02 = Address fail
ERR code 03 = Data fail

Example:
Address fail (ERR code 02) in a telegram of code 03_H (reading n words):

01 83 02 C0 C0

where $83_H = 80_H + 03_H$

4.5 Modbus telegram, MTR-1, MGC-1, MGP-1.

Various telegrams are used in the products MTR-1, MGP-1 and MGC-1 to transfer measurement, alarm and state data from the *multi-line* product to another system. Telegram details see paragraph 6 and 7.

The unit identifier in the beginning of the telegram can recognize the different telegram version.

- Telegrams from products with software version 1.0x have unit identifier 0300_D
- Telegrams from products with software version 1.1x have unit identifier 0301_D
- Telegrams from products with software version 1.2x and up have unit identifier 0302_D

5. Hardware level RS 232 / RS 485

The standard serial interfaces to the *multi-line* products are either RS-232 or RS-485. The connections to the interfaces can be found in the user manuals for the specific products together with the hardware specifications.

The interfaces work with V.24 voltage level and the idle line state is logical high.

The RS-232 interfaces include hardware handshaking with RTS and CTS lines. The hardware handshaking works with the following procedure: If the sender is ready with a new telegram and no transmission is currently in progress (CTS=OFF) RTS is set ON. The receiver gets the RTS signal and signals that it can receive the telegram by setting CTS=ON. When the sender receives the CTS signal it can start the transmission. If CTS signal is set OFF during transmission from the receiver the transmission is paused. The normal timeouts in the protocol prevents conflict by both units trying to transmit simultaneously.

5.1 RS 232 interface

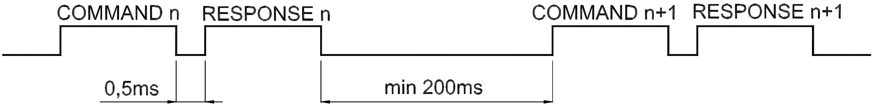
RS 232 interface works with the transmission parameters:

1 start bit
8 databit
1 (even) parity bit
1 stopbit
The transmission speed is fixed to 9600 baud.

5.2 RS 485 interface

RS 485 MOD bus RTU interface works with the transmission parameters:

Slave address 1 to 255
1 start bit
8 databit
no parity bit
1 stopbit
Transmission speed is fixed to 9600 baud.



Maximum time between characters within commands is 1 ms.

A new command is only recognised when the delay between commands is longer than 200 ms.

The response time in multi-line for the slave to answer is 0.5 ms.

6. Send telegrams for the products MTR-1, MGP-1 and MGC-1

Various telegrams are used in the products MTR-1, MGP-1 and MGC-1 to transfer measurement, alarm and state data from the *multi-line* product to another system. In telegram 302 the ID numbering has been shifted 1 place so ID numbering is starting from 1 instead of previously 0 in software version 1.49 and later. At the moment 3 different versions exist:

6.1 Unit identifier 300_D

Byte	ID	Content	Type
00 01	0	300_D	Unit identifier
02 03	1	U_{L1-L2}	0...9999 Generator voltage. Measured in [V]
04 05	2	U_{L2-L3}	0...9999 Generator voltage. Measured in [V]
06	3	U_{L3-L1}	0...9999 Generator voltage. Measured in [V]

07			
08 09	4	$U_{L1\text{-}N}$	0...9999 Generator voltage. Measured in [V]
10 11	5	$U_{L2\text{-}N}$	0...9999 Generator voltage. Measured in [V]
12 13	6	$U_{L3\text{-}N}$	0...9999 Generator voltage. Measured in [V]
14 15	7	F_{GEN}	0...7000 Generator Frequency. Measured in [Hz]:100
16 17	8	I_{L1}	0...9999 Generator Current. Measured in [A]
18 19	9	I_{L2}	0...9999 Generator Current. Measured in [A]
20 21	10	I_{L3}	0...9999 Generator Current. Measured in [A]
22 23	11	Cos-phi	-99...0...100 Generator cosinus-phi. Measured in cos-phi:100 Negative value means capacitive cos-phi
24 25	12	P_{GEN}	-9999...09999 Generator Active power. Measured in [W] Negative value means reverse power
26 27	13	Q_{GEN}	Generator Reactive power. Measured in [Var]. Positive value means generated inductive reactive power.
28 29	14	U_{BB}	Busbar voltage measured as $U_{L1\text{-}L2}$. Measured in [V]
30 31	15	F_{BB}	0...7000 Busbar Frequency. Measured in [Hz]:100
32	16	Exp. U_{GEN}	Exponent Generator voltage
33	16	Exp. I_{GEN}	Exponent Generator current
34	17	Exp. P/Q_{GEN}	Exponent Generator power/reactive power
35	17	Exp. U_{BB}	Exponent busbar voltage
36 37 38 39	18 19	E_{GEN}	Energy counter. Measured in [kWh]

See paragraph 6.4 for an explanation of the exponent system.

