User manual
myDatalogMUC

Valid from:

- **Firmware version: 03v002**
- **Modem Version: 03v014**
- **Server version: 44.1**
- **Hardware version: 1.0**

Microtronics
We live M2M

Chapter 1 Table of contents

Chapter 2 Declaration of conformity

EU-Konformitätserklärung
EU Declaration of Conformity / Déclaration de conformité UE

Produktbezeichnung: Product: Désignation du produit:	Stationäres Instrument zur Aufzeichnung und Übertragung von Messdaten und Ausführung verschiedenster Steuer- und Regelaufgaben
Type : Type code: Type:	**myDatalogMUC**

Microtronics
We live M2M

Hersteller: Manufacturer : Fabricant:	**Microtronics Engineering GmbH** Hauptstrasse 7 A-3244 Ruprechtshofen

Das bezeichnete Produkt stimmt mit den folgenden Europäischen Richtlinien überein.
The designated product is in conformity with the following european directives.
Le produit décrit est conforme aux directives européennes suivantes.

		Europäische Norm	Ausgabedatum
(2014/30/EU)	**EMC Directive**	ÖVE/ÖNORM EN61326-1	2013-08
(2014/35/EU)	**LVD Directive**	ÖVE/ÖNORM EN61010-1	2011-08
(2014/53/EU)	**RED Directive**	ÖVE/ÖNORM EN301489-1 V1.9.2 ÖVE/ÖNORM EN301489-7 V1.3.1 ÖVE/ÖNORM EN301511 V9.0.2	2011-09 2006-02 2003-06
(2011/65/EU)	**RoHS Directive**	ÖVE/ÖNORM EN 50581	2013-03

Ruprechtshofen, den 19.06.2017

Ort und Datum der Ausstellung
Place and date of issue
Lieu et date d'établissement

Hans-Peter Buber, Managing Director
Unterschrift
name and signature of authorised person
Nom et signature de la personne autorisée

Chapter 3 Technical data

Voltage supply	12...30VDC (+/-10%) Additional information is provided in "Technical details regarding the energy supply" on page 70.
Power consumption	typ. 1W (without sensors) max. 3W (without sensors)
Integrated rechargeable buffer battery	LiPo rechargeable battery with 900mAh to issue a message in the event of a power supply failure. Additional information is provided in "Technical details about the integrated rechargeable buffer battery" on page 68.
Housing	Material: Lexan/Noryl, UL94-V0 Weight: 320g Protection class: IP20 Dimensions (WHD): 157 x 86 x 64mm (without antenna)
Operating temperature	-20...+60°C
Air humidity	15...90%rH non-condensing
Storage temperature	-20...+60°C
Charging temperature (rechargeable buffer battery)	0...+40°C
Display	LED for indicating the operating state and error codes LED to display the self-testing result LED to indicate that the device is being supplied by the rechargeable buffer battery LEDs to indicate the switching states of the relays
Operation	Button to initiate ALOHA transmission mode Button to initiate self-testing
Antenna connector	SMA

Universal inputs	8 x analogue or digital Modes: 0...20mA: Resolution 6,3µA, max. 23,7mA, load 96Ω 4...20mA: Resolution 6,3µA, max. 23,7mA, load 96Ω 0...2V: Resolution 610µV, max. 2,5V, load 10k086 0...10V: Resolution 7,97mV, max. 32V, load 4k7 - PWM: 1...99%, max. 100Hz, min. pulse length 1ms, max. 32V, load 4k7 - Frequency: 1...1000Hz, max. 32V, 4k7 - Digital: low <1,31V, high >2,61V, max. 32V, load 4k7 - Day counter: min. pulse length 1ms, max. 32V, load 4k7 - Interval counter: min. pulse length 1ms, max. 32V, load 4k7 Additional information is provided in "Technical details about the universal inputs" on page 62.
External temperature sensor	1 x PT100/1000 (including auto detection) Additional information is provided in "Technical details about the PT100/1000 interface" on page 63.
Modbus	2 x RS485 switchable between modbus master and modbus slave - Baud rate: 300-115200 - Stop bits: 1,2 - Parity: N, E, O - Data bits: 7,8 Modes: - RTU - ASCII Number of processable interface channels - Inputs: modbus channels + serial channels = 64 - Outputs: modbus channels + serial channels = 64 Additional information is provided in "Technical details about the modbus interfaces (Com1, Com3)" on page 63.

Serial interface	1 x RS232 for the connection of a digital sensor
	• Baud rate: 300-115200
	• Stop bits: 1,2
	• Parity: N, E, O
	• Data bits: 7,8
	Modes:
	• ASCII
	Number of processable interface channels
	• Inputs: modbus channels + serial channels = 64
	• Outputs: modbus channels + serial channels = 64
	Additional information is provided in "Technical details about the RS232 interface (Com2)" on page 65.
Outputs	6 x relays (2 groups for various potentials)
	• U:240VAC
	• U_{max}: 400VAC
	• I_{max}:3A
	2 x analogue output (not galvanically isolated)
	• Load voltage: Corresponds to the supply voltage (12...30VDC , +/-10%)
	• max. load (32V, 20mA): 1200Ω
	• $I_{out\ max}$: 20mA
	Modes:
	• 4-20mA
	• 0-20mA
	Additional information is provided in "Technical details about the outputs" on page 67.
Data memory	Internal flash memory for up to 4.470 measurement cycles
	Additional information is provided in"Functionality of the internal data memory" on page 40.
Data type	f32 (32Bit floating point)
	Exception: Modbus register (see "format" configuration parameters in the chapter "Interface channels 1-32" on page 110)
Data transmission	By means of GSM/GPRS quad-band modem or per LAN interface to the respective myDatanet server
SIM	The myDatalogMUC is equipped with an integrated SIM chip.
Monthly data volume	2,0MB at 2 min. measurement cycle and 120 min. transmission cycle (only the 8 universal inputs are active)
	Additional information is provided in "Basic setting" on page 134.

Chapter 4 General specifications

The information in this manual has been compiled with great care and to the best of our knowledge. The manufacturer, however, assumes no liability for any incorrect specifications that may be provided in this manual. The manufacturer is not responsible for direct, indirect, accidental or consequential damages which arise from errors or omissions in this manual even if advised of the possibility of such damages. In the interest of continuous product development, the manufacturer reserves the right to make improvements to this manual and the products described in it at any time and without prior notification or obligation.

> *Note: The specifications in this manual are valid as of the versions listed on the front page. Revised versions of this manual, as well as software and driver updates are available in the service area of the myDatanet server.*

4.1 Translation

For deliveries to countries in the European Economic Area, the manual must be translated into the language of the respective country. If there are any discrepancies in the translated text, the original manual (German) must be referenced or the manufacturer contacted for clarification.

4.2 Copyright

The copying and distribution of this document as well as the utilisation and communication of its contents to others without express authorisation is prohibited. Contraventions are liable to compensation. All rights reserved.

4.3 General descriptive names

The use of general descriptive names, trade names, trademarks and the like in this manual does not entitle the reader to assume they may be used freely by everyone. They are often protected registered trademarks even if not marked as such.

4.4 Safety instructions

For the connection, commissioning and operation of the myDatalogMUC , the following information and higher legal regulations of the country (e.g. ÖVE), such as valid EX regulations as well as the applicable safety and accident prevention regulations for the respective application case must be observed.

Read this manual completely before unpacking, setting up or operating this device. Observe all hazard, danger and warning information. Non-observance can lead to serious injuries to the operator and/or damage to the device.

Ensure that the safety equipment of this measurement instrument is not impaired. Install and use the measurement system only in the manner and method described in this manual.

> *Important note: The product is not approved for use outdoors as it is not protected from penetrating moisture and only provides minimal protection against the ingression of dust.*

4.4.1 Use of the hazard warnings

DANGER:
Indicates a potential or threatening hazardous situation that will result in death or serious injuries if not avoided.

WARNING:
Indicates a potential or threatening hazardous situation that can result in death or serious injuries if not avoided.

CAUTION:
Indicates a potential hazardous situation that can result in minor or moderate injuries or damage to this instrument.

Important note: Indicates a situation that can result in damages to this instrument if it is not avoided. Information that must be particularly emphasised.

Note: Indicates a situation that does not result in any injury to persons.

Note: Information that supplements the specifications in the main text.

