RS485 Protocol Structure

Contents

1 Introduction

1.1 The RS-485 Interface

Besides up to 31 sensors the bus contains a master which represents a PC or also a PLC (Programmable Logic Controller). Communication with the desired sensor on the bus is initiated by the master, upon which the sensor answers. Data can be sent by the master to the sensor (WRITE) or requested by it (READ). No data is sent by the sensors without being requested by the master.

A message to the sensor contains:
• Sensor address
• Type: read or write
• Index: which command is to be executed
• Data (if required)

An answer from the sensor contains:
• Sensor address
• Type of answer (e.g. acknowledge, error, busy ....)
• Data (if required)

1.2 Commissioning Procedure with RS485

1	Connect the sensor according to the connection diagram, note recommended cables
2	Connect all the components to the required topology
3	Supply the sensor with +24 VDC
4	Enable the sensor for RS485 commands via the index command 010 "RS485 lock"
5	Control the sensor via the index commands with RS485

Zum Hagenbach 7 • D-48366 Laer • Fon: +49 (0)2554/9130-00 • Fax: +49 (0)2554/9130-10 • info@welotec.com
www.welotec.com
3/25

2 Connection

2.1 Sensor Connection Diagram

+Vs and 0V are used for the power supply and RS485 uses Rx/Tx+ and Rx/Tx- for data exchange. These 4 pins are required for the operation of RS485.
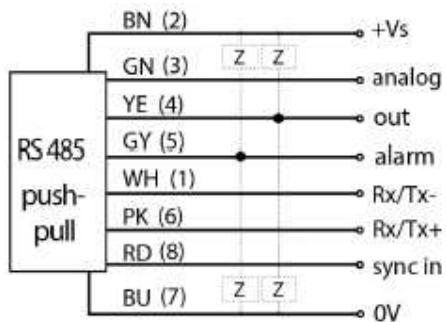

Example of an 8-pin connection diagram

3 Topology

There are two different recommended network structures depending on the sensors RS485 interface and the needed data transfer rate.

3.1 ≤ 115 kbit/s: Topology for Welotec standard RS485 interfaces[1]

This topology applies to most applications and sensor types with Welotec standard RS485 interface ≤115 kbit/s. Simple wiring and lower demands on the materials are the advantages.

Requirements:

- Shielded cables necessary
- Maximum 10m length from sensor to master
- At maximum 15 sensors[2] can be integrated in such a network
- Data transfer rates up to 115 kbit/s for RS485
- In order to define the resistance level when no transmitter is active, the master must have failsafe bias resistors R_B
- The topology for Welotec fast RS485 interfaces (≤ 3 Mbit/s) is recommended for up to 31 sensors in the same bus or for other cable lengths

Sensor directly connected to Master

Several sensors connected to master

---
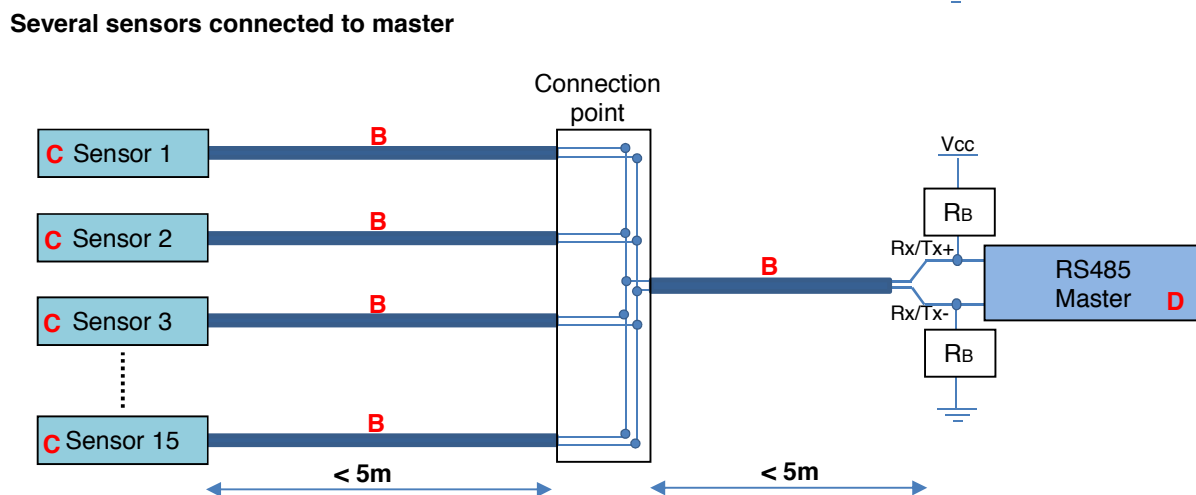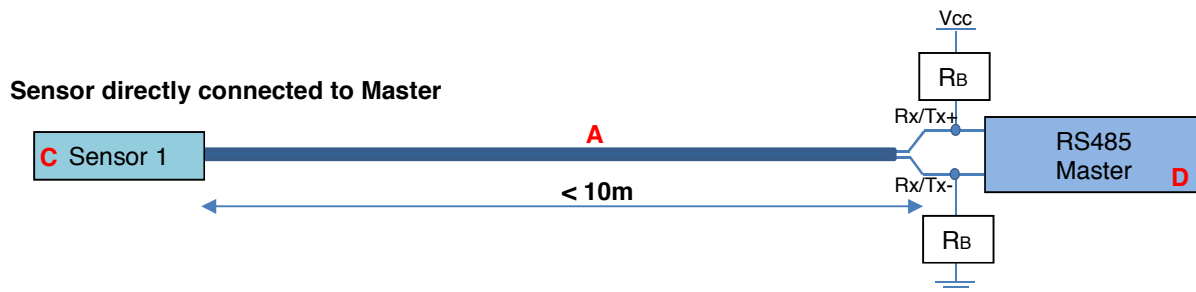
[1] According to the data sheet in the sensors operating manual
[2] Tested and released by Welotec