6.2 Unit identifier 301_D

Byte	ID	Content	Type
00 01	0	301_D	Unit identifier
02 03	1	$U_{L1\text{-}L2}$	0...9999 Generator voltage. Measured in [V]
04 05	2	$U_{L2\text{-}L3}$	0...9999 Generator voltage. Measured in [V]
06 07	3	$U_{L3\text{-}L1}$	0...9999 Generator voltage. Measured in [V]
08	4	$U_{L1\text{-}N}$	0...9999 Generator voltage. Measured in [V]

09			
10 11	5	$U_{L2\text{-}N}$	0...9999 Generator voltage. Measured in [V]
12 13	6	$U_{L3\text{-}N}$	0...9999 Generator voltage. Measured in [V]
14 15	7	F_{GEN}	0...7000 Generator Frequency. Measured in [Hz]:100
16 17	8	I_{L1}	0...9999 Generator Current. Measured in [A]
18 19	9	I_{L2}	0...9999 Generator Current. Measured in [A]
20 21	10	I_{L3}	0...9999 Generator Current. Measured in [A]
22 23	11	Cos-phi	-99...0...100 Generator cosinus-phi. Measured in cos-phi:100 Negative value means capacitive cos-phi
24 25	12	P_{GEN}	-9999...09999 Generator Active power. Measured in [W] Negative value means reverse power
26 27	13	Q_{GEN}	Generator Reactive power. Measured in [Var]. Positive value means generated inductive reactive power.
28 29	14	U_{BB}	Busbar voltage measured as $U_{L1\text{-}L2}$. Measured in [V]
30 31	15	F_{BB}	0...7000 Busbar Frequency. Measured in [Hz]:100
32	16	Exp. U_{GEN}	Exponent Generator voltage
33	16	Exp. I_{GEN}	Exponent Generator current
34	17	Exp. P/Q_{GEN}	Exponent Generator power/reactive power
35	17	Exp. U_{BB}	Exponent busbar voltage
36 37 38 39	18 19	E_{GEN}	Energy counter. Measured in [kWh]
40 41	20	Internal Errors 1	See internal error list paragraph 6.5
42 43	21	Internal Errors 2	See internal error list paragraph 6.5
44 45	22	Internal Errors 3	See internal error list paragraph 6.5
46 47	23	Internal Errors 4	See internal error list paragraph 6.5
48 49	24	Internal Errors 5	See internal error list paragraph 6.5
50 51	25	Internal Errors 6	See internal error list paragraph 6.5

See paragraph 6.4 for an explanation of the exponent system.

6.3 Unit identifier 302_D

Byte	ID	Content	Type
00 01	1	302_D	Unit identifier
02 03	2	U_{L1-L2}	0...9999 Generator voltage. Measured in [V]
04 05	3	U_{L2-L3}	0...9999 Generator voltage. Measured in [V]
06 07	4	U_{L3-L1}	0...9999 Generator voltage. Measured in [V]
08 09	5	U_{L1-N}	0...9999 Generator voltage. Measured in [V]
10 11	6	U_{L2-N}	0...9999 Generator voltage. Measured in [V]
12 13	7	U_{L3-N}	0...9999 Generator voltage. Measured in [V]
14 15	8	F_{GEN}	0...7000 Generator Frequency. Measured in [Hz]:100
16 17	9	I_{L1}	0...9999 Generator Current. Measured in [A]
18 19	10	I_{L2}	0...9999 Generator Current. Measured in [A]
20 21	11	I_{L3}	0...9999 Generator Current. Measured in [A]
22 23	12	Cos-phi	-99...0...100 Generator cosinus-phi. Measured in cos-phi:100 Negative value means capacitive cos-phi
24 25	13	P_{GEN}	-9999...09999 Generator Active power. Measured in [W] Negative value means reverse power
26 27	14	Q_{GEN}	Generator Reactive power. Measured in [Var]. Positive value means generated inductive reactive power.
28 29	15	U_{BB}	Busbar voltage measured as U_{L1-L2}. Measured in [V]
30 31	16	F_{BB}	0...7000 Busbar Frequency. Measured in [Hz]:100
32	17	Exp. U_{GEN}	Exponent Generator voltage
33	17	Exp. I_{GEN}	Exponent Generator current
34	18	Exp. P/Q_{GEN}	Exponent Generator power/reactive power
35	18	Exp. U_{BB}	Exponent busbar voltage
36 37 38 39	19 20	E_{GEN}	Energy counter. Measured in [kWh]
40 41	21	Internal Errors 1	See internal error list paragraph 6.5
42 43	22	Internal Errors 2	See internal error list paragraph 6.5