4.4.2 General safety instructions

WARNING:
Hazardous electric voltage can cause electric shock or burns. Always switch off all of the used power supplies for the device before installing it, completing any maintenance work or resolving any faults.

WARNING:
Never use this device in areas where the use of wireless equipment is prohibited. The device must not be used in hospitals and/or in the vicinity of medical equipment, such as heart pacemakers or hearing aids, as their functionality could be compromised by the GSM/GPRS modem contained in the device.

WARNING:
Never use this device in potentially explosive atmospheres and in the vicinity of highly combustible areas (fuel stations, storage areas for combustible material, chemical plants and detonation sites) or in the vicinity of flammable gases, vapours or dust.

4.4.3 Safety and preventative measures for handling GSM/GPRS modems

The following safety and preventative measures must be observed during all phases of installation, operation, maintenance or repair of a GSM/GPRS modem. The manufacturer is not liable if the customer disregards these preventative measures.

CAUTION:
The GSM/GPRS modem connection must not be used in hazardous environments.

No guarantee of any kind, whether implicit or explicit, is given by the manufacturer and its suppliers for the use with high risk activities.

In addition to the following safety considerations, all directives of the country in which the device is installed must be complied with.

> *Important note: No liability shall be assumed at any time and under no circumstances for connections via a GSM/GPRS modem for which wireless signals and networks are utilized. The GSM/GPRS modem must be switched on and operated in an area where sufficient signal strength is present.*

4.4.3.1 Safety and precautionary measures for the installation of the GSM/GPRS modem

- This device must only be installed by a trained technician who applies the recognised installation practices for a radio frequency transmitter including the correct grounding of external antennas.
- The device must not be operated in hospitals and/or in the vicinity of medical equipment such as heart pacemakers or hearing aids.
- The device must not be operated in highly flammable areas such as petrol filling stations, fuel storage sites, chemical factories and explosion sites.
- The device must not be operated in the vicinity of flammable gases, vapours or dusts.
- The device must not be subjected to strong vibrations or impacts.
- The GSM/GPRS modem can cause interferences if it is located in the vicinity of television sets, radios or computers.
- Do not open the GSM/GPRS modem. Any modification to the equipment is prohibited and will result in the operating licence being revoked.
- The use of GSM services (SMS messages/data communication/GPRS, etc.) may incur additional costs. The user is solely responsible for any resulting damages and costs.
- Do not install the device in any other way to the one described in the operating instructions. Improper use will invalidate the warranty.

4.4.3.2 Safety measures for installing the antenna

- Only use antennas that are recommended or supplied by the manufacturer.
- The antenna must be installed at a distance of at least 20 cm from individuals.
- The antenna must not be extended outside protected buildings and must be protected against lightning strikes.
- The voltage supply must be switched off before replacing the antenna.

4.5 Overview
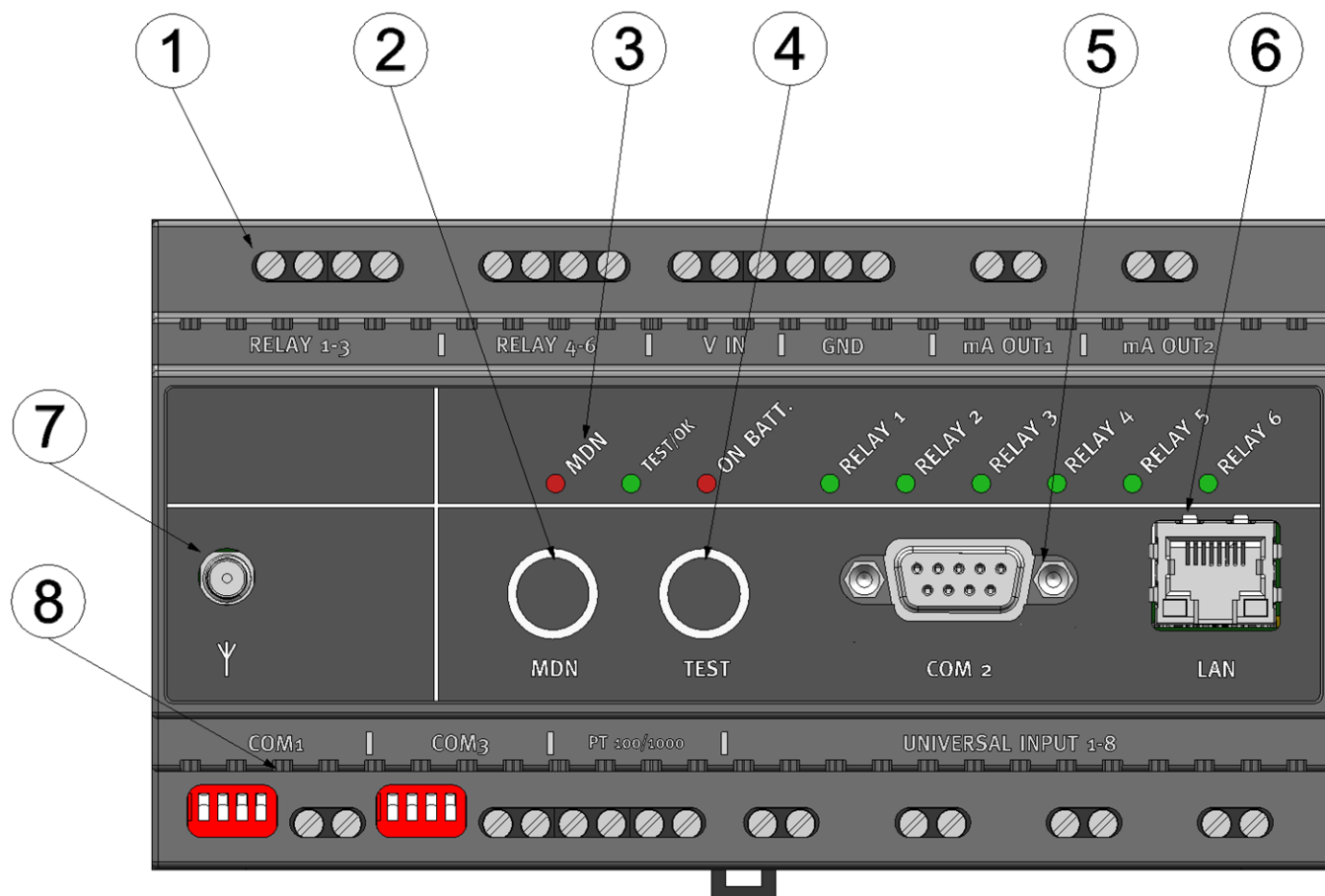
Overview myDatalogMUC

1	Upper connection strip	5	Sub-D connector of the serial interface (Com2)
2	Button to initiate ALOHA transmission mode	6	LAN interface
3	Signalling LEDs	7	Antenna connector
4	Button to initiate self-testing	8	Lower connection strip

