Modbus adapter for testo 350

Instruction manual

Contents

1 About this document

- The instruction manual is an integral part of the instrument.
- Pay particular attention to the safety instructions and warning advice in order to prevent injury and damage to the product.
- Please read this instruction manual through carefully and familiarize yourself with the product before putting it to use.

Symbols and writing standards

Display	Explanation
	Note: basic or further information
	Warning advice, risk level according to the signal word: **Warning!** Serious physical injury may occur. **Caution!** Minor physical injury or damage to the equipment may occur. > Take the specified precautionary measures.
1 2 …	Action: several steps, the sequence must be followed
-	Result of an action
✓	Requirement
>	Action

Warnings

Always pay attention to any information marked with the following warning notices along with warning pictograms. Implement the specified precautionary measures!

⚠ DANGER

Risk of death!

⚠ WARNING

Indicates possible serious injury.

⚠ CAUTION

Indicates possible minor injury.

CAUTION

Indicates possible damage to equipment.

2 Safety and disposal

General safety instructions

- Always operate the product properly, for its intended purpose and within the parameters specified in the technical data. Do not use any force.
- Do not commission the instrument if there are signs of damage on the housing.
- Dangers may also arise from the systems being measured or the measuring environment: Make sure you comply with the locally valid safety regulations when carrying out measurements.
- Do not expose the product to temperatures above 50 °C (122 °F).
- Do not store the product together with solvents. Do not use any desiccants.
- Only maintenance and repair work that is described in the documentation may be carried out on this instrument. Follow the prescribed steps exactly when doing the work. Only use original spare parts from Testo.

2.1 Product-specific information

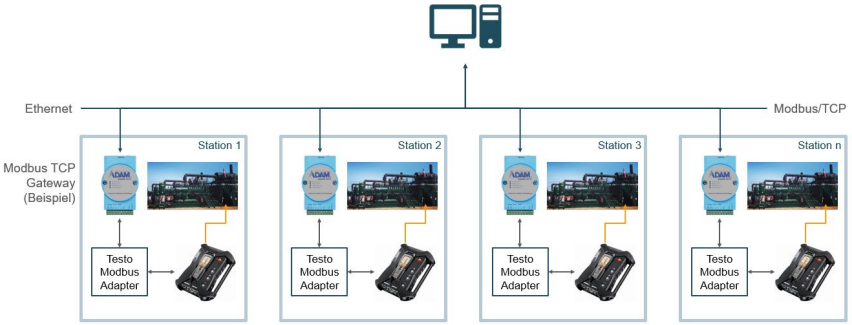
- Only have damaged adapters replaced by authorized specialist personnel.
- Have the adapter wired and connected by authorized specialists only when it is disconnected from the power supply.
- Always observe the regulations in force in your country for opening and repairing electrical equipment.
- Do not connect cables firmly to low-voltage parts.
- Before installing the components, check the top-hat rail for correct earthing.
- Have installation work carried out only by qualified and authorized personnel!
- Only open the unit if this is expressly described in the operating instructions for installation, maintenance or repair work.

2.2 Disposal

- At the end of its useful life, deliver the product to the separate collection point for electric and electronic devices (observe local regulations) or return the product to Testo for disposal.
-  WEEE Reg. No. DE 75334352

3 Intended use

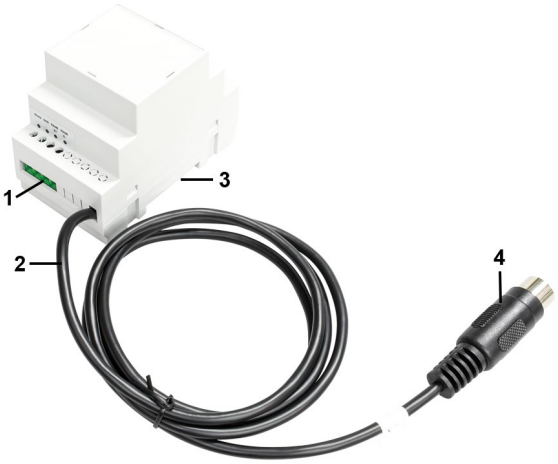
The testo modbus adapter is intended to connect a testo 350 device with the customer's modbus system.



- The power supply of the adapter comes from the socket of the testo 350.
- The RS 485 line (e.g. to a Gateway) can be connected via screw terminals.
- The input signal and power comes through a fixed cable with Hirschmann (DIN 8-pin) connector.
- Input and output signals are galvanic isolated.
- When the adapter is connected to the device, no additional probes or industrial probe can be connected.
- The modbus adapter always has to be the last device on a modbus interface.

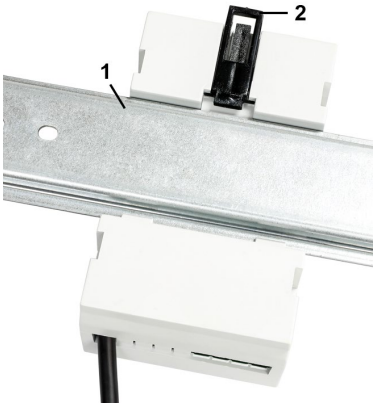
4 Product description

4.1 Overview of the modbus adapter



1	Cut-out for connection cable to customer system	**2**	Connection cable to testo 350
3	Recess for mounting rail installation (NS35 according to DIN 60715)	**4**	8 pin connector to testo 350

4.2 Rear side of the modbus adapter



1	mounting rail	**2**	Latch to open and close

4.3 Connector on testo 350 side

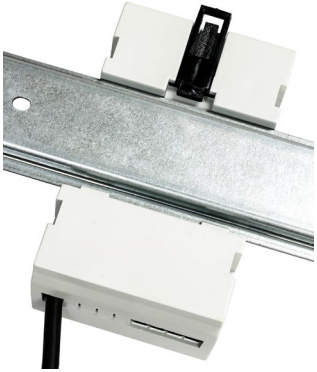
1	Connector to modbus adapter

5 Using the product

5.1 Mounting the device on a mounting rail

1. Push the latch on the back of the Modbus adapter upwards.

2. Hook the Modbus adapter into the mounting rail (NS35 according to DIN 60715) at the bottom and then push the top side backwards.



3. Push the latch on the back side downwards to fix the Modbus adapter to the mounting rail.

4. To secure the Modbus adapter against lateral slipping, attach an end clamp on both sides:
   - Hook the end clamp with the closed side facing outwards directly next to the Modbus adapter into the mounting rail.
   - Press the other side of the end clamp against the mounting rail until it audibly engages.

5.2 Connecting the device to testo 350

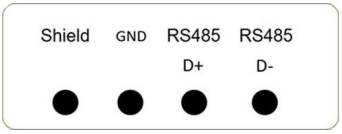
⚠ DANGER

Dangerous voltage, danger of electric shock!

- **Only have damaged adapters replaced by authorized specialist personnel.**
- **- Have the adapter wired and connected by authorized specialists only when it is de-energized.**
- **- Always observe the regulations applicable in your country for opening and repairing electrical equipment.**
- **- Do not connect cables firmly to low-voltage parts.**
- **- Before installing the components, check DIN rail for correct grounding.**

1 Connect the connection cable of the Modbus adapter to connection 1 on the testo 350.

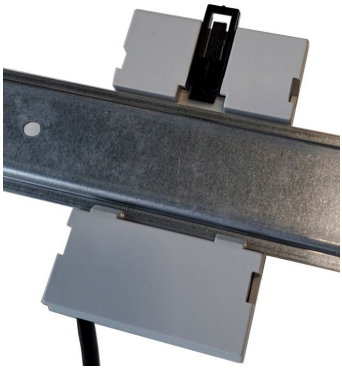
2 Connect customer-side connections to the bottom right of the Modbus adapter.



5.3 Unmounting the device from a mounting rail

1. To remove the side end clamps, use a screwdriver to loosen the lock of the end clamps.

2. Push the latch on the back of the Modbus adapter upwards.



3. Remove the Modbus adapter from the mounting rail.

6 Technical data

6.1 Technical data modbus adapter

Feature	Value
Storage and transport temperature	-20 to +50 °C / -4.0° to 122.0 °F
Operating temperature	-5 to +45 °C / 23.0° to 113.0 °F
Protection class	IP20
Fire protection class	V-2 UL94
Power supply	Via the socket of testo 350
Weight	140g / 0,31 lbs
Dimensions	53 x 90 x 65 mm / 2.1 x 3.5 x 2.6 In
Housing	ABS V0

6.2 Modbus implementation

The modbus implementation for testo 350 provides a subset of the interactions described in the Modbus Application Protocol Specification V1.1b.

6.2.1 Physical Link Layer

The modbus interface of the testo 350 is realized with a UART, using separated lines for receive and transmit.

- Logic high voltage is 3.3 V, logic low voltage is 0 V.
- Data rate is set to 9600 Baud (not changeable)
- Frame settings are 1 start bit, 8 data bit, 1 even parity bit, 1 stop bit.