A **Cable <10m**
Maximum 10m length from sensor to master

C **Sensor**
Sensor with Welotec standard RS485 interface. Data transfer rates up to 115 kbit/s.

B **Cable <5m**
Maximum 5m length to the connection point

D **RS485 Master**
The PC/PLC (Programmable Logic Controller) controls the devices as master. The master must have two Failsafe bias resistors R_B (300 Ohm - 5 kOhm, Pull-Up for Rx/Tx+ and Pull-Down for Rx/Tx- resistors).

3.2 ≤ 3 Mbit/s: Topology for Welotec fast RS485 interfaces[3]

The defined topology and also high performance cables are required for sensors with Welotec fast RS485 interfaces with data transfer rates up to 3Mbit/s. Each device is connected in series to a single bus cable, on both sides of the bus cable is a terminating resistor required. At maximum 31 sensors can be integrated in such a network
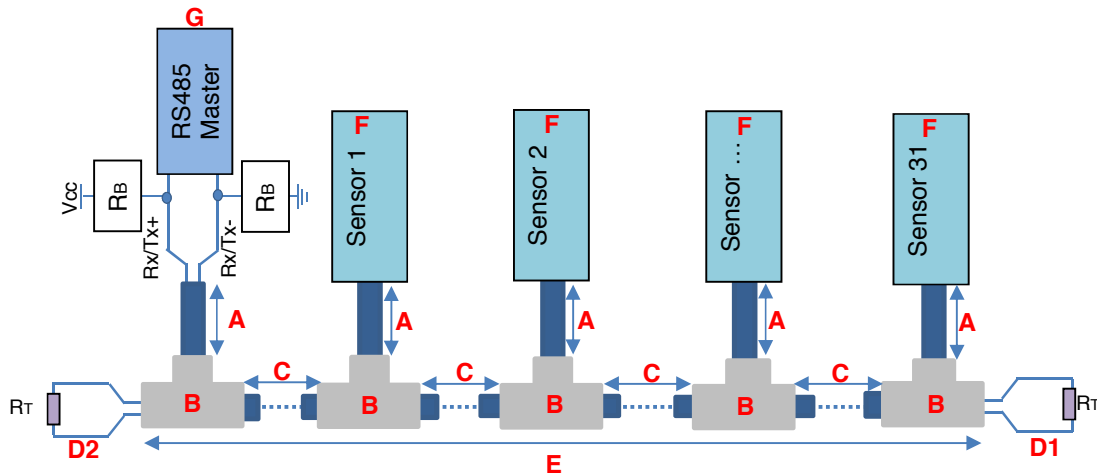
Requirements:
- Both ends of the bus cable must be terminated with terminating resistors R_T, value = 120 Ohm
- The terminating resistor can be integrated in the master or in the sensor
- The wave resistance of the bus cable must be 120 Ohm
- The stub line length has to be shorter than 0.3m (length = A)
- The cable lengths between the Tee connectors have to be in minimum 1m (length = C)
- The required transfer rate define the total length of the bus cable (length = E)
- At maximum 31 sensors in the same bus
- The cables must be twisted pair and shielded
- In order to define the resistance level when no transmitter is active, the master must have failsafe bias resistors R_B
- The total length of the bus cable depends on the data transfer rate and can be calculated by following formula: 10^8 [bps*m]/ Data transfer rate [bps]