4.5.1 Block diagram

Important note: *The current firmware does not support the hardware handeshake of the RS232 interface (CTS, RTS).*
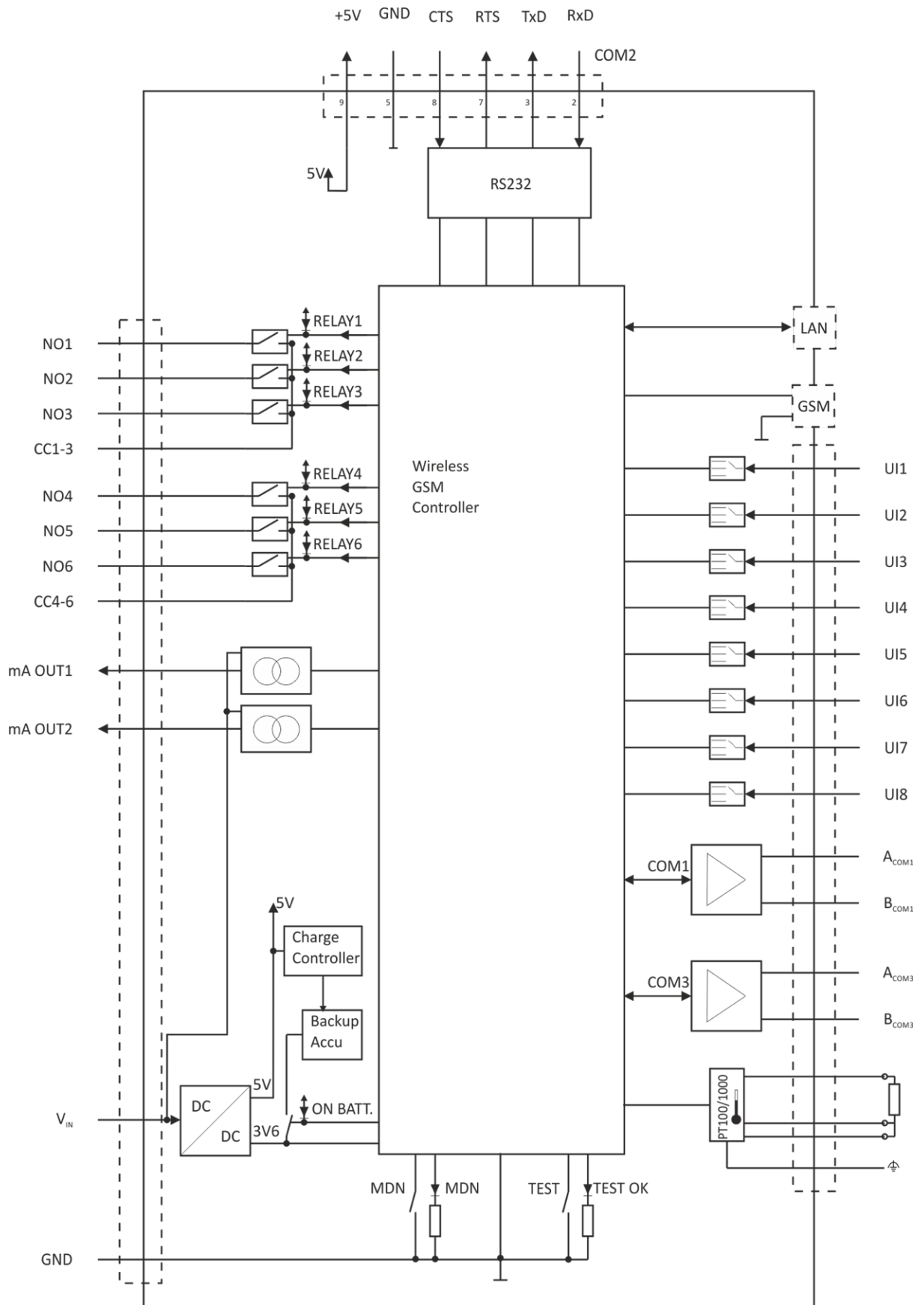
Block diagram of the myDatalogMUC

4.6 Intended use

The stationary measurement device provides a universal solution for a variety of signal recording and control tasks. The device requires a continuous power supply. The measured and recorded data is stored on a non-volatile memory medium. This stored data is sent via the mobile network to a central server for further processing. The device is equipped with an integrated SIM chip for this purpose. The maximum permissible limit values specified in chapter "Technical data" on page 15 must be observed. The manufacturer shall not be liable for any operational cases that deviate from these limit values and have not been approved by the manufacturer in writing.

> *Note: This device is exclusively intended to be used for the purposes as described before. Any other use or use beyond what is specified or a modification of the device shall be deemed to be not for the intended purpose and is not permitted without the express written consent of the manufacturer. The manufacturer shall not be held liable for any damages that may result from such unauthorised use or modification. The operator alone bears the associated risk.*

> *Note: The manufacturer is not liable for data loss of any kind.*

> *Note: The integrated SIM chip provides a mobile communications connection to a variety of international service providers. In order to be able to utilise all functions of the device, you must ensure that the device is located in the service area of one of these service providers. You can find a list of all supported countries and associated service providers under www.microtronics.com/footprint. A Managed Service contract with Microtronics Engineering GmbH is required for use of the mobile data transmission (see www.microtronics.com/managedservice). This includes the provisioning of the mobile communications connection via the network of the service provider included in the above-mentioned list.*

4.7 General product information

The device is a stationary device that can be used for a variety of control and regulating tasks in addition to recording and transmitting measurement data from various sources.

The following interfaces are available for recording measurement data:

- 8 universal inputs that can be operated in various analogue and digital modes
- An interface for connecting a PT100 or PT1000 including automatic detection of which type is being used
- 2 RS485 interfaces that can be configured as a modbus master or modbus slave
- A RS232 interface to which a digital sensor can be connected that provides its data in ASCII format separated in blocks by means of a delimiter

It should be noted that there are only 64 channels available for recording the measurement data read from the 3 serial interfaces (2 x RS485, 1 x RS232). These 3 serial interfaces can also be used to return data to the sensors and actuators. A total of 64 channels are provided for this purpose. The output data can either be specified via the input screen of the myDatanet server or can be calculated by the myDatalogMUC after each measurement cycle via the script (see "Pawn script" on page 149).

In addition to 2 0/4-20 mA analogue outputs, there are also 6 relays, of which, every 3 have a joint root (see "Technical details about the outputs" on page 67), that are also available for the output of regulating and control commands. The output value of the analogue outputs can either be assigned wirelessly from a central location or calculated by the myDatalogMUC by means of a script (see "Pawn script" on page 149). The relays can be configured so that they can each be switched by the device itself prior to a measurement (to

supply a sensor) or so that they can be switched wirelessly from a central location. The myDatalogMUC can also determine the setpoints for the relays itself by means of the script.

All of the input measurement data is temporarily saved to an internal data memory along with the output states and wirelessly transmitted to a central location at freely selected intervals. The device is also configured via this connection. The device is equipped with an integrated SIM chip.

4.8 Device labelling

The specifications in this user manual apply exclusively to the myDatalogMUC device type. The type plate is located on the right side of the device and contains the following specifications:

- Type designation
- Serial number
- MAC address of the LAN interface
- Voltage supply specifications
- Environmental conditions during operation
- Country list profile of the SIM chip
- Protection class
- Hardware revision
- Logo for the EU WEEE Directive
- CE marking

It is important that the correct type designation and serial number is specified for all queries and spare part orders. Only then can we process requests promptly and properly.
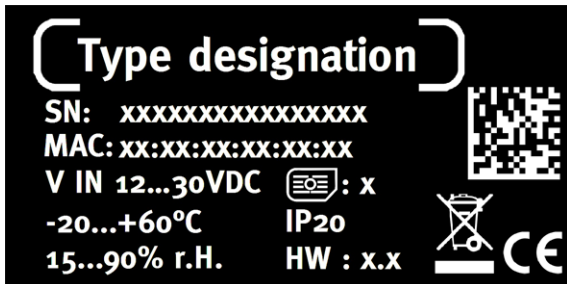
Type plate myDatalogMUC

Note: This symbol indicates the country list profile (see www.microtronics.com/footprint) of the SIM chip installed in the device.

Note: These operating instructions are part of the device and must be available to the user at all times. The safety instructions included therein must be observed.

DANGER:
It is strictly prohibited to disable the safety equipment or modify its mode of operation.

4.9 Installation of spare and wear parts

Be advised that spare and accessory parts that have not been supplied by the manufacturer have also not been inspected or approved by the manufacturer. The installation and/or use of such products can possibly have a negative impact on the specified constructional properties of the device. The manufacturer shall not be liable for any damages that arise from the use of non-original parts and non-original accessory parts.

4.10 Storage of the product

To safeguard the myDatalogMUC , ensure that all of the relevant data was transmitted to the myDatanet server. If necessary initiate the ALOHA transmission mode using the button (see "ALOHA transmission mode" on page 44) and then check again that all of the relevant data has been transmitted. Then disconnect the device from the supply voltage. If possible, switch off the supply voltage before disconnecting the cables from the V_{IN} and GND clamps (see "Connecting the sensors, actuators and power supply" on page 56). The remaining cables and the antenna can then be removed. Store the myDatalogMUC in its original packaging.

The configuration and most recently determined data are retained. However, the system clock will not continue to work without an energy supply. This means that following recommissioning, the time must be synchronised to ensure that the measurement and log data timestamps are correct. This is completed automatically during the first connection to the myDatanet server.

4.11 Obligation of the operator

⚠️ **WARNING:**
In the EEA (European Economic Area), the national implementation of the framework directive (89/391/EEC) as well as the associated specific directives and from these in particular, the directive (2009/104/EC) about the minimum safety and health requirements for use of work equipment by workers at work, each in their respective version are to be complied with.

The operator must obtain the local operating licence and the associated documents.