The testo 350 needs up to 400 ms between reception of a request and sending the answer. Timeout setting of the master should be greater or equal 400 ms.

6.2.2 Implemented Function Code Subset

The following simplifications were applied:

- Coils and discrete inputs are not used.
  Function code 0x01, 0x02, 0x05, 0x0F are not supported.
- Holding registers are not used.
  Function code 0x03, 0x06, 0x16, 0x17 are not supported.
- All exchanged information is packed in input registers.
  Function codes 0x04, 0x10 are supported.
- Serial line diagnostic commands are not supported.
  Function codes 0x07, 0x08, 0x0B, 0x0C, 0x11 are not supported.
- File Records, FIFOs and Encapsulated Interface Transport are not used.
  Function codes 0x14, 0x15, 0x18, 0x43 are not supported.

6.2.3 Data Representation

Datatypes provides by the testo 350 are

- BYTE (8-bit integer)
  A byte can be interpreted signed or unsigned, see detailed register description.
- WORD (16-bit integer)
  A word can be interpreted signed or unsigned, see detailed register description.
- DWORD (32-bit integer)
  A word can be interpreted signed or unsigned, see detailed register description.
- FLOAT (32-bit single precision floating point number)

Transmission follows “big endian” scheme, the most significant byte (MSB) is sent first. Modbus registers are always 16 bits wide. Therefor

- 32-bit data types are transmitted in 2 consecutive registers
- 8-bit data types are encapsulated in a 16-bit register, the MSB (transmitted first) is set to 0

Register addresses are 16 bits, allowed values are in 0 … 65535 (0x000 … 0xFFFF). MSB is transmitted first.

6.2.4 Error Handling

General error handling follows Modbus Application Protocol Specification V1.1b.

Contrary to the rules in the specification chapter 7 errors caused by erroneous content of register write commands are answered with error code 4.

Rationale:

- error code 4 has a meaning very close to “erroneous command”
- error code 4 is known by all modbus systems, it’s usage will not provoke additional problems

6.2.5 Device ID

Modbus is a bus-oriented protocol for one client and multiple servers. Therefor each client needs an ID. The testo bus address (see testo 350 instruction manual chapter 5.2.4.2) will be used for this purpose.

If you plan to have multiple testo 350 connected to one modbus master, make sure that each testo 350 analyzer box and each testo 350 control unit has a unique bus address.

Modbus specification uses address 0 for broadcast messages. Therefor address 0 must not be used as modbus slave address.

If no bus address has been specified, address 3 is selected by default. After changing the device address the analyzer box must be restarted.

6.2.6 Multimaster Mode

If a Control Unit is connected to a modbus controlled analyzer box, this box can be commanded by two “masters” simultaneously. It is obvious, that conflicting commands are possible in this mode.

ATTENTION

Caution is strongly recommended when the control unit and modbus are used simultaneously.

6.2.7 Watchdog

To avoid uncontrolled operation, a watchdog is active when no control unit is attached to the device. This watchdog will be fed by any modbus command. When no command is received for more than 60 sec. the device will

- switch off when accumulator powered
- go to standby (=battery charging) mode when AC powered 21

We strongly recommend to disconnect the control unit during modbus operation when it is not accessible (e.g., stored in a closed cabinet).

6.3 Function Codes

6.3.1 0x04: Read registers

This function code is used to read from 1 to 125 contiguous registers in a remote device. The Request PDU specifies the starting register address and the number of registers.

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

Request

Function Code	1 Byte	**0x04**
Start Address	2 Byte	0x0000 … 0xFFFF
Quantity of registers	2 Byte	0x0001 … 0x007D

Response

Function Code	1 Byte	**0x04**
Byte Count	1 Byte	2 * N
Input registers	N*2 Bytes	Data

N: Quantity of registers

Error

Error Code	1 Byte	**0x04** (= Function Code OR 0x80)
Exception code	1 Byte	0x01…0x04, see MODBUS Application Protocol Specification V1.1b

Example

Request 2 registers, starting at 0x1000. First one responds a word (0x1234), second one responds a byte (0x56).

Request		Response	
Field Name	Data	Field Name	Data
Function Code	0x04	Function Code	0x04
Start address (MSB)	0x10	Byte Count	0x04
Start address (LSB)	0x00	Reg. 0x1000 MSB	0x12
Quantity (MSB)	0x00	Reg. 0x1000 LSB	0x34
Quantity (LSB)	0x02	Reg. 0x1001 MSB	0x00
		Reg. 0x1001 LSB	0x56

6.3.2 0x10: Write registers

This function code is used to write a block of contiguous registers (1 to 123 registers) in a remote device.

The requested written values are specified in the request data field. Data is packed as two bytes per register.

The normal response returns the function code, starting address, and quantity of registers written.

Request

Function Code	1 Byte	**0x10**
Start Address	2 Byte	0x0000 … 0xFFFF
Quantity of registers	2 Byte	0x0001 … 0x007D
Byte Count	1 Byte	N*2
Registers values	N*2 Bytes	data
Function Code	1 Byte	0x10

N: Quantity of registers

Response

Function Code	1 Byte	**0x10**
Start address	2 Byte	0x0000 … 0xFFFF
Quantity of registers	2 Byte	0x0000 … 0x007B

Error

Error Code	1 Byte	**0x90** (= Function Code OR 0x80)
Exception code	1 Byte	0x01…0x04, see MODBUS Application Protocol Specification V1.1b

Example

Write 2 registers, starting at 0x1000. First one receives a word (0x1234), second one receives a byte (0x56).

Request		Response	
Field Name	Data	Field Name	Data
Function Code	0x10	Function Code	0x10
Start address (MSB)	0x10	Byte Count	0x04
Start address (LSB)	0x00	Start address (MSB)	0x10
Quantity (MSB)	0x00	Start address (LSB)	0x00
Quantity (LSB)	0x02	Quantity (MSB)	0x00
Byte Count	0x04	Quantity (LSB)	0x02
Reg. 0x1000 MSB	0x12		
Reg. 0x1000 LSB	0x34		
Reg. 0x1001 MSB	0x00		
Reg. 0x1001 LSB	0x56		

6.4 Registers

6.4.1 Device Identification

6.4.1.1 0x1000: Device Type

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read Only

This register is used to identify the device type. For testo 350 it will always contain 0x15E (= 350). Write attempt will result in error code 4 and leave no changes in the device.

Device ID	
0x1000, MSB	0x01
0x1000, LSB	0x5E

6.4.1.2 0x1001, 0x1002: Device Serial number

Data type in testo 350: DWORD

Data type on modbus: Two WORDs

Access: Read Only

This register is used to identify the device type. Write attempt will result in error code 4 and leave no changes in the device.

Device Serial number	
0x1001, MSB	SN Byte 3 (MSB)
0x1001, LSB	SN Byte 2
0x1002, MSB	SN Byte 1
0x1002, LSB	SN Byte 0 (LSB)

6.4.1.3 0x1003: Firmware Revision

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read Only

This register is used to read the firmware revision of the testo 350 measurement device. Write attempt will result in error code 4 and leave no changes in the device.

Firmware revision	
0x1003, MSB	Major revision code
0x1003, LSB	Minor revision code

6.4.2 Measurement System States

6.4.2.1 0x2000: Measurement Application

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read / Write

This register is used to get or set the selection measurement application.

Measurement application	
0x2000, MSB	0
0x2000, LSB	Application code

Application codes are defined as follows:

Code	Application
0x00	Application Burner
0x01	Application turbine
0x02	Motor with lambda > 1
0x03	Motor with lambda <= 1
0x04	User defined application
others	Return Error code

6.4.2.2 0x2001: Measurement Type

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read / Write

This register is used to get or set the selection measurement type.

In testo 350 we must distinguish between measurements and measurement programs:

- A single measurement (also called spot measurement) is a free running evaluation of physical values. The results are displayed continuously, but no result will be stored at the end of the measurement.
- A measurement program is a fix sequence of one or more measurements with defined duration, sometimes manual assistance (like “open air valve”). Results are stored at the end of the measurement programs (in many cases averaged values).

The current modbus implementation gives no access to measurement programs. All measurement modes describes below are of type “single measurement”.