3.2.1 Point to point structure

---
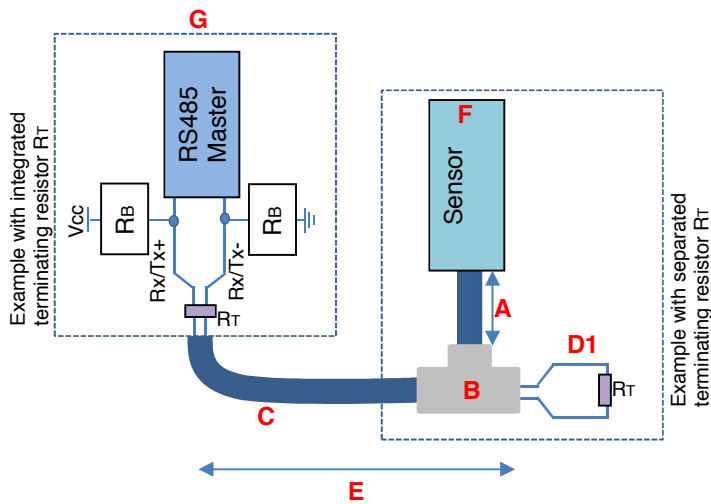
[3] According to the data sheet in the sensors operating manual
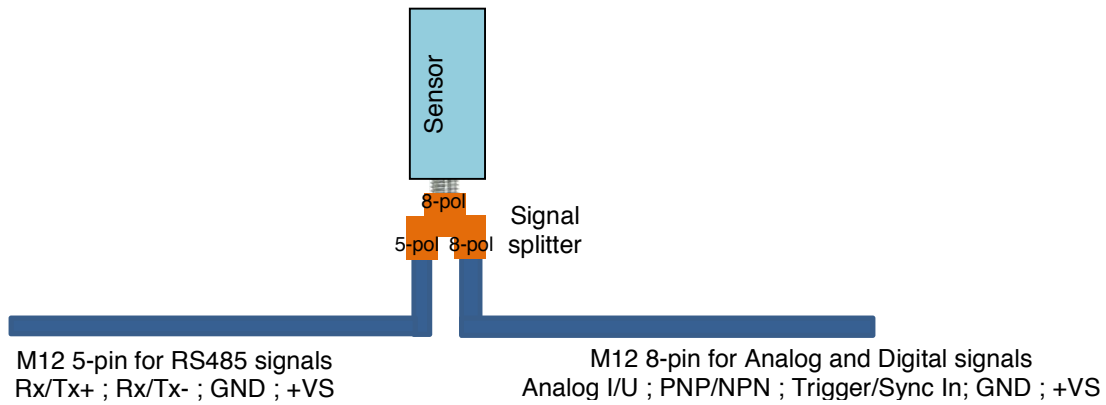
3.2.2 Structure for multiple sensors

A	**Stub cable** Maximum length 0.3m

B	**Tee connector** T-junction M12 CAN 5-pin (1 male/2 female)

C	**Bus cable (between devices)** Minimum 1m and maximum 5m length

D1 D2	**R$_T$ Terminating resistor** M12, 5-pole, 120 Ohm, male or female

E	**Bus cable (maximum length over all)** The total maximum length between the both terminating resistors depends on the data transfer rate: 10^8 [bps*m]/ Data transfer rate [bps] Example: 10^8/3000000=33 --> 33m

F	**Sensor** Sensor with Welotec fast RS485 interface. Data transfer rates up to 3Mbit/s.

G	**RS485 Master with ≤ 3Mbit/s** The PC/PLC (Programmable Logic Controller) controls the devices as master. The master must have two Failsafe bias resistors R$_B$ (600 Ohm - 5 kOhm, Pull-Up for Rx/Tx+ and Pull-Down for Rx/Tx- resistors). Data transfer rate ≤ 3Mbit/s.

3.3   Using other outputs in addition to RS485

For using the other outputs like digital and analog outputs separated, the Signal splitter can be used for.

Sensor

8-pol

5-pol 8-pol   Signal splitter

M12 5-pin for RS485 signals
Rx/Tx+ ; Rx/Tx- ; GND ; +VS

M12 8-pin for Analog and Digital signals
Analog I/U ; PNP/NPN ; Trigger/Sync In; GND ; +VS

Zum Hagenbach 7 • D-48366 Laer • Fon: +49 (0)2554/9130-00 • Fax: +49 (0)2554/9130-10 • info@welotec.com
www.welotec.com
9/25

4   Commands

4.1   Command Structure

:01W 020; 10 ;41BE\r\n

PAYLOAD

START
DEVICE ADDR
TYPE
INDEX
SEPARATOR
PAYLOAD ELEMENT SEPARATOR
CHECKSUM
END

The information to be transmitted is called PAYLOAD and has to be sent in a so-called frame so that the command can be recognized and processed.
This frame always has the same structure and contains a start, a device address, a PAYLOAD, a checksum and an end.

START	DEVICE ADDR	PAYLOAD	CHECKSUM	END
1 char	2 char	n char	4 char	2 char
:	01…31	Commands	****	\r\n

4.1.1   START

:01W020;10;41BE\r\n

The beginning of the command is indicated by a colon : .