In addition, the operator must comply with the local legal requirements for

- the safety of the personnel (accident prevention measures)
- the safety of the equipment (protective equipment and maintenance)
- the product disposal (waste disposal law)
- the material disposal (waste disposal law)
- the cleaning (cleaning agents and disposal)
- and the environmental protection amendments.

Before commissioning, the operator must ensure that the installation and commissioning – provided these were performed by the operator himself – are in compliance with the local regulations.

4.12 Personnel requirements

Installation, commissioning and maintenance may only be completed by personnel who meet the following conditions:

- Qualified specialist personnel with the relevant training
- Authorised by the facility operator

Note: Qualified personnel

In the context of these instructions and the warnings on the product itself, individuals responsible for the setup, installation, commissioning and operation of the product must have gained relevant qualifications relating to their activities, including, for example:

- *Training, instruction and authorisation to activate/deactivate, ground and label electric circuits and devices/systems in accordance with the standards of safety engineering.*
- *Training or instruction on the maintenance and use of suitable safety equipment in accordance with the standards of safety engineering.*
- *First aid training*

Chapter 5 Functional principle

In the graphic below, all of the components that are part of the myDatanet are illustrated in grey. All of the other components must be provided/created by the customer.
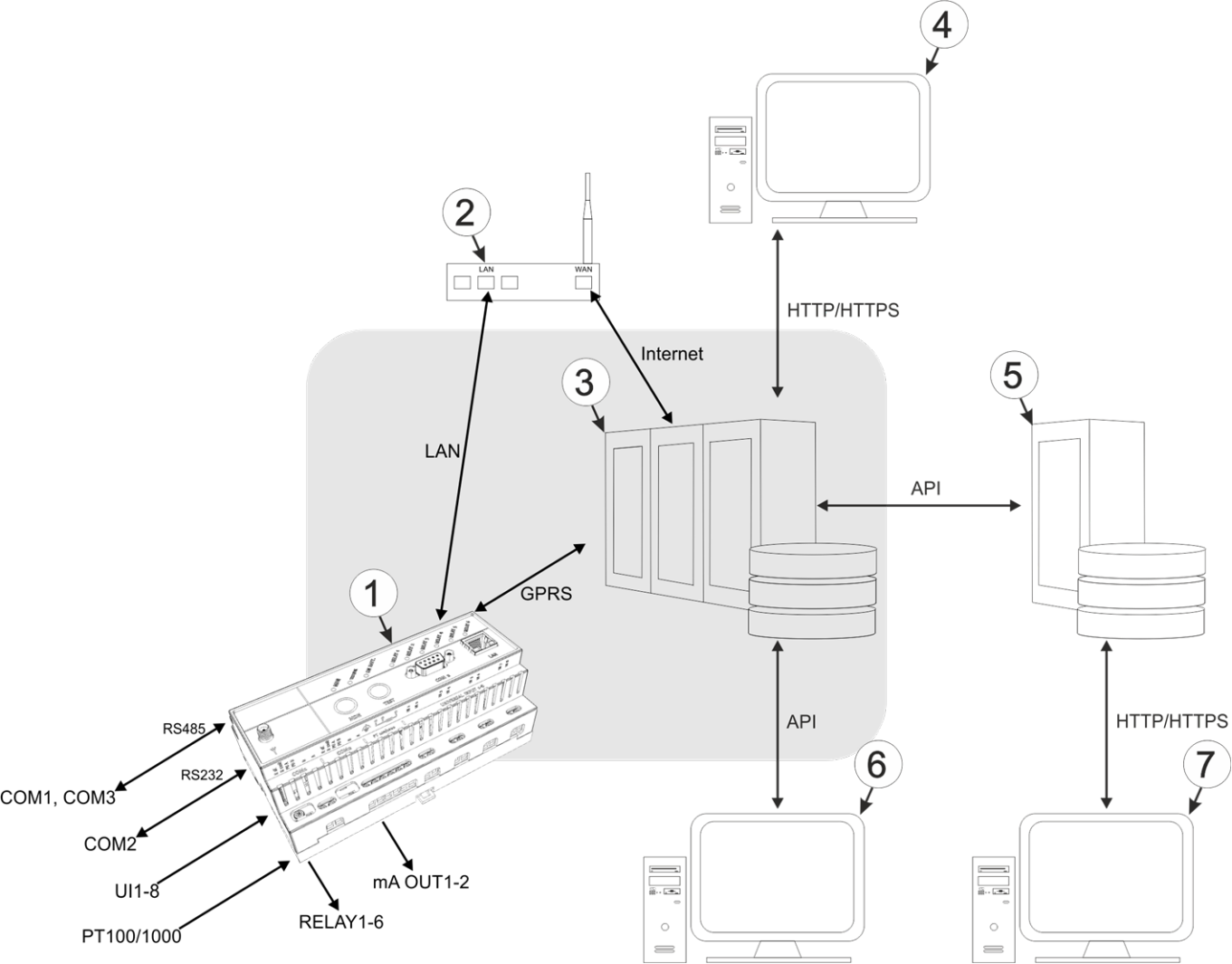
Functional principle

1	myDatalogMUC with integrated managed service SIM chip (including data transmission)
2	Router to connect the myDatalogMUC to the Internet via the LAN interface.
3	myDatanet server to which the data is transferred
4	Client that accesses the interface of the myDatanet server via the web browser
5	Customer-specific server that provides clients with their own interface. The customer-specific server obtains the data via the API of the myDatanet server (see "API" on page 207).
6	Client, on which a PC program is running, that obtains its data via the API of the myDatanet server (see "API" on page 207)
7	Client that accesses the interface of the customer-specific server via the web browser

Functions and components provided by myDatanet :

- myDatalogMUC

  The myDatalogMUC is a stationary device for connecting sensors (UI1-8, PT100/1000, COM1-3) and actuators (RELAY1-6, mA OUT1-2, COM1-3) to the myDatanet server (GPRS).

- Managed Service

  Managed Service is the basis for operating your devices and provides you with a wide range of services. Managed Service includes updates for device firmware, mobile data transmission on a global scale and free support - providing you with one contact person for the entire solution.

- myDatanet server

  Database for saving the measurement data and configurations. Data is either accessed via the API of the server (see "API" on page 207) or web interface of the server.

Functions and components provided by the customer

- Sensor and actuators

  Sensor and actuators that include interfaces that are compatible with the specifications listed in the chapter "Technical data" (see "Technical data" on page 15).

- Router to connect the myDatalogMUC to the Internet (optional)

  This means that in addition to the GPRS connection, the LAN connection is available as a second communication channel for connecting to the myDatanet server. The LAN connection can then be used as a preferred communication channel as well as a backup channel. Additional information is provided in "Use of the communication channels" on page 41.

- Customer-specific server with web interface for the clients (optional)

  It is therefore possible to create an individual web interface for the clients. Using this method, the data is read out of the myDatanet server via the API (see "API" on page 207) by the customer-specific server.

5.1 Internal processing of the measurement values
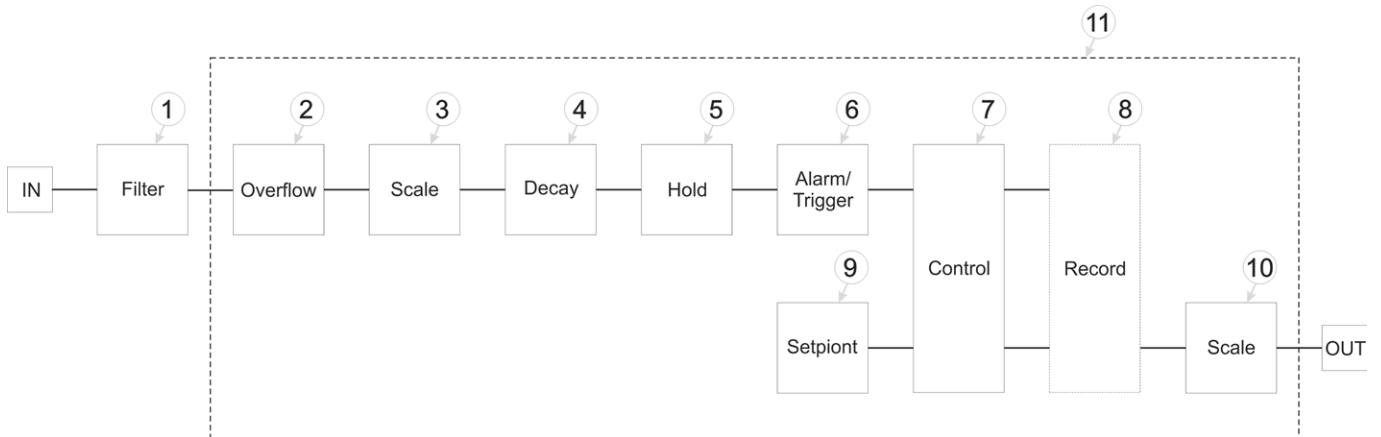
Diagram of the internal processing of the measurement values