The following measurement modes are used in testo 350:

Code	Mnemonic	Description
0x01	MEAS_FLUEGAS	Flue gas measurement
0x02	MEAS_DELTAP	Differential pressure measurement
0x03	MEAS_DRAUGHT	Draught measurement
0x04	MEAS_FLUEGAS_CAT	Flue gas measurement before/behind catalyzer
0x05	MEAS_FLUEGAS_MS	Flue gas measurement m/s
0x06	MEAS_FLUEGAS_DP	Flue gas measurement delta P
	others	In case of write: invalid, will result in error code 4

6.4.2.3 0x2002: Measurement State

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read / Write

When read, this register reports the state of the current measurement:

Code	Mnemonic	Description
0x00	E_MEAS_VIEW_STATE_STARTUP	Measurement not initialized
0x01	E_MEAS_VIEW_STATE_IDLE	Measurement idle
0x02	E_MEAS_VIEW_STATE_RUNNING	Measurement in progress
0x03	E_MEAS_VIEW_STATE_ZERO	Measurement zeroing
0x04	E_MEAS_VIEW_STATE_RINSE	Measurement rinsing
0x05	E_MEAS_VIEW_STATE_WANTS_START	Measurement zeroing with automatic start afterwards
0x06	E_MEAS_VIEW_STATE_DEADTIME	Dead time (*)
0x07	E_MEAS_VIEW_STATE_STABILIZATION	Stabilization time running (*)
0x08	E_MEAS_VIEW_STATE_WAIT_ZERO	Measurement program waits for zeroing start confirmed by user
0x09	E_MEAS_VIEW_STATE_WAIT_RAMP	Measurement program waits for ramp up start confirmed by user
0x0A	E_MEAS_VIEW_STATE_WAIT_TEST	Measurement program waits for test start confirmed by user
0x0B	E_MEAS_VIEW_STATE_WAIT_RINSE	Measurement program waits for rinsing start confirmed by user
0x0C	E_MEAS_VIEW_STATE_WAIT_EXIT	Measurement program waits for exit confirmed by user

Measurement programs cannot be initiated through the modbus interface.

When written, this record commands a new state of the current measurement.

Code	Mnemonic	Description
0x00	MEAS_START	Start selected measurement
0x01	MEAS_STOP	Stop measurement and, in case of a measurement program, save results

Code	Mnemonic	Description
0x02	MEAS_CANCEL	Stop measurement and, in case of a measurement program, discard results
0x03	MEAS_ZERO	Start conditional zeroing (zeroing will be skipped if already done)
0x04	MEAS_RINSE	Start normal rinsing (rinsing stops automatically when gas is clean)
0x05	MEAS_UNUSED	Do not use
0x06	MEAS_ZERO_FORCE	Start unconditional zeroing (zeroing is executed in any case)
0x07	MEAS_RINSE_FORCE	Start endless rinsing (rinsing must be stopped by command)
0x08	MEAS_UNUSED	Do not use
0x09	MEAS_ZERO_FORCE_2	Start unconditional pressure sensor zeroing during an active measurement

Measurement programs cannot be initiated through the modbus interface.

6.4.2.4 0x2003: Fuel ID

Data type in testo 350: WORD

Data type on modbus: One WORD

Access: Read / Write

This register is used to get or set the fuel used for calculations:

Code (decimal)	Fuel name (for US devices only)
9000	Natural Gas
9001	Propan
9002	Butane
9003	Fueloil #2
9004	Fueloil #5
9005	Fueloil #6
9006	Kerosine
9007	Diesel
9008	Benzin
9009	Anthrazit Coal
9010	Bitumen Coal
9011	Destillate #1
9012	Wood 10%M.
9013	Wood 20%M.

Code (decimal)	Fuel name (for US devices only)
9014	Wood 30%M.
9015	Wood 40%M.
9016	Bark 15%M.
9017	Bark 30%M.
9018	Bark 45%M.
9019	Bark 60%M.
9020	Bagasse
9021	Heavy Oil
9022	Light Oil
9023	Gasoleo
9024	LPG
9025	Natur.gas Mexico
	Mexico
9026	Bioheat 5
9027	Bioheat 12
9028	Bioheat 20
9029	Bioheat 15
9030	Gas Natural (Mex.)
9031	Fueloil #4
9032	Bioheat 10
99	Test gas
65	User defined fuel #1
66	User defined fuel #2
67 … 89	User defined fuel #3 … #24

There are more fuel definitions in other country versions. These can be delivered on request.

It is recommended to execute a “Get Fuel ID” after any “Set Fuel ID” to get a confirmation.

6.4.2.5 0x2004: Fresh Air Valve

Data type in testo 350: Byte

Data type on modbus: One WORD

Access: Read / Write

This register is used to get or set the state of the fresh air valve:
Possible answers to read access:

Code	Mnemonic	Description
0x00	SWITCH_TO_AIR	Apply fresh air to measurement path

Code	Mnemonic	Description
0x01	SWITCH_TO_GAS	Apply gas probe to measurement path
0x02	SWITCH_TO_NOT_ALLOWED	Switch of fresh air valve not allowed

Possible arguments to write access:

Code	Mnemonic	Description
0x00	SWITCH_TO_AIR	Apply fresh air to measurement path
0x01	SWITCH_TO_GAS	Apply gas probe to measurement path

Normally the valve is controlled by the testo 350 automatically according to the selected measurement modes. For standard applications it is not necessary to change the valves state manually.

Before issuing a write command a previous read of the valve's state is recommended. If reported state is SWITCH_TO_NOT_ALLOWED do not change the valve state. Changing the valve's state will influence measurement results.

6.4.2.6 0x2006: Standby

Data type in testo 350: Byte

Data type on modbus: BYTE

Access: Read / Write

This register is used to get or set the device on and off:
Possible arguments to write access, possible answers to read access:

Code	Mnemonic	Description
0x00	OPERATE	Write: Set device in operating state (needs AC power). Read: Device is in operating state
0x01	STANDBY	Write: Set device in Standby (=accu charging) mode when AC powered Switch device off when accumulator powered. Read: Device is in standby state

Several units (e.g., the gas cooler) are still working when measurement system is idle. To shut down the system as much as possible, this command can be used.

Writing a 1 into this register will set the device in Standby mode.

- When AC powered, the control units display the state will be indicated as "charging accumulator". Watchdog feeding is no more required in this state.
- When powered from accumulator, the device will switch off completely. When switched off, no more modbus communication is possible.

Writing a 0 into this register will set the device in operating state.

ATTENTION

Return to operating state is not possible when the system is powered from the battery.

ATTENTION

Writing a 0 into this register will set the device in operating state. In this case the device needs a minimum delay of 5 seconds for initialization. During this time no Modbus commands are possible.

6.4.2.7 0x2008: Set Dilution

Data type in testo 350: Byte

Data type on modbus: BYTE

Access: Write

Basic requirement: The option "Fresh air valve" and "Measurement range extension (for individual slot)" are available in the testo 350.

Available control values:

		Device options		Comment
Control Value	Dilution Factor	Fresh air valve	Fresh air valve & Dilution module	
0x01	1	☑	☑	no Dilution / regular measurement
0x02	2	- n.a. -	☑	
0x05	5	- n.a. -	☑	
0x0A	10	- n.a. -	☑	
0x14	20	- n.a. -	☑	
0x28	40	- n.a. -	☑	
0x40	5, 1	- n.a. -	☑	Automatic switching
0x80	5	☑	☑	Dilution all gas sensors

6.4.3 Measurement values

Measurement results in testo 350 (named ViewValues) are organized in a list using the same sequence as is visible on the display during the measurement. The values visible on the display are dependent to the selected measurement and to user settings.

Beside the numerical value each measurement value is accompanied by some attributes:

- An identifier for the measured value
- If applicable, the current dilution factor for this gas
- The unit used to express the result

- A code for the recommended display resolution 23

In the testo 350 the ViewValues are stored in records (aka structures), on the modbus these records are splitted in several arrays. For each record component (e.g. measurement value, measurement ID) exists a group of registers. In such a group the components of all available ViewValues are listed in sequential registers. All these register lines will use the same sequence of channels. So the ID of the first ViewValue is at the lowest register of the ID row, the measurement value of this ViewValue is at the lowest register of the value row and so on.

It is recommended to gather the attributes of all channels once before each measurement and read the values only during the measurement.

6.4.3.1 0x3000: NumberOfViewValues

Data type in testo 350: WORD

Data type on modbus: One WORD

Access: Read only

This register is used to get the number of displayed (= available) measurement values. For any subsequent request to registers inside the 0x3000 range the channel index shall be limited to this result minus 1.

Up to 25 ViewValues can be accessed by modbus.

6.4.3.2 0x3100 … 0x3131: Measurement Idents

Data type in testo 350: 25 * DWORD

Data type on modbus: 50 * WORD

Access: Read only

These registers are used to gather the measurement ID for each displayed ViewValue. The measurement ID shows the physical measurand attached to all entries with the same index.

If an unused channel is requested, 0xFFFFFFFF is replied to indicate an invalid channel.