ASCII	HEX
:	0x3A

4.1.2   DEVICE ADDR

:01W020;10;41BE\r\n

DEVICE ADDR stands for device address and corresponds to the address which is required to actuate the correct sensor in a network with several sensors. The device address 01 … 31 can be programmed in the sensor. Sensors in the same network must not have the same device address. The standard device address is 01.
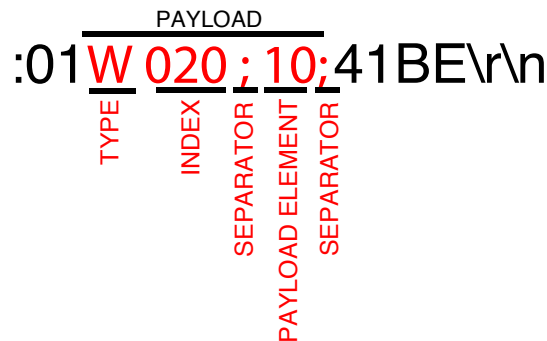
ASCII	HEX		
0…31	0x30 0x30…0x33 0x31		

4.1.3 PAYLOAD

The information to be transmitted is called PAYLOAD.
There are 2 different PAYLOAD types:

- **Legible Coding-** Developed for controlling the sensor with a terminal program
- **Machine Coding-** Developed to ensure efficient and reliable communication between devices

4.1.3.1 PAYLOAD for Legible Coding

Each PAYLOAD begins with the letter for TYPE, followed by the 3-digit index command (during sending). Then any number of payload elements can follow (depending on the index command), each separated by so-called separators.

TYPE
Defines how the sensor has to deal with the command (read or write).

TYPE List for Send Command

ASCII	HEX	MESSAGE	Explanation
R	0x52	READ	Read from the sensor. No PAYLOAD ELEMENT.
W	0x57	WRITE	Write to the sensor. With PAYLOAD ELEMENT.

TYPE List for Sensor Answer

ASCII	HEX	MESSAGE	Explanation
A	0x41	ACK	The command was received and executed successfully.
a	0x61	ACKBUSY	The command was received successfully, but execution requires additional time. Applies to very time-intensive commands and does not mean that the command was not executed.
B	0x42	BUSY	The device is busy and could not receive the command.
E	0x45	ERROR	An error occurred during analysis or execution of the command. For more information, see section on Error Correction.
e	0x65	ERROR LASTCMD	An error occurred during the last command (applies to read/write - delayed commands). The transmitted command was ignored. For more information, see section on Error Correction.

INDEX

Consists of 3 characters and defines the command to be executed (See separate RS485 Index Command List).

ASCII	Hex
000 … 999	0x30 0x30 0x30 … 0x39 0x39 0x39

SEPARATOR

Separator to subdivide commands.

ASCII	HEX	MESSAGE
⬚	0x3B	READ

PAYLOAD ELEMENT

The corresponding value is added depending on the INDEX command and the TYPE.

If TYPE = R (read), there is no need for a PAYLOAD ELEMENT.

If TYPE = W (write), a PAYLOAD ELEMENT is needed for transmitting the value to the sensor.

PAYLOAD for writing to device

	TYPE	INDEX	SEPARATOR	PAYLOAD ELEMENT	SEPARATOR	PAYLOAD ELEMENT	SEPARATOR	…
	1 char	3 char	1 char	n char	1 char	n char	1 char	…
Example	W	020	;	10	;	….	;	…

Example:

:01W020;10; **41BE\r\n** (W=Write; 020= Measurement Type Selection; 10 = AVG Distance)

PAYLOAD for reading from device

	TYPE	INDEX
	1 char	3 char
Example	R	020

Example:

:01R020; **99F5\r\n** (R=Read; 020=Measurement Type Selection)

4.1.3.2 PAYLOAD for Machine Coding

PAYLOAD from master to device (command)

TYPE	INDEX	DATA
uint8	uint8	uint8 [ ]
7-bit ASCII bin*		

PAYLOAD from device to master (answer)

TYPE	PAYLOAD
uint8	uint8 [ ]
7-bit ASCII bin*	

*The PAYLOAD consists of TYPE + INDEX + DATA as a 7-bit ASCII bit stream, see section "7-bit ASCII bin coding"

PAYLOAD example:
12Aq34oit&/&()