1	Filter to compensate for brief signal fluctuations (see "Filter module" on page 32). The filter module is operated permanently.	**7**	Determination of the setpoints, script processing (see "Control module" on page 36)
2	Monitoring of the measurement range limits (see "Overflow module" on page 33)	**8**	As the record interval and measurement cycle can be selected individually, the measurement values and setpoints are not recorded at the time of every measurement (see "Record module" on page 38).
3	Rescaling from the raw value to the measurement value (see "Scale module (inputs)" on page 34)	**9**	Specification of the setpoints for the outputs (siehe "Setpoint module" on page 39)
4	Decay module to summarise several measurement values (see "Decay module" on page 35)	**10**	Rescaling from the setpoint to the physical size for the output (siehe "Scale module (outputs)" on page 40)
5	Module to maintain the last valid value in the event of invalid measurement values (see "Hold module" on page 36)	**11**	This module chain is started at the time of every measurement and is completed once. For universal inputs that are operated in Digital or Digital LP mode, this module chain (with exception of the control module) is also executed at second intervals to be able to react to level changes as quickly as possible.
6	Monitoring of the alarm limits and trigger levels (see "Alarm/trigger module" on page 36)		

Measurement channels -> Basis		Mode	Digital
Measurement channels ->Config.	⟷	Filter time	800ms

Input signal	Input signal at the universal input
Input signal after filter	Input signal taking the "filter time" into consideration
Logic level	The input signal at the universal input is analysed once per second.
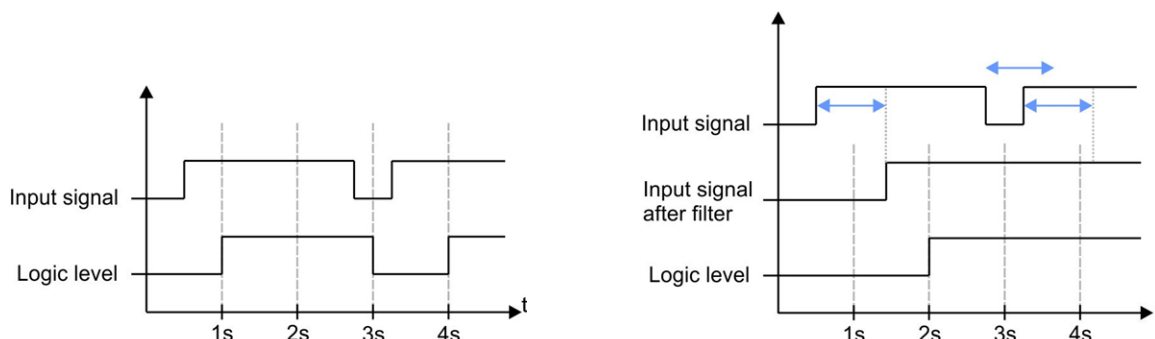
Filter module to compensate for brief signal fluctuations disabled ("filter time" = 0)

Filter module to compensate for brief signal fluctuations active ("filter time" not equal to 0)

Explanation: To react to a level change at the universal inputs as quickly as possible, the module chain detailed in "Internal processing of the measurement values" on page 31 (with exception of the control module) for the relevant universal input is operated once per second in Digital or Digital LP operating mode. This ensures that alarms and triggers are detected asynchronously to the measurement cycle. The signal must be present for at least 1 second to definitely detect a pulse safely. Additionally, any brief faults can be eliminated with the help of the "filter time".

5.1.1 Filter module

The filter module is designed to compensate for brief fluctuations to the input signal. This module is only available for the 8 universal inputs.

The following table specifies the relevant parameters for the module:

Configuration section	Mode	Parameter	Explanation
Measurement channels -> Config.	Digital	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).
	Cnt.Day		
	Cnt.Intrvl.		
	Freq		
	PWM		
	4-20 mA	Filter time	Time in [ms] during which the analogue signal is averaged for signal smoothing. Used to suppress signal noise (also see "Example to explain the filter time in conjunction with the ext. warmup time" on page 131).
	0-20 mA		
	0-2V		
	0-10V		

5.1.2 Overflow module

This module monitors the measurement range limits of the raw value. If a universal input was, for example, switched to "4-20 mA" mode, a raw value of 2 mA will violate the measurement range. The overflow module is only available for the 8 universal inputs in channel modes "Freq", "PWM", "4-20 mA", "0-20 mA", "0-2V" and "0-10V" as well as for the interface channels (Com1 and Com3 only) with activated scaling.

The following table specifies the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Measurement channels -> Config.	Digital	---	---
	Cnt.Day		
	Cnt.Intrvl.		
	Freq	Overflow	Procedure in the event of measurement range violations
	PWM		
	4-20 mA		
	0-20 mA		
	0-2V		
	0-10V		
Interface channels 1-64 -> Scaling	Com1	Overflow	Procedure in the event of measurement range violations
	Com3		

5.1.3 Scale module (inputs)

This module rescales the raw value (e.g. mA) to the required measurement value (e.g. mm). The scale module is only available for the 8 universal inputs and for the interface channels (Com1 and Com3 only).

The following table specifies the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Measurement channels -> Basis	Digital	Invert	Inverts the input signal
	Cnt.Day	Impulse	Counted measurand of a pulse in the measurement unit
	Cnt.Intrvl.		
	Freq	Factor	Factor by which the input signal is multiplied
	PWM 4-20 mA 0-20 mA 0-2 V 0-10 V	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
Interface channels 1-64 -> Scaling	Com1 Com3	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		0% modbus	Start of the measurement range in the unit of the modbus slave
		100% modbus	End of the measurement range in the unit of the modbus slave

5.1.4 Decay module

The decay module is designed to summarise several measurement values. The average value over a required time frame or the minimum value within a required time frame can be determined, for example. The decay module is only available for the 8 universal inputs, with the exception of "Cnt.Day" channel mode.

The following table specifies the relevant parameters for the module:

Configuration section	Mode	Parameter	Explanation
Measurement channels -> Config.	Digital	Decay	Temporal function in the measurement cycle
		Time	Time x, that is used for decay modes "up", "down" and "up & down"
	Cnt.Day	---	---
	Cnt.Intervl. Freq PWM 4-20 mA 0-20 mA 0-2 V 0-10 V	Decay	Temporal function in the measurement cycle
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).

5.1.5 Hold module

With the help of the hold module, it can be determined how to proceed in the event of invalid measurement values. It ensures the last valid measurement value is retained until a new valid measurement value is available or passes the error on after a certain number of invalid measurements. The hold module is only available for the universal inputs in channel modes "Freq", "PWM", "4-20 mA", "0-20 mA", "0-2V" and "0-10V" as well as the interface channels.

The following table specifies the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Interfaces -> Basis	Com1	Hold	Hold the last valid measurement value for x measurement cycles
	Com2		
	Com3		
Measurement channels -> Config.	Digital	---	---
	Cnt.Day		
	Cnt.Intrvl.		
	Freq	Hold	Hold the last valid measurement value for x measurement cycles
	PWM		
	4-20 mA		
	0-20 mA		
	0-2V		
	0-10V		

5.1.6 Alarm/trigger module

The alarm limits and trigger levels are monitored by this module. An entry is created in the alarm list if required. The global triggers are also set (see "Alarm flags" or "Trigger flags" in chapter "Constants" on page 163). The alarm/trigger module is only available for inputs (universal inputs, ext. temperature sensor, interface channels, internal channels).

All of the parameters of the "Alarms" and "Trigger" tabs of the "Measurement channels" (see "Measurement channels" on page 92), "Interface channels 1-32" (see "Alarms" on page 116 and "Trigger" on page 117) and "Internal channels" (see "Internal channels" on page 131) configuration sections are relevant to this module. Additionally, the parameters in the following tables are also relevant:

Configuration section	Parameter		Explanation
Alarm settings	On alarm On warning On fault alarm On fault warning	A	The alarm is recorded in the alarm list.