Register mapping:

Register	Content
0x3100, MSB	ViewValue[0] Ident, Byte 3
0x3100, LSB	ViewValue[0] Ident, Byte 2
0x3101, MSB	ViewValue[0] Ident, Byte 1
0x3101, LSB	ViewValue[0] Ident, Byte 0
0x3102, MSB	ViewValue[1] Ident, Byte 3
0x3102, LSB	ViewValue[1] Ident, Byte 2
0x3103, MSB	ViewValue[1] Ident, Byte 1
0x3103, LSB	ViewValue[1] Ident, Byte 0
…	
0x3130, MSB	ViewValue[24] Ident, Byte 3

Register	Content
0x3130, LSB	ViewValue[24] Ident, Byte 2
0x3131, MSB	ViewValue[24] Ident, Byte 1
0x3131, LSB	ViewValue[25] Ident, Byte 0

The IDs listed in these registers are coded as follows:

Code	Mnemonic	Description
0x00000101	AT	flue gas temperature
0x00000102	VT	ambient temperature
0x00000103	GT	analyzer temperature
0x0000010b	TEMP_AMB	ambient temperature
0x00000301	DRAUGHT	draught value
0x00000302	PDIFF	differential pressure
0x00000303	PABS	absolute pressure
0x00000304	FINEDRAUGHT	fine draught value
0x0000030a	EXT_DRAUGHT	ext. draught value
0x0000030b	EXT_DELTAP	ext. pressure value
0x00000124	AT_MEAN	flue gas temperature average
0x00000125	VT_MEAN	ambient temperature average
0x0000091b	O2_MEAN	O2 average value
0x00000901	O2	O2 value
0x00000902	CO	CO value
0x00000903	CO_AMB	CO ambient
0x00000904	CO_UNDIL	CO undiluted
0x00000905	H2	H2 value
0x00000906	NO	NO value
0x00000907	NO2	NO2 value
0x00000908	SO2	SO2 value
0x00000909	CO2	CO2 value
0x0000090a	CxHy	CxHy value
0x0000090b	H2S	H2S value
0x00021282	LAMBDA	LAMBDA value
0x00021281	EXA	excess air
0x00020915	Nox	NOx value
0x00021a06	NO_RED	NO value O2 reduced
0x00021a02	CO_RED	CO value O2 reduced
0x00021a07	NO2_RED	NO2 value O2 reduced
0x00021a08	SO2_RED	SO2 value O2 reduced
0x00021a15	NOx_RED	NOx value O2 reduced
0x00000501	PUMP_FLOW	pump flow value
0x00000601	AKKU_VOLTAGE	battery voltage

Code	Mnemonic	Description
0x00000911	CO2_MEAS	CO2 value measured
0x00020a02	MFLOW_CO	mass flow CO
0x00020a15	MFLOW_NOX	mass flow NOx
0x00020a07	MFLOW_NO2	mass flow NO2
0x00020a08	MFLOW_SO2	mass flow SO2
0x00020a0b	MFLOW_H2S	mass flow H2S
0x00020a11	MFLOW_CO2IR	mass flow CO2 measured
0x0000090d	CO2_MAX	fuel CO2 max value
0x0000090c	O2_REF	fuel O2 ref value

There are other metrics, the corresponding definitions can be provided if required.

6.4.3.3 0x3200 … 0x3231: Measurement Values

Data type in testo 350: 25 * single float (32 Bit)

Data type on modbus: 50 * WORD

Access: Read only

These registers are used to get the measurement result for each displayed ViewValue. Measurement results are represented in 32-bit float value (single precision float values). These float values are split into two 16-bit integer MODBUS registers.

If an unused channel is requested, 0xFFFFFFFF (=Not A Number) is replied to indicate an invalid channel.

Register mapping:

Register	Content
0x3200, MSB	ViewValue[0] Value, Byte 3
0x3200, LSB	ViewValue[0] Value, Byte 2
0x3201, MSB	ViewValue[0] Value, Byte 1
0x3201, LSB	ViewValue[0] Value, Byte 0
0x3202, MSB	ViewValue[1] Value, Byte 3
0x3202, LSB	ViewValue[1] Value, Byte 2
0x3203, MSB	ViewValue[1] Value, Byte 1
0x3203, LSB	ViewValue[1] Value, Byte 0
…	
0x3230, MSB	ViewValue[24] Value, Byte 3
0x3230, LSB	ViewValue[24] Value, Byte 2
0x3231, MSB	ViewValue[24] Value, Byte 1
0x3231, LSB	ViewValue[25] Value, Byte 0

To use these values, the two 16-bit values have to be combined into one 32-bit value interpreted as single float.

If no numerical value can be displayed (e.g., due to overload conditions), a special code is transferred which can't be converted into a valid float number.

Code	Mnemonic	Description
0x00000081	FE_OVERRANGE	Overload, violation of upper measurement range limit
0x00000082	FE_UNDERRANGE	Overload, violation of lower measurement range limit
0x00000083	FE_OUTRANGE	Overload, violation of measurement range limit
0x00000084	FE_DEFECT	Invalid due to a defect
0x00000085	FE_EMPTY	Value not (yet) available (e.g., measurement not started)
0x00000086	FE_WAKEUP	Sensor in wakeup condition
0xFFFFFFFF	FE_NAN	General "not a number" in case of other errors

6.4.3.4 0x3400 … 0x3418: Measurement Units

Data type in testo 350: 25 * WORD

Data type on modbus: 25 * WORD

Access: Read only

These registers are used to gather the measurement unit for each displayed ViewValue. If an unused channel is requested, 0xFFFF is replied to indicate an invalid channel.

Register mapping:

Register	Content
0x3400	ViewValue[0] Unit
0x3401	ViewValue[1] Unit
…	
0x3418	ViewValue[24] Unit

The units listed in these registers are coded as follows:

Code	Mnemonic	Description
0x01	GRAD_C	°C
0x02	GRAD_F	°F
0x03	REL_FEUCHTE	%rF
0x04	REL_HUMIDITY	%
0x05	VELOCITY	m/s
0x16	LAMDDA	Lambda
0x17	DRUCK_MBAR	mBar
0x18	DRUCK_HPA	hPa

Code	Mnemonic	Description
0x19	PSI	psi
0x2C	PPM_CO2	ppm CO2
0x4D	VOLSTROMJ	m³/h
0x52	DRUCK_MM_H2O	mm H2O
0x63	INVALID	Channel is not configured.
0x82	VOLPROZ	Volume %
0x83	PPM	ppm
0x85	BAR	bar
0x88	MG_PER_KWH	mg/kWh

There are other units existing, the corresponding definitions can be provided if required.

6.4.3.5 0x3500 … 0x3518: Measurement Resolution

Data type in testo 350: 25 * BYTE

Data type on modbus: 25 * WORD

Access: Read only

These registers are used to gather the recommended display resolution for each displayed ViewValue. If an unused channel is requested, 0x80 (=-128) is replied to indicate an invalid channel.

Register mapping:

Register	Content
0x3500	ViewValue[0] Resolution code in LSB
0x3501	ViewValue[1] Resolution code in LSB
…	
0x3518	ViewValue[24] Resolution code in LSB

The recommended resolution is expressed as the exponent of the least significant digit to be shown.

Examples:

Code	Example
0	12
-1	12.1
-2	12.13

6.4.3.6 0x3600 … 0x3631: Dilution Factor

Data type in testo 350: 25 * single float (32 Bit)

Data type on modbus: 50 * WORD

Access: Read only

These registers are used to get the dilution factor used to measure the displayed value. This figure is applicable on gas values only. Dilution factors are represented in 32-bit float value (single precision float values). These float values are split into two 16-bit integer modbus registers.

If an unused channel is requested, 0xFFFFFFFF (=Not A Number) is replied to indicate an invalid channel.

Register mapping:

Register	Content
0x3200, MSB	ViewValue[0] Dilution Factor, Byte 3
0x3200, LSB	ViewValue[0] Dilution Factor, Byte 2
0x3201, MSB	ViewValue[0] Dilution Factor, Byte 1
0x3201, LSB	ViewValue[0] Dilution Factor, Byte 0
0x3202, MSB	ViewValue[1] Dilution Factor, Byte 3
0x3202, LSB	ViewValue[1] Dilution Factor, Byte 2
0x3203, MSB	ViewValue[1] Dilution Factor, Byte 1
0x3203, LSB	ViewValue[1] Dilution Factor, Byte 0
…	
0x3230, MSB	ViewValue[24] Dilution Factor, Byte 3
0x3230, LSB	ViewValue[24] Dilution Factor, Byte 2
0x3231, MSB	ViewValue[24] Dilution Factor, Byte 1
0x3231, LSB	ViewValue[25] Dilution Factor, Byte 0

To use these values, the two 16-bit values must be combined into one 32-bit value interpreted as single float.

Values are updated when main pump is running.

6.4.4 Error Messages

Testo 350 can handle an undefined number of errors at the same time. Each error message is regarded “active” from the moment of its creation until it has been confirmed (by the user or automatically by the application). On the modbus interface, the ten most serious active errors can be requested.