TYPE

VALUE	MESSAGE
0	Reserved
1	READ
2	WRITE
3	ACK
4	ACKBUSY
5	BUSY
6	ERROR
7	ERROR LASTCMD

INDEX

VALUE
0…255

DATA

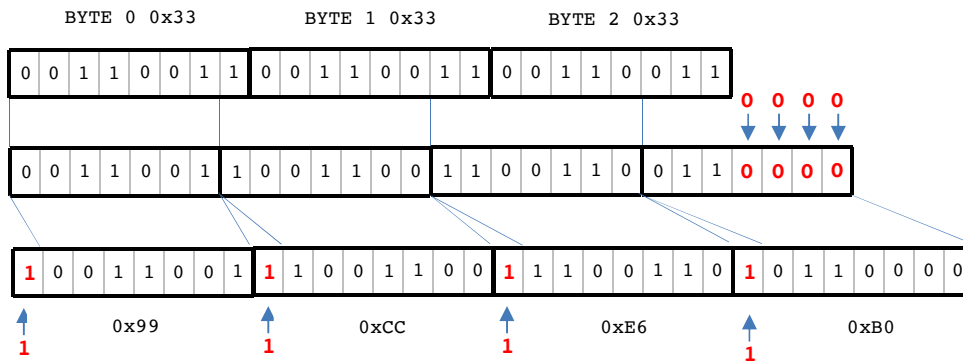| Data type	Coding
uint8, uint16, uint32, int8, int16, int32	Little endian integer (low byte first)
float32	32 bit according to IEEE754
bool	Coded as uint8, whereby: 0: false 1: true
string	Zero terminated string with a defined maximum length. The field is filled with zeros after the zero termination up to the defined maximum length. The specified maximum length includes the zero termination byte.
fixlist[]	All entries are linked.
varlist[]	All entries are linked and pre-pended with uint32 which contains the number of entries in the list.

7-bit ASCII bin coding

Ciphering method
- The binary data is divided into 7-bit groups
- The last 7-bit groups are filled with 0 bits

Extend each group with a most significant bit of 1 to 8 bits

Example:

Raw data: 0x33 0x33 0x33

Transmission data: 0x99 0xCC 0xE6 0xB0

```
      BYTE 0 0x33          BYTE 1 0x33          BYTE 2 0x33

  0 0 1 1 0 0 1 1 | 0 0 1 1 0 0 1 1 | 0 0 1 1 0 0 1 1           0 0 0 0
                                                                ↓ ↓ ↓ ↓

  0 0 1 1 0 0 1 | 1 0 0 1 1 0 0 | 1 1 0 0 1 1 0 | 0 1 1 0 0 0 0


  1 0 0 1 1 0 0 1 | 1 1 0 0 1 1 0 0 | 1 1 1 0 0 1 1 0 | 1 0 1 1 0 0 0 0
  ↑                 ↑                 ↑                 ↑
  1      0x99       1      0xCC       1      0xE6       1      0xB0
```

4.1.4 CHECKSUM

:01W020;10;41BE\r\n

The CHECKSUM is used to check for correct transmission. It consists of the START, DEVICE ADDR and PAYLOAD values and always has 4 digits.

Calculation:
CRC16-ARC / CRC-IBM
Bit-reflected polynom : $x^{16} + x^{15} + x^2 + 1 = 0x8005$, bit-reflected $= 0xA001$.
Init value : $0x0000$, no final XOR value.
See http://reveng.sourceforge.net/crc-catalogue/16.htm for details.

Calculation example:

START	:
DEVICE ADDR	01
PAYLOAD	W020;10;

Accordingly, the checksum consists of: **:01W020;10;** = **41BE**

Wildcard character for checksum:
A wildcard character can also be used as CHECKSUM.

ASCII	HEX
****	0x2A0x2A0x2A0x2A

### 4.1.5 END

:01W020;10;41BE\r\n

The end of the frame or command is marked by the 4-digit combination \r\n.
Important: This command must always be sent as HEX.

ASCII	HEX
\r\n	0x0D 0x0A

Zum Hagenbach 7 • D-48366 Laer • Fon: +49 (0)2554/9130-00 • Fax: +49 (0)2554/9130-10 • info@welotec.com
www.welotec.com
15/25

4.2 Time Flows (Timing)

The time sequences are defined as follows:

| Name	Description	Min	Max
t_answer	Time from reception of the last character of request to sending the first character of the answer.		2.5 ms
t_answer_diagmode	Time from reception of last character of request to sending the first character of the answer if the sensor is in diagnosing mode.		200 ms
t_idle	Time from reception of the last character of answer to sending the first character of the next request.	0.1 ms	
t_break	If a request or an answer is not completed during t_break, the message is rejected. Note: There are exceptions for index Commands with a lot of data.		500 ms
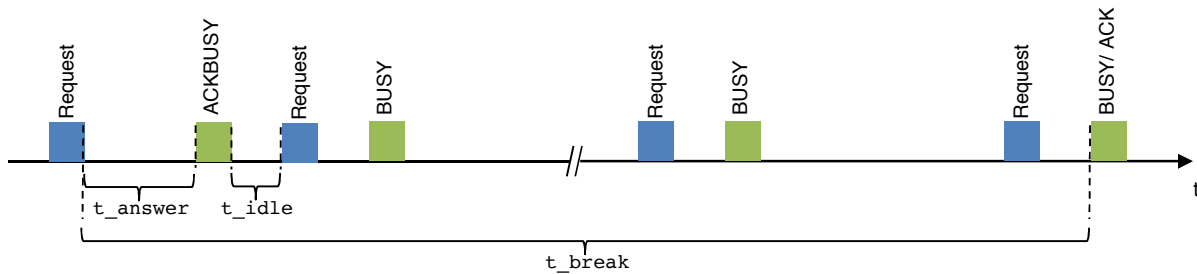
4.2.1 Sequences