5.1.7 Control module

The control module completes the script processing and determines the setpoints for the outputs. When determining the setpoints for the outputs, it must be taken into account that a value calculated by means of a script overwrites the value determined via the input screen on the myDatanet server. The values of a

measurement channel can also still be changed by the script before it is recorded by the record module. If the "Pawn" script type was selected, the alarm status of a channel can be changed via the "Mdn_SetAlarm()" function (see Mdn_SetAlarm()). An entry created by the alarm/trigger module in the alarm list cannot be changed during this process, only an additional entry for the relevant channel can be created.

The control module execute the complete script if the "IL" (Instruction list) script type was selected. The "Mdn_CtrlFinish()" function (see Mdn_CtrlFinish()) is called up by the control module if the "Pawn" script type is selected.

In addition to the parameters in the following table, all of the parameters of the "Control" configuration section (see "Control" on page 79) are also relevant:

Configuration section	Modus/interface	Parameter	Explanation
Output channels	Off	---	---
	0-20 mA		
	4-20 mA		
	Ext warmup time	Ext warmup time	The output channel is switched on "Ext warmup time" seconds prior to the measurement. If the value is "0", the output channel is not switched on.
	Digital	---	----

5.1.8 Record module

The record module records the measurement values. As the record interval and measurement cycle can be selected individually, a record is not saved at the time of every measurement. Depending on the global triggers set by the alarm/trigger module (see "Alarm flags" or "Trigger flags" in chapter "Constants" on page 163), the record interval is modified, if necessary, the transmission is initiated or a new measurement is triggered.

The following tables specifie the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Measurement channels -> Trigger	Digital	QU	Fastrecording (record interval = record interval / factor)
		SL	Slow recording (record interval = record interval * factor)
		MS	Start measurement cycle immediately
		XM	Initiate transmission
		ON	Activate online mode
	Cnt.Day Cnt.Intervl. Freq PWM 4-20 mA 0-20 mA 0-2 V 0-10 V ext. temp. sensor	QU	Fastrecording (record interval = record interval / factor)
		SL	Slow recording (record interval = record interval * factor)
		RO	Switch on recording
		RF	Switch off recording
		XM	Initiate transmission
		ON	Activate online mode
Interface channels 1-64 -> Trigger	Com1 Com2 Com3 Script	QU	Fastrecording (record interval = record interval / factor)
		SL	Slow recording (record interval = record interval * factor)
		RO	Switch on recording
		RF	Switch off recording
		XM	Initiate transmission
		ON	Activate online mode
Internal channels -> Trigger	---	QU	Fastrecording (record interval = record interval / factor)
		SL	Slow recording (record interval = record interval * factor)
		RO	Switch on recording
		RF	Switch off recording
		XM	Initiate transmission
		ON	Activate online mode

Configuration section	Parameter		Explanation
Alarm settings	On alarm On warning On fault alarm On fault warning	Ü	An immediate transmission is initiated.

Configuration section	Parameter	Explanation
Basic setting	Record interval	Time between measurement data recordings
	Measure quick divisor	Record interval = record interval / factor (from triggering)
	Measure slow factor	Record interval = record interval * factor (from triggering)

5.1.9 Setpoint module

This module uses the setpoints for the outputs entered via the configuration interface of the myDatanet-Servers.

The following table specifies the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Interface output channels 1-64 -> Basis	off	---	---
	Com1	Setpoint	Output value in the measurement unit
	Com2		
	Com3		
	Script		
Output channels	off	---	--
	0-20mA[1]	Setpoint	Output value in the measurement unit
	4-20mA[1]		
	Ext. warmup time[2][3]	---	---
	Digital[2]	Setpoint	Setpoint (on/off) that should be issued

[1] This mode is only available for the analogue outputs.

[2] This mode is only available for the relays.

[3] In "Ext. warmup time" mode, the output is actuated by the device itself, for example to switch on the supply of a sensor according to the measurement (see "Output channels" on page 128).

5.1.10 Scale module (outputs)

This module rescales the setpoint (e.g. mm) into the desired physical size (e.g. mA) for the output.

The following table specifies the relevant parameters for the module:

Configuration section	Modus/interface	Parameter	Explanation
Interface output channels 1-64 -> Scale	Com1 Com3	0% Modbus	Start of the output range in the unit of the modbus slave
		100% Modbus	End of the output range in the unit of the modbus slave
		0%	Start of the output range in the measurement unit
		100%	End of the output range in the measurement unit
Output channels	off	---	---
	0-20mA[1]	0%	Start of the output range in the measurement unit
		100%	End of the output range in the measurement unit
	4-20mA[1]	0%	Start of the output range in the measurement unit
		100%	End of the output range in the measurement unit
	Ext. warmup time[2][3]	---	---
	Digital[2]	Invert	Inverts the level issued on the device

[1] This mode is only available for the analogue outputs.

[2] This mode is only available for the relays.

[3] In "Ext. warmup time" mode, the output is actuated by the device itself, for example to switch on the supply of a sensor according to the measurement (see "Output channels" on page 128).

5.2 Functionality of the internal data memory

Structure	Circular buffer
Total size	4.470
Number of sectors	5
Sector size	894 measurement cycles

The internal data memory of the myDatalogMUC is designed as a circular buffer with 5 sectors. If the maximum number of data records (4.470) is achieved, the sector with the oldest data is deleted fully before new data can be saved in this sector again. This means that the internal data memory at the very least

contains the measurement values of the last 3576 cycles, however at most the measurement values of the last 4.470 cycles.

For this reason, it is recommended to coordinate the transmission cycle and record interval in such a way that a maximum of 3576 measurement cycles have to be recorded between two transmissions. Please note, that if the measurement cycle is shorter than the record interval, the record interval still has to be used for the calculation. In this case, the reason for this is that although the measurement is completed in the measurement cycle, the determined data is saved in the data memory in the record interval. If it can be expected that individual transmissions fail due to poor network coverage or the measure quick divisor is activated via the trigger, this must also be taken into consideration when calculating the measurement cycles that are to be saved.

Note:

Additional explanation regarding the functionality of the circular buffer

Data memory after the first measurement cycle:
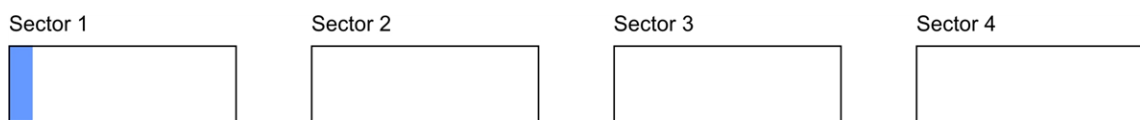
Data memory after 4.470 measurement cycles:
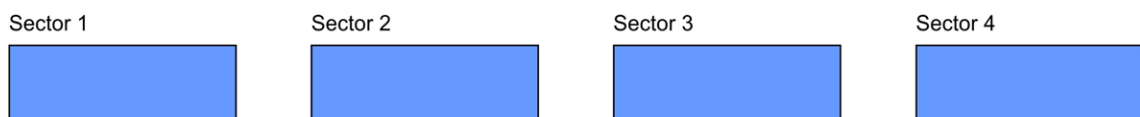
Data memory after 4.470 +1 measurement cycles:
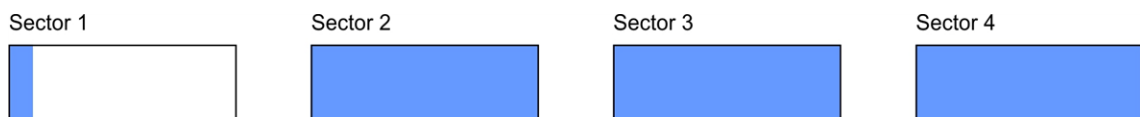
5.3 Use of the communication channels

Two channels (LAN and GPRS) are available to the myDatalogMUC for communicating with the myDatanet server. Their use can be configured via the "Communication channel" parameter located in the "Basic settings" configuration section (see "Basic setting" on page 134). There is an option to just use the GPRS connection or to use both connections, whereby one of the two connections has to be used as the preferred connection and the other as the backup connection. The LAN interface is configured via the "LAN settings" configuration section (see "LAN settings" on page 137).