Each error is coded in one 16-bit wide register word. This word contains the following fields:

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Content	Category		Source			Group			Error Number							

6.4.4.1 Error code Interpretation at Device Error:

Bit 0…7: Error Number
Bit 8…10: Group/Slot:#

Code	Source “Device”		Source “Sensor n”	
	Mnemonic	Error caused …	Mnemonic	Error caused …
0x00	MEAS	while executing a measurement		
0x01	ADJUST	during adjustment	GENERAL	
0x02	SERVICE		INITING	during startup sequence
0x03	CONFIG	in configuration	ZEROING	during zeroing of the sensor
0x04	COMMUNICATION	from communication	MEASUREMENT	during measurement execution
0x05	SYSTEM	in general system function	ALIGNMENT	during alignment execution
0x06	ENVIRONMENT	from environment	LIFETIME	by lifetime surveillance

Bit 11,12: Category:

Code	Mnemonic	Description
0x03	CATEGORY_INTERNAL	Normally used for user guidance. Message appears, will be deleted from error memory when confirmed
0x00	CATEGORY_INFO	Used for warnings. Device is still working, but soon maintenance is recommended. Message can be closed in display but will remain in error memory.
0x01	CATEGORY_WARNING	Used for minor errors. Message cannot be closed in display.
0x02	CATEGORY_ERROR	Serious errors. Device cannot be operated any more, only switched off

6.4.4.2 Error code Interpretation at Sensor Error:

Bit 0…7: Error Number
Bit 8…10: Group/Slot:

Code	Mnemonic	Description
0x00	SOURCE_DEVICE	Error was caused from T350 device
0x01	SOURCE_GASSENSOR1	Error was caused from sensor in place 1

Code	Mnemonic	Description
0x02	SOURCE_GASSENSOR2	Error was caused from sensor in place 2
0x03	SOURCE_GASSENSOR3	Error was caused from sensor in place 3
0x04	SOURCE_GASSENSOR4	Error was caused from sensor in place 4
0x05	SOURCE_GASSENSOR5	Error was caused from sensor in place 5
0x06	SOURCE_GASSENSOR6	Error was caused from sensor in place 6
0x07	Unused	

Bit 15: Signals a sensor error.

6.4.4.3 Error codes for testo 350 device

Error codes of the testo 350 consist of a combination of the fields “Group” and “Error number”. Combining these two fields we get a unique number for each error.

Group	Error	Mnemonic	Comment
0x00	0x00	CO_EXCEEDED	Value for CO is exceeded
0x00	0x01	CO2_EXCEEDED	Value for CO_2 is exceeded
0x00	0x02	CONDENSATE_TRAP_WARNING	Condensate trap should be emptied
0x00	0x03	CONDENSATE_TRAP_FULL	Condensate trap full, pump will be stopped
0x00	0x04	COOLING_SYS_ERROR	Error in the gas cooling system
0x00	0x05	CXHY_NEGATIVE	C_xH_y is negative, zeroing required
0x00	0x06	LONGLIFE_NOT_ENABLED	Longlife sensor not allowed
0x00	0x07	MEAS_PRECISION_LOW	Measprecision getting low, Calibration recommended
0x00	0x08	NO_SENS_UNSTABLE	NO sensor after power down unstable, regeneration time till 2 hours possible
0x00	0x09	NO_TEST_GAS	No test gas, or the sensor is exhausted
0x00	0x0A	O2_VAL_ILLEGAL	Value for O_2 is illegal, Reboot device or change the sensor
0x00	0x0B	O2_SENS_CHANGE	O_2 sensor should be changed

Group	Error	Mnemonic	Comment
0x00	0x0C	OVERPRESSURE_TOO_HIGH	Overpressure at probe port is too high
0x00	0x0D	PROBE_IN_SERVICE	Probe should be sent into service
0x00	0x0E	PROBE_NOT_CONNECTED	Probe is not connected
0x00	0x0F	PROBE_HAS_CHANGED	Probe has connected
0x00	0x10	MAIN_PUMP_FLOW_TOO_LOW	Pump flow is too low
0x00	0x11	MAIN_PUMP_FLOW_TOO_HIGH	Pump flow is too high
0x00	0x12	RINSE_PUMP_ERROR	The rinse pump is defect
0x00	0x13	SENS_CHANGE	Sensor should be changed
0x00	0x14	SENS_DEFECT	Sensor is defect
0x00	0x15	SENS_TWICE	Two identical sensors plugged in
0x00	0x16	SENS_WRONG_SLOT	Sensor is plugged to the wrong slot
0x00	0x17	SENS_ALMOST_EXHAUSTED	Sensor is almost exhausted, Order new one
0x00	0x18	SENS_NOT_ENABLED	Sensor not enabled
0x00	0x19	SENS_TABLE_FULL	Sensor table complete
0x00	0x1A	SIGNAL_TOO_HIGH	Signal is too high, wait for regeneration
0x00	0x1B	SYSTEM_OVERHEATING	The board temperature is over 95 °C
0x00	0x1C	THRESHOLD_STOP_EXCEEDED	The stop threshold is exceeded
0x00	0x1D	THRESHOLD_ALARM_EXCEEDED	The alarm threshold is exceeded
0x00	0x1E	THRESHOLD_SAFETY_EXCEEDED	The sensor safety threshold is exceeded
0x00	0x1F	SENS_CUT_OFF_LEVEL_EXCEEDED	The sensor cut off level is exceeded
0x00	0x20	SENS_CUTOFF_LEVEL_EXCEEDED_MODUL2_CHAN1	The sensor cut off level is exceeded
0x00	0x21	SENS_CUTOFF_LEVEL_EXCEEDED_MODUL2_CHAN2	The sensor cut off level is exceeded
0x00	0x22	SENS_CUTOFF_LEVEL_EXCEEDED_MODUL3_CHAN1	The sensor cut off level is exceeded
0x00	0x23	SENS_CUTOFF_LEVEL_EXCEEDED_MODUL3_CHAN2	The sensor cut off level is exceeded

Group	Error	Mnemonic	Comment
0x00	0x24	DILUTION_PUMP_FLOW_TOO_HIGH	dilution pump flow is too high
0x00	0x25	DILUTION_PUMP_FLOW_TOO_LOW	dilution pump flow is too low
0x00	0x26	DEVICE_CALIB_ERR	error in device calibration, Servicing recommended
0x00	0x27	PERIST_PUMP_DEFECT	peristaltic pump defect, Servicing recommended
0x00	0x28	WATER_IN_CONDENSATE_GUARD	water in condensate guard, consider service note
0x00	0x29	DEVICE_THIN_FAKTOR_FAULT	Thin Faktors not ok
0x00	0x2A	DEVICE_TEMP_FAULT	Temperature of Device is Outside
0x00	0x2B	GAS_SENSOR_INTERNAL_ERROR	Gas sensor internal error (not used here!)
0x00	0x2C	DILUTION_MODULE_CALIBRATION	dilution error calibration
0x00	0x2D	TEMP_AT_IS_SAVED_AS_VT	if no VT is available, AT is saved as VT
0x00	0x2E	INFO_MEASPROG_FINISHED	info message for measprog finished
0x00	0x2F	CO2_PROBE_VALUE_TOO_HIGH	CO_2 probe concentration is too high
0x00	0x30	TEMP_VT_TOO_HIGH_A	VT is out of the expected range (only for country Austria)
0x00	0x32	INVALID_FIRMWARE_VERSION	Sensor has invalid firmware version
0x00	0x33	CO2_IR_WARMUP	CO_2 sensor temperature is too low
0x00	0x39	NUM_SENSORS	Missing gas sensors
0x00	0x57	DEVICE_TAMB_INSTABLE	Ambient temperature has changed during measurement
0x00	0x61	INFO_MEASPROG_FINISHED_NO_PROT	info message for measprog finished, but no measvalues where saved.
0x00	0x6C	DEVICE_TEMP_FAULT_PARTICLE	Temperature of device is out of range
0x00	0x78	SENS_RINSING_AKTIV	Sensor protection active

Group	Error	Mnemonic	Comment
0x00	0x79	DEVICE_DIFF_TEMP_FAULT	Difference temperature of Device is Outside (for IAS)
0x00	0x7A	O2_SENS_UNSTABLE	O_2 sensor after power down unstable, regeneration time till x hours/minutes possible
0x00	0x7B	SENSOR_ZERO_IMPOSSIBLE	Sensor zero impossible
0x00	0x7D	SENSOR_ZERO_NOT_READY	Sensor isn't ready to zero
0x01	0x00	DEVICE_ELADJUST_CHECKSUM	Electrical adjustment
0x01	0x01	DEVICE_CROSSDATA_DRAUGHT_CHECKSUM	Cross data draught
0x01	0x02	DEVICE_CROSSDATA_PUMP_CHECKSUM	Cross data pump
0x01	0x03	DEVICE_CTRLDATA_PUMP_CHECKSUM	Control data pump
0x01	0x04	DEVICE_CTRLDATA_PUMPFLOW_CHECKSUM	Control data pumpflow
0x01	0x05	DEVICE_CTRLDATA_DEFAULT_CHECKSUM	Control data default data: for example: gaswayfactor
0x01	0x06	DEVICE_CTRLDATA_DILUTION_PUMP_CHECKSUM	Control data dilution - pump
0x01	0x07	DEVICE_CTRLDATA_DILUTION_PUMPFLOW_CHECKSUM	Control data dilution - pumpflow
0x01	0x09	DEVICE_ELADJUST_MISSING	Electrical adjustment not performed yet
0x01	0x0F	DEVICE_CTRLDATA_KFAKTOREN_CHECKSUM	Control data KFactors
0x02	0x00	DEVICE_IN_SERVICE	Device should be sent to the service
0x02	0x01	DEVICE_SERVICE_UPDATE_DETECTED	Software update was detected
0x03	0x00	DEVICE_KONFIG_CFG_CHECKSUM	Checksum wrong for device configuration data
0x03	0x01	DEVICE_KONFIG_SENS_FREEING_CHECKSUM	Checksum wrong for gas sensors allowances
0x03	0x02	DEVICE_KONFIG_CFG_BASE_CHECKSUM	Configuration Checksum wrong for basic device data
0x03	0x03	DEVICE_KONFIG_ERROR_CONTROL	Checksum wrong for control probe heating