44 45	23	Internal Errors 3	See internal error list paragraph 6.5
46 47	24	Internal Errors 4	See internal error list paragraph 6.5
48 49	25	Internal Errors 5	See internal error list paragraph 6.5
50 51	26	Internal Errors 6	See internal error list paragraph 6.5
52 53	27	Operation Mode	See operation mode paragraph 6.6

See paragraph 6.4 for an explanation of the exponent system.

6.4 Building values from data and exponent

Examples of building values from the data above:

U_{L1-L2} = 103	Exponent = 2	$103*10^{2}$ = 10,3kV
I_{L1} = 80	Exponent = -1	$80*10^{-1}$ = 8,0A
P_{GEN} =123	Exponent = 4	$123*10^{4}$ = 1,23MW
f_{BB} = 5230		5230/100 = 52,30Hz

Format: Motorola 16/32 Bit (two´s complement). Low-order address = high Byte / high Word

6.5 Explanation of internal Errors

Internal errors 1

Bit	Value	Meaning
15/14	1/0	Overfrequency step 2
13/12	1/0	Underfrequency step 2
11/10	1/0	Overvoltage step 2
9/8	1/0	Undervoltage step 2
7/6	1/0	Unbalanced load
5/4	1/0	Overcurrent step 1
3/2	1/0	Overload
1/0	1/0	Reverse Power

As an example value 1/0 on bit 15/14 means that the overfrequency step 2 is activated.

Internal errors 2

Bit	Value	Meaning
15/14	1/0	Overfrequency step 1
13/12	1/0	Underfrequency step 1
11/10	1/0	Overvoltage step 1
9/8	1/0	Undervoltage step 1
7/6	1/0	Free
5/4	1/0	df/dt
3/2	1/0	Voltage Asymmetry
1/0	1/0	Vector shift

Internal errors 3

Bit	**Value**	**Meaning**
15/14	1/0	Free
13/12	1/0	Free
11/10	1/0	Reactive Power, inductive
9/8	1/0	Reactive Power, capacitive
7/6	1/0	Free
5/4	1/0	Free
3/2	1/0	Overcurrent step 2
1/0	1/0	Serial interface fault

Internal errors 4

Bit	**Value**	**Meaning**
15/14	1/0	Overfrequency Busbar
13/12	1/0	Underfrequency Busbar
11/10	1/0	Overvoltage Busbar
9/8	1/0	Undervoltage Busbar
7/6	1/0	Free
5/4	1/0	Free
3/2	1/0	Free
1/0	1/0	Free

Internal errors 5

Bit	**Value**	**Meaning**
15/14	1/0	Free
13/12	1/0	Free
11/10	1/0	Free
9/8	1/0	Free
7/6	1/0	Free
5/4	1/0	Free
3/2	1/0	Free
1/0	1/0	"Power Max" relay

Internal errors 6

Bit	Value	Meaning
15/14	1/0	Free
13/12	1/0	Free
11/10	1/0	Free
9/8	1/0	Free
7/6	1/0	Free
5/4	1/0	Free
3/2	1/0	Free
1/0	1/0	Free

6.6 Explanation of operation Mode (MGC-1 only)

Bit	Value	Mode	Terminal
15	1	Feedback Breaker "ON"	32
14	1	Start Sync/reg. "ON"	31
13	1	Mode 1 "ON”	74
12	1	Mode 0 "ON"	75
11	1	Free	
10	1	Free	
9	1	Free	
8	1	Free	
7	1	Free	
6	1	Free	
5	1	Free	
4	1	Synchronisation	
3	1	Shut down	
2	1	Automatic control "ON"	
1	1	Part Load Lead "ON"	
0	1	Free	

7. Fetch telegram for the product MGC-1 (Reply on fetch)

Only one telegram exists in the product MGC-1 to transfer remote operation data and regulator set-points from the other system to the *multi-line* product. This telegram has no unit identifier.