If both of the communication channels are to be used, an attempt is always made to establish the connection via the preferred interface first. The backup connection is only used if no connection is possible. When establishing the next connection, the first attempt to establish the connection is once again made via the preferred interface. It must be noted that the determination of the instrument position can only be updated if the connection to the myDatanet server is established via GPRS.

5.3.1 Behaviour in connection modes when using both communication channels

5.3.1.1 Interval

In this mode, an attempt is made to establish a connection via the preferred communication interface according to the set transmission cycle. If no connection can be established, an attempt is made to establish a connection via the backup interface. If this is also not possible, no further attempts are made to establish the connection until the next transmission time.

5.3.1.2 Interval & wakeup

The behaviour basically corresponds to that of "Interval" mode. If the GPRS interface is to be used as the preferred communication channel, the myDatalogMUC can also be placed in ALOHA transmission mode via the myDatanet server (see "ALOHA transmission mode" on page 44).

5.3.1.3 online

The behaviour of the backup connection, if the preferred connection fails, can be selected via the checkbox "Online at backup connection" located in the "Basic settings" configuration section (see "Basic setting" on page 134).

- Checkbox not selected (standard)

retry	2min
interval	Transmission cycle
t_{backup}	5sec.

Note: The transmission cycle cannot be changed if "online" connection mode is activated. To change the transmission cycle, the mode must first be changed to "Interval" or "Interval & wakeup".
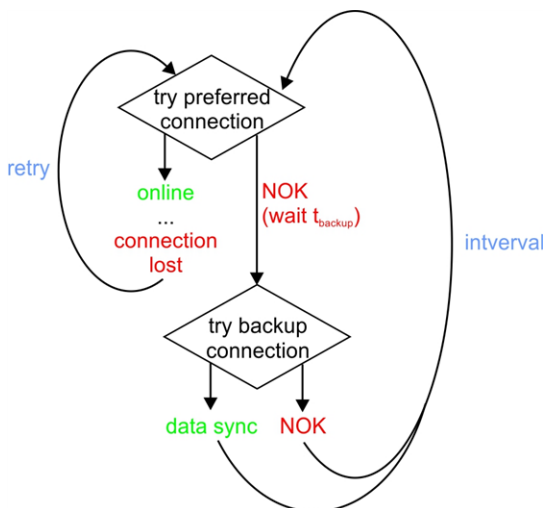
Sequence of the connection process

An online connection is only established via the preferred communication channel. If it is not possible to establish the connection, an attempt is made to synchronise the data via the backup channel after 5sec. . Once the data has been transferred via the backup channel, the myDatalogMUC remains offline until the transmission cycle elapses. Following expiry of the transmission cycle, an initial attempt is made to establish the online connection via the preferred communication channel.

If a connection established via the preferred communication channel is lost, a new attempt is made to establish the connection via the preferred channel after 2min .

- Checkbox selected (always online)

retry	2min
retry$_{backup}$	30sec.
t$_{backup}$	5sec.
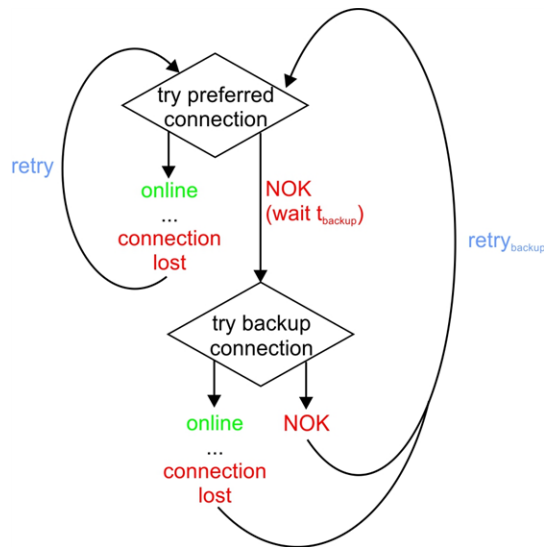
Sequence of the connection process

An initial attempt is made to establish an online connection via the preferred communication channel. If it is not possible to establish the connection, an attempt is made to establish an online connection via the backup channel after 5sec. . If the connection via the backup channel is lost or is not established in the first place, another attempt is made to establish the online connection via the preferred communication channel after 30sec. .

If a connection established via the preferred communication channel is lost, a new attempt is made to establish the connection via the preferred channel 2min .

5.3.1.4 ALOHA

If ALOHA transmission mode (see "ALOHA transmission mode" on page 44) was initiated, the myDatalogMUC tries to establish a connection via the preferred communication interface. If this is not possible, an attempt is made to establish a connection via the backup interface.

> *Note: ALOHA transmission mode is not actively interrupted by myDatalogMUC , this means that any change to the connection type during ALOHA transmission mode only comes into effect following completion of the ALOHA transmission.*

5.4 Procedure in case of connection aborts

The procedure described in the following only applies to the preferred connection. A retry is not attempted if a connection established via a backup interface is terminated.

If the connection is terminated, another attempt to establish a connection is made after 2min for all connections, except for ones that were established by initiating ALOHA transmission mode. The connection is attempted up to 2 times, as long as the previous attempt to establish a connection was successful. This means that if the connection fails and it is not possible to re-establish the connection during the first retry after 2min , a further retry will not be completed after 2min .

This process also applies if the connection is terminated during online mode.

5.5 ALOHA transmission mode

ALOHA transmission mode is a special connection mode whereby the myDatalogMUC establishes a connection to the myDatanet server for a period of time configured via the "Basic settings" configuration section (see "Aloha/wakeup duration" in chapter "Basic setting" on page 134).

There are two options for initiating the ALOHA transmission mode:

- Directly on the device using a button (see "Button to initiate ALOHA transmission mode" on page 75). The GSM network is re-selected during this process. This means that a list of all of the GSM networks in the receiving range of the device is created from which the strongest GSM network is then selected that is not on the "Ban list" (see "Ban list" on page 45) and that is part of one of the service providers supported by the integrated SIM chip (see www.microtronics.com/footprint). The internal fault counter is pre-initialised in such a way that if it was not possible to log into the selected GSM network, it is immediately added to the "Ban list". In this case, the next GSM network from the previously created list is used immediately. If the connection with all of the available GSM networks fails, the ban list is deleted and the connection attempt is aborted. If ALOHA transmission mode is then initiated again, the ban list is empty and all of the GSM networks in the receiving range, that are part of one of the service providers supported by the integrated SIM chip, are taken into consideration when selecting the network. The network selection also takes into consideration whether GPRS is available on the selected GSM network.

- Via the myDatanet server if the "Interval & wakeup" connection type has been selected (see "Basic setting" on page 134).

A speech bubble in the measurement instrument list (see "myDatanet Server Manual " 805002) with the title "Aloha" indicates that a device is in ALOHA transmission mode.

Clicking on the speech bubble with the "Aloha" title opens the ALOHA data window (see "myDatanet Server Manual " 805002), It contains the internal "Voltage" and "GSM level" measurement values, as well as the measurement values of the 8 universal inputs. The measurement values for the last thirty minutes are always shown in the ALOHA data window, which means that data from a previous ALOHA transmission may be included. The ALOHA data is generated every three seconds independently of the normal measurement values and is thus not saved with the standard measurement data.

If a measurement dataset is recorded during ALOHA transmission mode, it is immediately transferred to the myDatanet server and saved in the standard measurement data.

If a display element that uses one of the device's measurement values is shown on the server interface, this data is also transferred to the myDatanet server each time a measurement value is generated. If the measurement is completed more frequently than the recording ("measurement cycle" is not equal to zero and/or "burst interval" is not equal to zero) (see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136), this increases the data transmission. However, this data is not permanently stored on the server or on the device, but is instead used to update the display elements on the server (see "myDatanet Server Manual " 805002).

5.6 Automatic selection of the GSM network

The GSM network to which the device should register must be selected, as the myDatalogMUC is equipped with a SIM chip that provides a mobile connection via a variety of international service providers (see www.microtronics.com/footprint). This is completed automatically by the device. During this process it is taken into consideration if a connection to a certain GSM network is possible although the use of this GSM network repeatedly causes transmission errors. Such a GSM network is added to a ban list and is ignored during the next login attempt.