Group	Error	Mnemonic	Comment
0x03	0x04	DEVICE_KONFIG_CFG_DILUTION_MOTOR_FAULT	Checksum wrong for dilution settings
0x04	0x00	MEMORY_DEFECT	Memory is defect
0x04	0x01	MEMORY_FULL	Memory is full
0x04	0x02	PROTOCOL_ERROR_CREATE	Protocol error. Creation failed.
0x04	0x03	FOLDER_ERROR_CREATE	Folder creation failed
0x04	0x04	MEASPLACE_ERROR_CREATE	Measplace creation failed
0x04	0x05	PROTOCOL_ERROR_ADD	Protocol error while reading
0x04	0x06	PROTOCOL_ERROR_READ	Protocol error while reading
0x04	0x07	PROTOCOL_ERROR_REMOVE	Protocol could not be removed
0x04	0x08	FOLDER_ERROR_GET	Folder error GET data
0x04	0x09	FOLDER_ERROR_SET	Folder error SET data
0x04	0x0A	FOLDER_ERROR_REMOVE	Folder could not be removed
0x04	0x0B	FOLDER_ERROR_REMOVE_CONT	Folder content could not be removed
0x04	0x0C	FOLDER_ERROR_PASTE	Folder error paste
0x04	0x0D	MEASPLACE_ERROR_GET	Measplace error GET data
0x04	0x0E	MEASPLACE_ERROR_SET	Measplace error SET data
0x04	0x0F	MEASPLACE_ERROR_REMOVE	Measplace could not be removed
0x04	0x10	MEASPLACE_ERROR_REMOVE_CONT	Measplace content could not be removed
0x04	0x11	MEASPLACE_ERROR_PASTE	Measplace error paste
0x04	0x12	MEASPLACE_ERROR_EXTRAS	Measplace error extra data
0x04	0x13	MEMORY_10PERCENT_LEFT	Less than 10% of memory is left
0x04	0x14	MEMORY_NOT_ENOUGH_MEMORY_LEFT	Not enough memory to start the action
0x04	0x15	PROTOCOL_TOO_BIG	Memory is full (Protocol)
0x05	0x00	BLUETOOTH_MAX_NUM_OF_PAIR_DEVICES	max number of pair devices reached
0x05	0x01	IDENTICAL_CAN_IDS	CAN identical IDs
0x05	0x02	CAN_QUEUE_FULL	CAN queue is full
0x05	0x03	DEVICE_CFG_STARTUP	Config EEPROM: Startup communication error
0x06	0x00	ACCU_UNDER_10_PC	Battery capacity under 10%

Group	Error	Mnemonic	Comment
0x06	0x01	ACCU_UNDER_20_PC	Battery capacity under 20%
0x06	0x02	ACCU_UNDER_30_PC	Battery capacity under 30%
0x06	0x03	ACCU_LOW	Battery capacity low
0x06	0x04	ACCU_VERY_LOW	Battery capacity very low
0x06	0x05	LEAKAGE_SENS_ERROR	Error in the leakage sensor
0x06	0x06	OPPERATING_HOURS_1000	1000 operating hours achieved; Servicing recommended
0x06	0x07	ACCU_TEMP_OUTRANGE_CHARGE	Battery temperature out of range for charging
0x06	0x0E	ACCU_CELL_VOLTAGE_OUTRANGE	Battery LiIon cell voltage out of range (Battery defect)
0x06	0x10	WATCHDOG_DETECTED	Watchdog reset detected
0x06	0x11	EEPROMACCESS_INTERN	Access to internal EEPROM failed
0x06	0x14	ACCU_PROTECTION_RELEASED	Battery protection released, serious battery problem

Not listed error codes on request.

6.4.4.4 Error Codes for the Gas Sensors

For the gas sensors the field “Group” indicates the sensor status when the error occurred. For the gas sensors the numbers in field “Error” are already unique, this number is sufficient to distinguish between all possible errors. Field “Group” can be used as additional information.

Most of these errors shall not occur on series sensors, because they are caused by wrong firmware or wrong configuration. Errors which may appear in the field are marked by color in the table below.

Code	Mnemonic	Comment
0x01	GS_ERRNUM_CHECKSUM	Checksum error in EEPROM data
0x02	GS_ERRNUM_EEPROM_WRITE	Error during writeback to EEPROM
0x03	GS_ERRNUM_EEPROM_READ	General EEPROM read error
0x0A	GS_ERRNUM_EEPROM_MAX_AUXCHANNEL S	Too many auxiliary channels requested for this EEPROM dataset
0x0B	GS_ERRNUM_MAX_PCBAUXCHANNELS	Too many auxiliary channels requested for this hardware
0x0C	GS_ERRNUM_MAX_PCBOUTCHANNELS	Too many outchannels requested for this hardware

Code	Mnemonic	Comment
0x0D	GS_ERRNUM_MAX_PCBGASCHANNELS	Too many gaschannels requested for this hardware
0x0E	GS_ERRNUM_AUX_INCHANNELIDX	Invalid inchannelindex for auxiliary channel
0x0F	GS_ERRNUM_AUX_OUTCHANNELIDX	Invalid outchannelindex for auxiliary channel
0x10	GS_ERRNUM_EEPROM_MAX_GASCHANNEL S	Too many gaschannels requested for this EEPROM dataset
0x11	GS_ERRNUM_GAS_INCHANNELIDX	Invalid inchannelindex for gaschannel
0x12	GS_ERRNUM_GAS_OUTCHANNELIDX	Invalid outchannelindex for gaschannel
0x13	GS_ERRNUM_COEFF_NUMCROSS	Too many cross gases for gas channel
0x14	GS_ERRNUM_NUMEXTERNGASES	Too many extern gases requested
0x15	GS_ERRNUM_MISSING_COMPANION	Required associated channel (i.e. temperature) not found
0x16	GS_ERRNUM_CALCSCHEME	Unknown calcscheme for gas calculation
0x17	GS_ERRNUM_EALIGN_MISSING	Not enough alignment channels
0x18	GS_ERRNUM_EALIGN_HWINDEX	Invalid hardware channel index in alignment data
0x19	GS_ERRNUM_EALIGN_ADCGAIN	Invalid ADC gain selection
0x1A	GS_ERRNUM_EALIGN_COEFF	Invalid coefficients in E-Alignment section
0x1B	GS_ERRNUM_FORMULA_IDINVALID	Invalid formula ID
0x1C	GS_ERRNUM_FORMULA_COEFF	Error in formula coefficients
0x1D	GS_ERRNUM_EXTPOL_HISTOBUFSIZE	Requested extrapolation history buffer too big or 0
0x1E	GS_ERRNUM_GALIGN_CALCERR	Gas alignment entries are causing numerical errors
0x1F	GS_ERRNUM_FILTER_NAN	Invalid entries in filter surveillance section
0x20	GS_ERRNUM_RECAL_NAN	Invalid entries in recalibration surveillance section
0x21	GS_ERRNUM_STAT_NAN	Statistics partially unavailable due to corrupted EEPROM content

Code	Mnemonic	Comment
0x22	GS_ERRNUM_GALIGN_NUMERICERROR	Numeric overflow in calculation of gas alignment
0x23	GS_ERRNUM_FMLA_NUMERICERROR	Numeric overflow in calculation of gas formula
0x24	GS_ERRNUM_HW_FW_MISMATCH	Hardware platform does not match to firmware configuration
0x25	GS_ERRNUM_HW_EEP_MISMATCH	Hardware platform does not match to EEPROM entry
0x26	GS_ERRNUM_FW_EEP_MISMATCH	Firmware configuration does not match to EEPROM entry
0x27	GS_ERRNUM_EEP_STRUCT_MISMATCH	Firmware configuration does not match to EEPROM structure
0x28	GS_ERRNUM_EEP_FW_RELEASE	Firmware release too low
0x32	GS_ERRNUM_ZERO_SENSHIGH	Sensitivity is too big
0x33	GS_ERRNUM_ZERO_SENSLOW	Sensitivity is too low
0x34	GS_ERRNUM_ZERO_SENSLOW_WARN2	Sensitivity is less than warning threshold 2
0x35	GS_ERRNUM_ZERO_SENSLOW_WARN1	Sensitivity is less than warning threshold 1
0x36	GS_ERRNUM_ZERO_ABSOLUT	Sensor output during zeroing too high
0x37	GS_ERRNUM_ZERO_DELTA	Sensor output during zeroing unstable
0x38	GS_ERRNUM_ZERO_O2REF	O_2 reference value invalid
0x39	GS_ERRNUM_ZERO_NOSAVE	Tried to zero without previous SAVE
0x3A	GS_ERRNUM_ZERO_NUMERICERROR	Numeric error during zeroing
0x3B	GS_ERRNUM_ZERO_UNUSED	
0x3C	GS_ERRNUM_FILTERLIMIT	Gas filter depleted
0x3E	GS_ERRNUM_RECAL_RECOMM	Recalibration recommended
0x3F	GS_ERRNUM_PELI_OVERLOAD	Overload of an pellistor, sensor supply switched off
0x40	GS_ERRNUM_FILTER_NALLOWED	Reset of filter counter not allowed
0x41	GS_ERRNUM_CMDSIZE	Wrong command size
0x42	GS_ERRNUM_ALIGNCHANNEL	Invalid channel number in alignment command