7.1 Telegram content

Byte	ID	Content	Type
00 01	1	Power regulator set-point	0...32,000 kW.
02 03	2	Cos-phi regulator set-point	-99...100 stated as cos-phi value/100. The cos-phi regulator is only active when the reactive power regulator is inactive (bit 14 in control command). A negative value means capacitive reactive power and a positive value means inductive reactive power. The value 100 means Cos-phi =1.
04 05	3	Control command	See control command list in paragraph 7.2.
06 07	4	Frequency regulator set-point	3,200...6,800 stated as frequency value/100.
08 09	5	Voltage regulator set-point	0...480 The voltage regulator set-point is related to the voltage on the connectors on MGC-1. This means that if a voltage transformer is applied, the set-point must be stated as the secondary voltage.
10 11	6	Reactive power regulator set-point	-32,000...32,000 kvar. A negative value means capacitive reactive power and a positive value means inductive reactive power. The reactive power regulator is only active in power control mode.

7.2 Coding of the Control command

Bit	Value	Mode
15	1	Inhibits all alarms except overfrequency and overvoltage. This function is only active if it is activated in display.
14	1	Activates reactive power controller in power regulation mode. If inactive the cos-phi controller will be active in power regulation mode.
13	1	Sets mode pin 1 (Same function as connection no.74)
12	1	Sets mode pin 0 (Same function as connection no.75)
11	1	Free
10	1	Free
9	1	Free
8	1	Starts synchronisation or regulation (Same function as connection no 31, start sync/reg)
7	1	Free
6	1	Free
5	1	Free
4	1	Acknowledge of alarms. Reacts only on a change from 0 to 1, which means that this bit must be set back to 0 after acknowledgement of an alarm.
3	0	Serial communication watchdog. At least every 15 sec. A “0” must be sent to this bit. Otherwise the serial communication alarm will be activated.
2	0	Serial communication watchdog. At least every 15 sec. A “0” must be sent to this bit. Otherwise the serial communication alarm will be activated.
1	1	Free
0	1	Free

8. Send telegrams for the product MPC-1

Only one telegram exists in the products MPC-1 to transfer measurement, alarm and state data from the *multi-line* product to another system.

8.1 Unit identifier 12_D

Byte	Content	Type
00 01	0012_D	Unit identifier
02 03	U_{L1-L2}	0...9999 Generator Voltage. Measured in [V]
04 05	U_{L2-L3}	0...9999 Generator Voltage. Measured in [V]
06 07	U_{L3-L1}	0...9999 Generator Voltage. Measured in [V]
08 09	f_{GEN}	0...7000 Generator frequency. Measured in [Hz]:100
10 11	I_{L1}	0...9999 Generator Current. Measured in [A]
12 13	I_{L2}	0...9999 Generator Current. Measured in [A]
14 15	I_{L3}	0...9999 Generator Current. Measured in [A]
16 17	cos-phi	-99...0...100 Generator cosinus-phi. Measured in cos-phi:100 Negative value means capacitive cos-phi
18 19	P_{GEN}	-9999...0…9999 Generator Active power. Measured in [kW] Negative value means reverse power
20 21	Q_{GEN}	-9999...0...9999 Generator Reactive power. Measured in [kvar] Negative value means reverse reactive power
22 23	U_{L1-L2}	0...9999 Busbar Voltage. Measured in [V]
24 25	f_{BB}	0...7000 Busbar Frequency. Measured in [Hz]:100
26 27	U_{L1-L2}	0...9999 Mains Voltage. Measured in [V]
28 29	U_{L2-L3}	0...9999 Mains Voltage. Measured in [V]
30 31	U_{L3-L1}	0...9999 Mains Voltage. Measured in [V]
32 33	F_{MAINS}	0...7000 Mains Frequency. Measured in [Hz]:100
34 35	I_{L1}	0...9999 Mains Current. Measured in [A] Positive value means current delivered. Negative value means current consumed
36 37	cos-phi	-99...0...100 Mains cosinus-phi. Measured in cos-phi:100 Negative value means capacitive cos-phi
38 39	P_{GEN}	-9999...0...9999 Mains active power. Measured in [kW] Negative value means import of power
40	Breaker	0000_H = Both breakers open

41	positions	0003_H = Generator closed, mains open 0300_H = Generator open, mains closed 0303_H = Generator closed, mains closed
42 43	Running hours	Measured in [hours]
44 45	Service hours	Measured in [hours]
46 47	Battery voltage	Voltage on the supply voltage input. Measured in [V]:10
48 49	Alarm field 1	See 8.2 for coding of alarm field 1
50 51	Alarm field 2	See 8.2 for coding of alarm field 2
52 53	Alarm field 3	See 8.2 for coding of alarm field 3
54 55	Alarm field 4	See 8.2 for coding of alarm field 4
56 57	Alarm field 5	See 8.2 for coding of alarm field 5
58 59	Alarm field 6	Spare alarm field
60 61	Alarm field 7	Spare alarm field
62 63	Running mode	See 8.2 for coding of the different running modes
64 65	Failure class	0000_H = No failure 0003_H = F1 (Common alarm) $000C_H$ = F2 (Soft stop) 0030_H = F3 (Shutdown) More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes)
66 67 68 69	E_{GEN}	Energy counter. Measured in [kWh]
70 71 72 73	Reactive energy	Reactive energy counter. Measured in [kvarh]
74 75	Analog input 1	
76 77	Analog input 2	
78 79	Analog input 3	
80 81	Analog input 4	
82	Analog input 5	