Simple Read

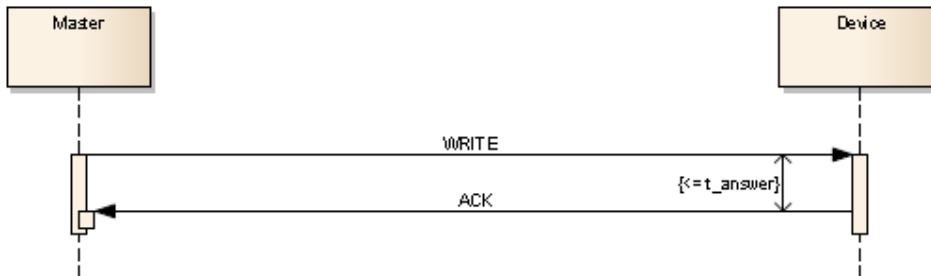Read access to an index, the data is replied immediately.
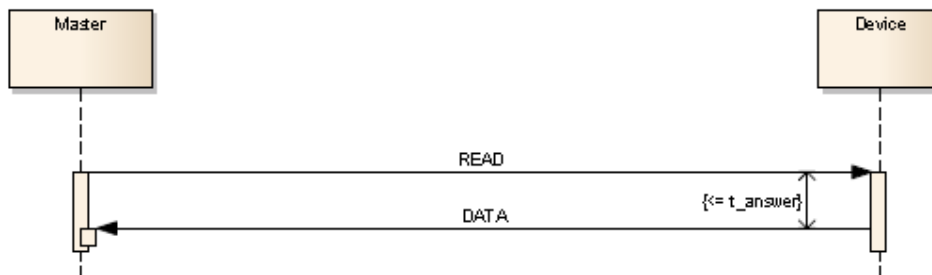
1) READ command
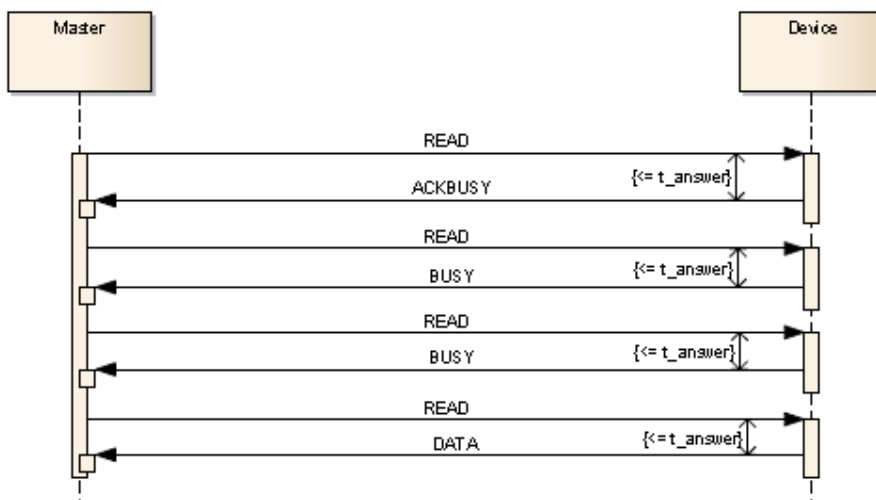2) Answer time <= t_answer
3) Acknowledgment and data ACK,<DATA>