Code	Mnemonic	Comment
0x43	GS_ERRNUM_ALIGNTYPE	Invalid alignment type in alignment command
0x44	GS_ERRNUM_ALIGNMODE	Invalid operation in current alignment mode
0x45	GS_ERRNUM_ALIGNINDEX	Invalid index number in alignment command
0x46	GS_ERRNUM_TEMPERATUREINDEX	Invalid temperature channel index
0x47	GS_ERRNUM_NOSTATISTICS	No alignment statistic possible due to corrupted eeprom content
0x48	GS_ERRNUM_NOTIMPLEMENTED	Operation not implemented
0x49	GS_ERRNUM_ALIGNDENIED	Alignment denied, not allowed in current sensor state
0x4A	GS_ERRNUM_CROSSFORCEDTO1	Cross compensation is forced to 1.0 as result of invalid input data
0x4B	GS_ERRNUM_RECAL_NOSAVE	Recalibration will not be saved for non-toxic sensors

6.4.4.5 0x4000: Number of active messages

Data type in testo 350: Byte

Data type on modbus: WORD

Access: Read only

This register can be used to request the number of currently active messages (errors, warnings, information). It should be read before reading the message queue.

6.4.4.6 0x4001 … 0x400A: Messages

Data type in testo 350: 10 * WORD

Data type on modbus: 10 * WORD

Access: Read only

These registers can be used to request all currently active messages (errors, warnings, information). The messages are sorted by importance in descending order.

Each register contains one message coded as described above. If more registers than active errors are requested, the undefined registers will be filled with 0.

6.4.4.7 0x4010 … 0x401A : Error Messages in ASCII

Data type in testo 350: ---

Data type on modbus: 60 * WORD (corresponding to a 120-character string of the error text)

Access: Read only

This error is output in a readable string with error number and sensor slot number.

The command can be used to get an ASCII text of the error.

E.g.:

If 0x4000 (number of active Messages) responds 2:

4010 delivers 1 message text

4011 delivers 2 message text

…

if no warning/error exists, 0 are returned.

The error/warning text is embedded in /* */ , like comments in C

E.g. :

Device error :

xxxxxxxxxx/*Error: #1026 : error in device calibration, Servicing recommended*/xxxxxxxxxx

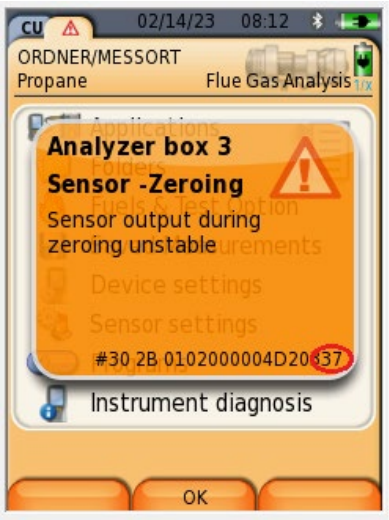
Device Warning:

xxxxxxxxxx/*Warning: #0008 : NO-Sensor after power down unstable, regeneration time till 2hours possible*/xxxxxxxxxx 22

Error at Sensor (slot 3)

xxxxxxxxxx/*Sensor 3 Error: #9b36 : Sensor output during zeroing too high*/xxxxxxxxxx

Corresponding error screen at control unit: Error number is last 2 digits of displayed number:



6.4.4.8 0x4020: Set Ubias warning confirmed

Data type in testo 350: 1 * BYTE

Data type on modbus: 1 * WORD

Access: Write only

Value: 0x5A / all other values are not accepted.

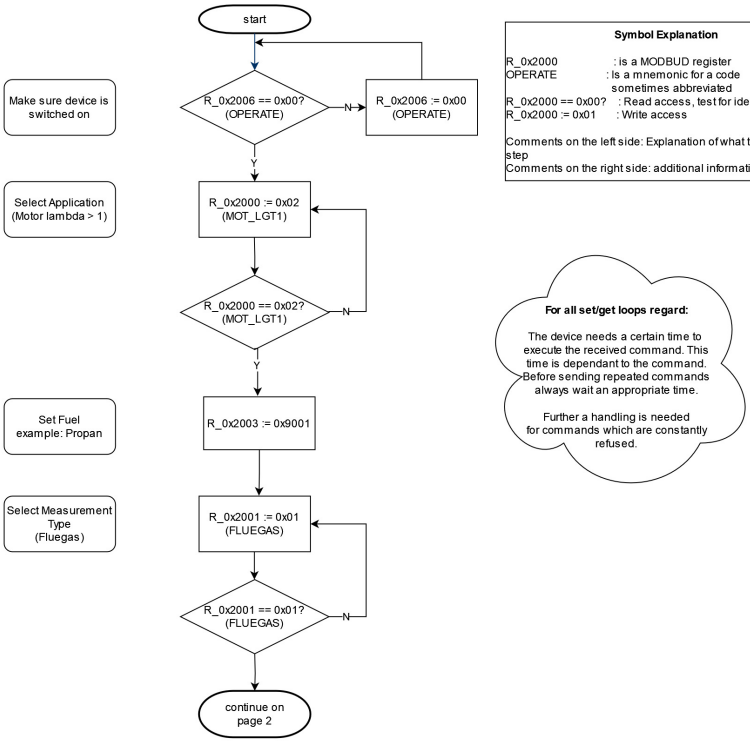
In cases where no internal battery is used, this is a possibility to reset the warning of a mains voltage interruption.

If the Warning occurs:

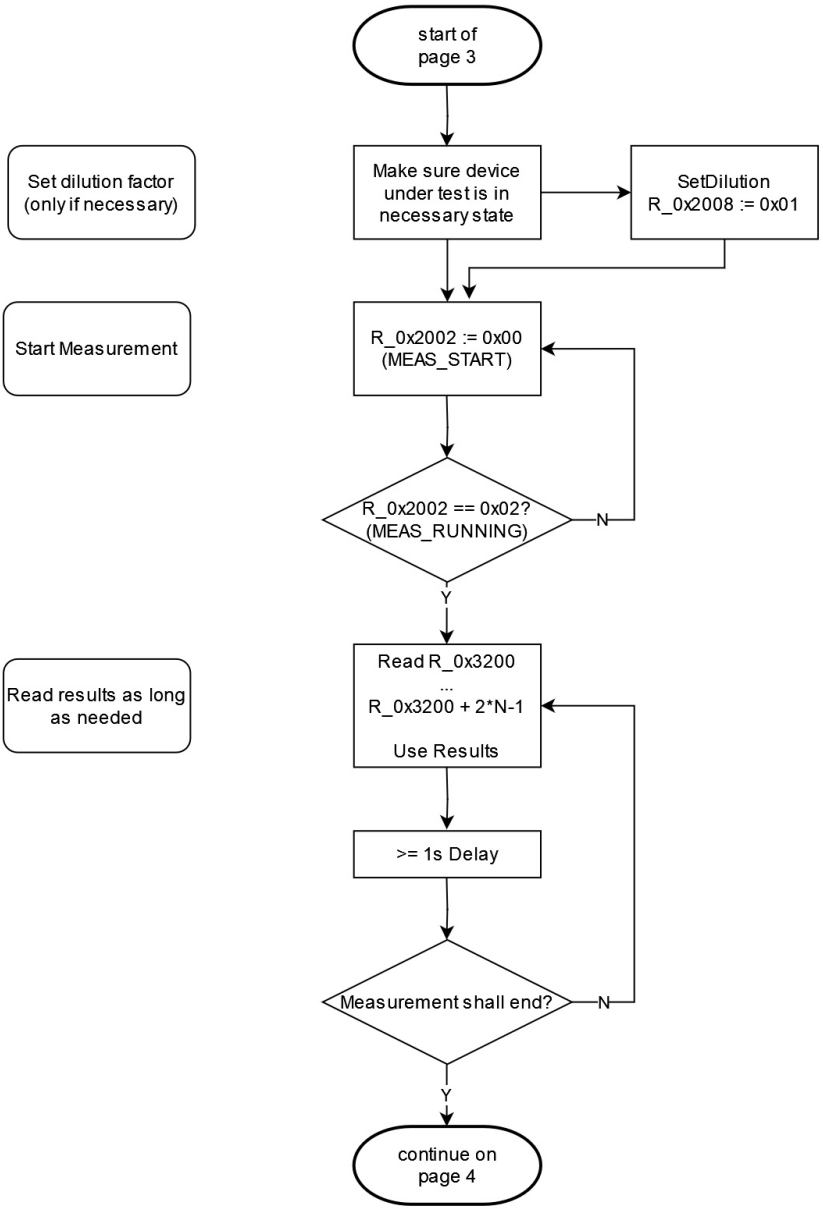
It is up to the user to react accordingly to the warning. If the corresponding waiting times are not observed, this can lead to incorrect measurement.