83		
84 85	Analog input 6	
86 87	Analog input 7	
88 89	Telegram counter	Incremented by 1 each time a new telegram is sent.

8.2 Alarm field and running modes coding

Alarm field 1 coding

More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes on bit level). 0000_H means no alarms.

Alarm	**Code (Hex)**
Analog In 1	0003_H
Analog In 2	$000C_H$
Analog In 3	0030_H
Analog In 4	$00C0_H$
Analog In 5	0300_H
Analog In 6	$0C00_H$
Analog In 7	3000_H
Pickup Gen. freq.	$C000_H$

Alarm field 2 coding

More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes on bit level). 0000_H means no alarms.

Alarm	**Code (Hex)**
Gen. overcurrent	0003_H
Sync. Failure	$000C_H$
Asymmetric load	0030_H
Start failure	$00C0_H$

Alarm field 3 Coding

More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes on bit level). 0000_H means no alarms.

Alarm	**Code (Hex)**
Mains voltage	0003_H
Mains frequency	$000C_H$
Generator voltage	0030_H
RPM	$00C0_H$
Reverse power	0300_H
Overload	$0C00_H$
Battery voltage	3000_H
Service	$C000_H$

Alarm field 4 Coding

More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes on bit level). 0000_H means no alarms.

Alarm	Code (Hex)
Terminal 34	0003_H
Terminal 35	$000C_H$
Terminal 36	0030_H
Terminal 61	$00C0_H$
Terminal 62	0300_H
Terminal 63	$0C00_H$
Terminal 64	3000_H
Terminal 65	$C000_H$

Alarm field 5 Coding

More failures can occur simultaneously, and the alarm codes are then added (logic OR for the given codes on bit level). 0000_H means no alarms.

Alarm	Code (Hex)
Terminal 66	0003_H
Terminal 67	$000C_H$
Terminal 68	0030_H
Terminal 69	$00C0_H$
Terminal 70	0300_H
Terminal 71	$0C00_H$
Terminal 72	3000_H
Terminal 73	$C000_H$

Running mode coding

Running mode	Code (Hex)
Auto. (Stand by)	$000C_H$
Stop	0030_H
Auto 2	$00C0_H$
Auto 1	0300_H
Test	$0C00_H$
Manual	3000_H
Sprinkler	$C000_H$

9. Fetch telegram for the MPC-1 (Reply on fetch)

Only one telegram exists in the product MPC-1 to transfer remote operation data and regulator set-points from the other system to the *multi-line* product. This telegram has no unit identifier.

9.1 Telegram content

Byte	Content	Type
00 01	Remote start	Code 00F0$_H$ sets the start signal Code 000F$_H$ resets the start signal
02 03	Remote stop	Code 00F0$_H$ sets the Stop signal Code 000F$_H$ resets the Stop signal
04 05	Power regulator set-point	See 9.2 for coding power regulator set-point
06 07	Cos-phi regulator set-point	Set-point for the generator cosinus-phi regulator. The value is set in cos-phi*100 (Same scaling as measured cos-phi) Negative value means capacitive cos-phi
08 09	Alarm acknowledge	Code 00F0$_H$ sets the acknowledge signal Code 000F$_H$ resets the acknowledge signal
10...19	Spare	

9.2 Coding of power regulator set-point

Active power set-point can be one of the following: Fixed value (F-value), delivery to mains (D-value) or consumption from mains (C-value) (see user's manual for MPC-1). The set-point is binary placed in bits 0...13. Value setting is in bits 14 and 15. The following applies:

Setting	Bit 15	Bit 14
F-value	0	1
D-value	0	0
C-value	1	1

Example: A fixed value of 150 kW is wanted. The signal is:
01/00 0000 1001 0110 B 4096 Hex
A D-value of 300 kW is wanted. The signal is:
00/00 0001 0010 1100 B 012C Hex
A C-value of 600 kW is wanted. The signal is:
11/11 1101 1010 1000 B FDA8 Hex.

Errors and changes excepted.