Simple Write

Write access to an index.
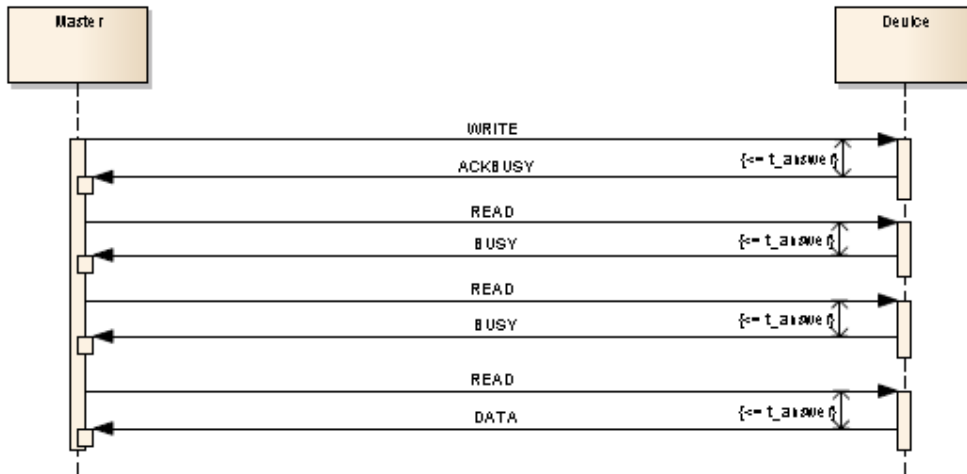

1) WRITE command
2) Answer time <= t_answer
3) Acknowledgment ACK

Read with postponed answer

1) READ command
2) Answer time <= t_answer   1. Sequence
3) ACKBUSY answer

4) READ command
5) Answer time <= t_answer   2. Sequence
6) BUSY answer

7) READ command
8) Answer time <= t_answer   3. Sequence
9) BUSY answer

10) READ command
11) Answer time <= t_answer   4. Sequence
12) Acknowledgment and data ACK,<DATA>

Write with postponed answer

1) WRITE command,<DATA>
2) Answer time <= t_answer   ⎫ 1. Sequence
3) ACKBUSY answer
4) READ command
5) Answer time <= t_answer   ⎫ 2. Sequence
6) BUSY answer
7) READ command
8) Answer time <= t_answer   ⎫ 3. Sequence
9) BUSY answer
10) READ command
11) Answer time <= t_answer   ⎫ 4. Sequence
12) ACK acknowledgment
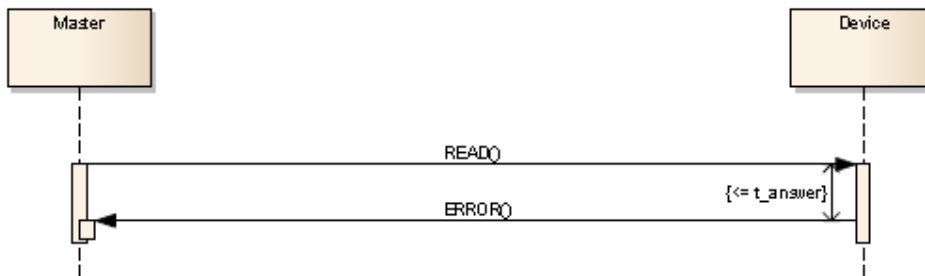
Read with error

1) READ command
2) Answer time <= t_answer
3) ERROR answer

Write Postponed with error
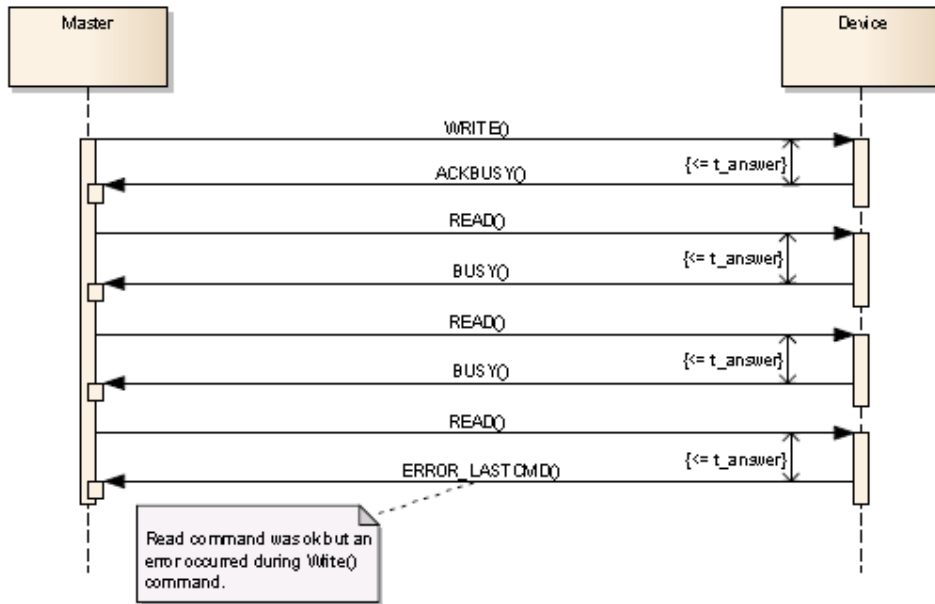
1) WRITE command,<DATA>
2) Answer time <= t_answer
3) ACKBUSY answer

    1. Sequence

4) READ command
5) Answer time <= t_answer
6) BUSY answer

    2. Sequence

7) READ command
8) Answer time <= t_answer
9) BUSY answer

    3. Sequence

10) READ command
11) Answer time <= t_answer
12) ERROR_LASTCMD answer

    4. Sequence

5 Error handling

5.1 General information

<p style="text-align:center;font-size:2em;">:01E;11;2E72\r\n</p>

If an error occurs, the sensor answers with an E or e in the payload. The E is written as an uppor or a lower case letter depending on what the error was.

| E | 0x45 | ERROR | An error occurred during analysis or execution of the command. |
| e | 0x65 | ERROR LASTCMD | An error occurred during the last command (applies to read/write - delayed commands). The transmitted command was ignored. |

The error type is also sent with a number in the payload.