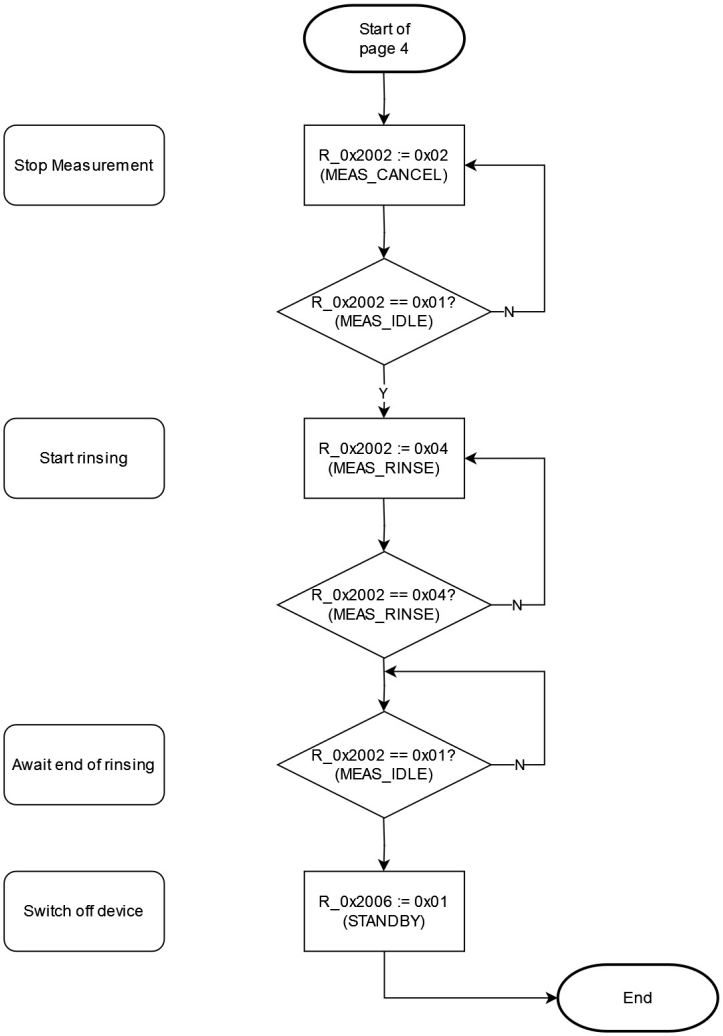
6.5 Example

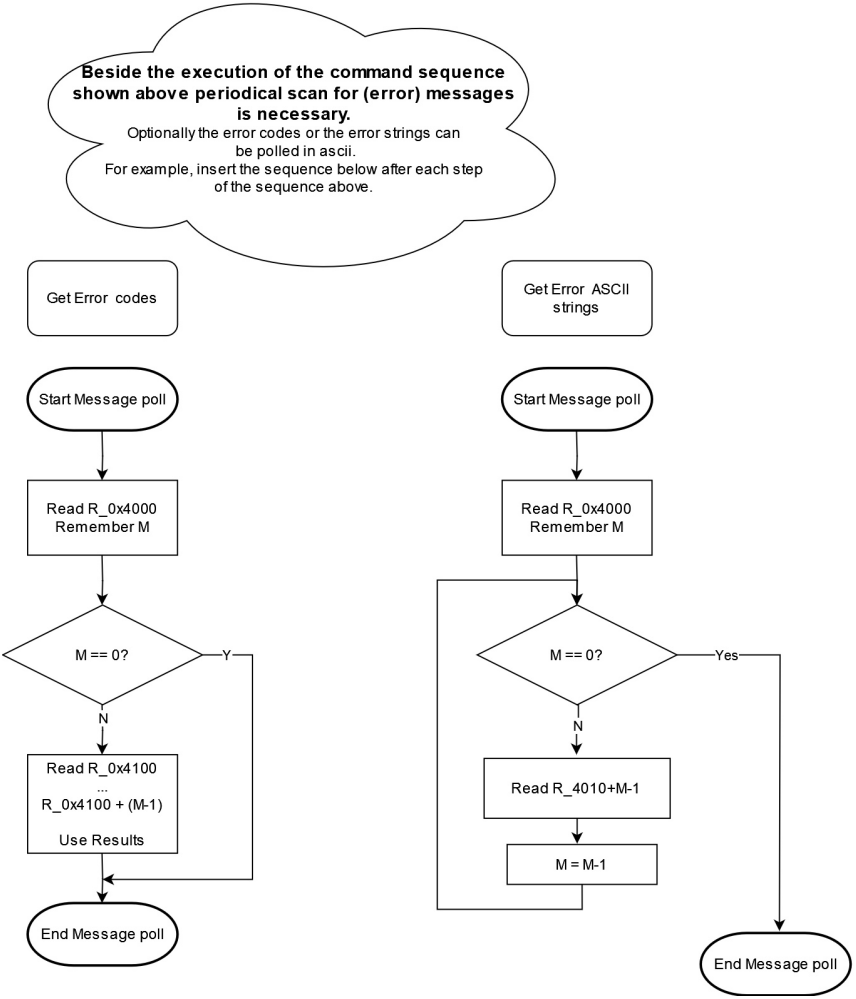
The flowchart shown on the next pages shows a typical application for a modbus controlled device. The example shows the necessary commands to measure the flue gas components of a motor.




Start of page 2
Prepare Measurement
R_0x2002 == 0x01? (MEAS_IDLE)
N
R_0x2002 := 0x02 (MEAS_CANCEL)
Start Zeroing
R_0x2002 := 0x03 (MEAS_ZERO)
If 0x03 (MEAS_ZERO) is used, zeroing is only proceeded when no zeroing has been executed before.
Usage of 0x06 (MEAS_ZERO_FORCE) will start zeroing in any case
R_0x2002 == 0x03? (MEAS_ZERO)
N
Y
Wait for end of Zeroing
R_0x2002 == 0x01? (MEAS_IDLE)
N
During zeroing the ViewChannel sequence changes to a default setting.
Normally it is not necessary to read measured values during zeroing.
If needed, ViewChannel sequence must be checked.
Y
Read ViewChannel Sequence
Read R_0x3000 Remember N
Read R_0x3100
...
R_0x3100 + 2*N-1
Remember channel sequence
Knowledge of units and resolutions is assumed for this example
continue on page 3


start of page 3
Set dilution factor (only if necessary)
Make sure device under test is in necessary state
SetDilution R_0x2008 := 0x01
Start Measurement
R_0x2002 := 0x00 (MEAS_START)
R_0x2002 == 0x02? (MEAS_RUNNING)
N
Y
Read results as long as needed
Read R_0x3200 ... R_0x3200 + 2*N-1
Use Results
>= 1s Delay
Measurement shall end?
N
Y
continue on page 4


Start of page 4
Stop Measurement
R_0x2002 := 0x02 (MEAS_CANCEL)
R_0x2002 == 0x01? (MEAS_IDLE)
N
Y
Start rinsing
R_0x2002 := 0x04 (MEAS_RINSE)
R_0x2002 == 0x04? (MEAS_RINSE)
N
Await end of rinsing
R_0x2002 == 0x01? (MEAS_IDLE)
N
Switch off device
R_0x2006 := 0x01 (STANDBY)
End


Beside the execution of the command sequence shown above periodical scan for (error) messages is necessary.
Optionally the error codes or the error strings can be polled in ascii.
For example, insert the sequence below after each step of the sequence above.
Get Error codes
Start Message poll
Read R_0x4000
Remember M
M == 0?
Y
N
Read R_0x4100
...
R_0x4100 + (M-1)
Use Results
End Message poll
Get Error ASCII strings
Start Message poll
Read R_0x4000
Remember M
M == 0?
Yes
N
Read R_4010+M-1
M = M-1
End Message poll

Testo SE & Co. KGaA
Celsiusstr. 2
79822 Titisee-Neustadt
Germany
Phone: +49 (0)7653 681-0
E-mail: info@testo.de
www.testo.com

0970 3570 en 01 – 10.2023