1	Wrong message type	Only Read or Write are valid for requests from master to device.
2	Wrong payload format	For example a missing separator
3	Wrong argument	The given argument type doesn't match the expected type (ex. Float value if an integer value is expected).
4	Wrong argument count	Number of arguments mismatch.
5	Not enough data	The length of the payload is below its minimum (type and index).
6	Index do not exist	
7	Index locked	
8	Access not allowed	Ex. Write access to read only index.
9	Not enough memory for encoding	Internal error.
10	Not possible to encode argument	Internal error.
11	Application specific error	Not RS485 stack related error, read out chapter "Application specific error 11" and the sensor-belonging Index Command 000 "Application error" for detailed information.
12	Wrong state	Internal error.

Sample answer of a sensor when an error occurred: :01**e**;**11**;2E72\r\n
Error during the last command: Application specific error

WELOTEC®

Zum Hagenbach 7 • D-48366 Laer • Fon: +49 (0)2554/9130-00 • Fax: +49 (0)2554/9130-10 • info@welotec.com   20/25
www.welotec.com

5.2   Application specific error 11

If the sensor outputs an error with the error number 11 - Application specific error -, the exact error type can be identified via the Index Command 000 (for instructions see Index Command List).
For information to the exact application error number please refer to the sensor specific RS485 manual.

Example:

Step	Command to sensor	Answer from the sensor	Description
1		:01E;11;2E72 \r\n	The sensor sends back an error 11 (application specific error)
2	:01R000;5954 \r\n		The exact error code is requested with the index command 000 (read out actual pending error)
3		:01A; 99;EC05 \r\n	After which the sensor sends back the exact application error: "Application error: 99: argument out of range"

Zum Hagenbach 7 • D-48366 Laer • Fon: +49 (0)2554/9130-00 • Fax: +49 (0)2554/9130-10 • info@welotec.com
www.welotec.com

21/25

6   Appendix

6.1   Data types

Data type	Coding
`uint8`	1-3 characters Max: `255` Min: `0`

`int8`	1 sign (optional) and 1-3 characters
	
	Max: `127` Min: `-128`
	
	Examples:
	`-120`
	`+33`
	`78`

`uint16`	1-5 characters Max: `65535` Min: `0`

`int16`	1 sign (optional) and 1- 5 characters
	
	Max: `32767` Min: `-32768`
	
	Examples
	`-32768`
	`+378`
	`258`

`uint32`	1-10 characters Max: `4'294'967'295` Min: `0`

`int32`	1 sign (optional) and 1- 10 characters
	
	Max: `2'147'483'647`
	Min: `-2'147'483'648`
	
	Examples
	`41647869`
	`-52148877`
	`+54547657`

`float32`	1 sign (optional) and 1-12 characters
	
	Decimal separator '`.`'(`0x2E`)
	
	Note: Since the device uses floating point numbers according to IEEE754, rounding errors may appear. This rounding error can lead to a small difference between the written and the read value.
	
	Examples
	`123.23487824`
	`-123.23487824`
	`124578`
	`+91.27`

`bool`	Coded as `uint8` whereby: `1 (0x31)`: false `0 (0x30)`: true

string	Variable number of characters, without zero termination.
	The string must not contain ';' (`0x3B`).
	Only readable ascii values are allowed (`0x20` … `0x7E`).
	The string may be shorter than the specified length.
	Note: Due to compatibility with the machine coding, the actual length is 1 character less than the specified maximum length in the index table.
`Fixlist[type]`	`<SPACE><Entry 1><SPACE><Entry2><SPACE>`
	Example: Fixed list containing the three elements 77, 22, 33:
	`77 22 33`
`varlist[type]`	`<Nr of entries><SPACE><Entry1><SPACE><Entry2><SPACE>…`
	Example: List containing the three elements 7, 22, 333:
	`3 7 222 33`

7   History of changes

Date	Version	Description
12.12.2014	1.0	Document created
12.01.2015	1.1	Examples implemented, smaller changings e.g. Payload Element
10.02.2015	1.2	Updated with information from R&D
31.07.2015	1.2	Reference text concerning Application specific Error 11
05.01.2016	1.3	3 Mbit structure changes (Terminator male and female)
07.06.2017	1.4	Errors on S.10 corrected