The following conditions ensure that the myDatalogMUC logs itself into a GSM network:

- PowerOn
- Expiry of the transmission cycle if the "Interval" connection type was selected (see "Basic setting" on page 134)
- Activation of the "Interval & Wakeup" or "Online" connection type (see "Basic setting" on page 134)
- Initiation of ALOHA transmission mode (see "ALOHA transmission mode" on page 44)

1. The myDatalogMUC checks whether any errors occurred during the last login attempt.

    No errors:

    The myDatalogMUC logs into the same GSM network that was also used during the last attempt.

    Error:

    Continue with step 2

    Note: A transmission error is not the same as an error that occurs when logging into the GSM network. It is possible for the login to the GSM network to work correctly but that a transmission error still occurs.

The following steps are only completed if an error was detected during step 1:

2. The device searches for GSM networks within its receiving range and creates a list of available GSM networks.

3. A search of the list of available GSM networks is completed and the strongest GSM network is selected that is not on the "ban list" and is assigned to one of the service providers supported by the integrated SIM chip (see www.microtronics.com/footprint).

    Note: An explanation detailing the conditions when a GSM network is added to the "ban list" is provided in "Ban list" on page 45.

5.6.1 Ban list

The ban list provides space for 5 entries and is not saved in a volatile memory. This means that it is also still available following a PowerOn.

If there is a GPRS error (see error codes -975 to -993 in chapter "Modem error" on page 222), then an internal counter is increased by 2 . If a transmission was completed without errors, i.e. all of the data was transferred, this internal counter is reduced by 1 . The current GSM network is added to the ban list if the internal counter reaches the value 6 . The relevant GSM network is automatically added to the ban list, if the "MODEM CMME ERROR" error occurs (see "Modem error" on page 222) with parameter 32 (NETWORK NOT ALLOWED) or 529 (MODEM CMME ERROR). The ban list is deleted if no GSM network can be selected from the list of available GSM networks in the receiving range taking the ban list into consideration. Another search for GSM networks in the receiving range is completed once the ban list is deleted and the

network selection is completed with the ban list that is now empty (see step 3 in chapter"Automatic selection of the GSM network" on page 45).

The quality of the data transfer is monitored in addition to the establishment of the GPRS connection. If the data transmission fails 10 times in succession despite an existing GPRS connection, the currently selected GSM network is also added to the ban list. The same applies in this case, which means that if, when taking the ban list into consideration, there is no further GSM network available, the ban list is deleted and selection of the strongest GSM network is restarted.

5.7 Script eradication

The PAWN script must include the following two functions:
main();
This function is executed during a PowerOn and when the PAWN script is exchanged. It should include all initialisations that only have to executed once during program start-up.

Mdn_CtrlFinish();
This function is called up at the time of every measurement once all of the measurement values have been generated. It should comprise all of the calculations and functions that should be performed cyclically.

> *Note: If a pawn script is used to write directly to an output, the setpoint entered via the input screen of the myDatanet server is overwritten with the calculated value.*

If an error occurs during this process, the script execution is stopped and deactivated. The relevant error code is also entered in the device log (see "Pawn script error codes" on page 180).

5.7.1 Script parsing

If script parsing was activated for one of the COM interfaces, the relevant interfaces can be accessed by means of the "Mdn_SerialEvent()", "Mdn_SerialRx()", "Mdn_SerialTx()" and "Mdn_SerialFinish()" PAWN script functions. The baud rate, stop bits, parity, data bits, frame timeout and activation of a retry are still configured via the "Interface" configuration section (see "Interfaces" on page 80). The following two graphics indicate at what times the "Mdn_SerialEvent()" (see Mdn_SerialEvent()) and "Mdn_SerialRx()" (see Mdn_SerialRx()) callback functions are called up and what parameters are transferred during this process.

Basic setting		Measurement cycle	60 sec.
Interfaces ->Config.		Warmup time	30 sec.

Sequence of the serial communication

Sequence of the serial communication in detail

1	Function: Mdn_SerialEvent() Event: MDN_SERIAL_EVENT_INIT	4	Only when "Retry" is active Function: Mdn_SerialEvent() Event: MDN_SERIAL_EVENT_TIMEOUT or Mdn_SerialRx()
2	Function: Mdn_SerialEvent() Event: MDN_SERIAL_EVENT_MEASURE	5	Function: Mdn_CtrlFinish()
3	Function: Mdn_SerialEvent() Event: MDN_SERIAL_EVENT_TIMEOUT Event: MDN_SERIAL_EVENT_MEASURE or Mdn_SerialRx()		

Explanation of 1

A delay can be set between the PowerOn and the dispatch of the init sequence via the warmup time (call up of the "Mdn_SerialEvent()" callback function by the system with the "MDN_SERIAL_EVENT_ INIT" event). However, the warmup time can only be entered for Com2 via the configuration interface. The relevant parameter for configuring the interface is located in the "Config" tab of the input screen (see "Config" on page 90). The warmup time for interfaces Com1 and Com3 is always 0 sec. The user should react to the "MDN_SERIAL_EVENT_INIT" event by using the "Mdn_SerialTx()" function to send the init sequence via the interface.

Explanation of 2

If it is time to initiate a measurement, the "Mdn_SerialEvent()" callback function is called up by the system with the "MDN_SERIAL_EVENT_MEASURE" event. The internal processing of the measurement values also starts at this time (see "Internal processing of the measurement values" on page 31). The user must react to the "MDN_SERIAL_EVENT_MEASURE" event by using the "Mdn_SerialTx()" function to send the measurement sequence via the interface.

> *Note: The time of the first measurement following the PowerOn is calculated and is not completed exactly according to the time after the PowerOn specified via the measurement cycle. If the measurement cycle is 1 minute long, the first time of the measurement is selected in such a way that it is completed at the full minute mark. This means that if the PowerOn is completed at 12:05:34, the first measurement is taken at 12:06:00, i.e. 26 sec. after the PowerOn.*

Explanation of 3

This point is achieved if data has been received or a timeout has occurred. The time for the timeout is entered via the "Basis" tab of the input screen for configuring the interface (see "Config" on page 90).

If a retry is activated, the "Mdn_SerialEvent()" callback function with the "MDN_SERIAL_EVENT_MEASURE" event is called up by the system in the event of a timeout. The user must react to this event by using the "Mdn_SerialTx()" function to resend the measurement sequence.

If a retry is not activated, the "Mdn_SerialEvent()" callback function with the "MDN_SERIAL_EVENT_TIMEOUT" event is called up by the system in the event of a timeout. In this case, the receipt of data must be terminated by the "Mdn_SerialFinish()" function.

However, if valid data has been received, the "Mdn_SerialRx()" callback function is called up by the system. It supplies the received data as an array. Following the evaluation of the data, the user must copy the required measurement values to the interface channel by means of the "Mdn_SetCh()" function (see Mdn_SetCh()).

> *Important note: During this process, only interface channels, that are connected to an interface for which script parsing was activated, should be used. Otherwise the data generated by the system will be overwritten.*

If all of the required data has been received, the receipt of data must be terminated by the "Mdn_SerialFinish()" function. Otherwise the internal processing of the measurement values cannot be continued. Following the call up of the "Mdn_SerialFinish()" function the internal processing of the measurement values is continued from the decay module onwards (see "Internal processing of the measurement values" on page 31). This means the levels for the alarms or triggers can still be edited via the "Alarm" and "Trigger" tabs in the input screen for configuring interface channels 1-64 (see "Alarms" on page 116 or "Trigger" on page 117).

Explanation of 4 (only when "Retry" is activated)

In the event of a timeout, the "Mdn_SerialEvent()" callback function is called up by the system with the "MDN_SERIAL_EVENT_TIMEOUT" event. In this case, the receipt of data must be terminated by the "Mdn_SerialFinish()" function. However, if valid data has been received, the "Mdn_SerialRx()" callback function is called up by the system. It supplies the received data as an array. Following the evaluation of the data, the user must copy the required measurement values to the interface channel by means of the "Mdn_SetCh()" function (see Mdn_SetCh()).

> ***Important note:*** *During this process, only interface channels, that are connected to an interface for which script parsing was activated, should be used. Otherwise the data generated by the system will be overwritten.*

If all of the required data has been received, the receipt of data must be terminated by the "Mdn_SerialFinish()" function. Otherwise the internal processing of the measurement values cannot be continued. Following the call up of the "Mdn_SerialFinish()" function the internal processing of the measurement values is continued from the decay module onwards (see "Internal processing of the measurement values" on page 31). This means the levels for the alarms or triggers can still be edited via the "Alarm" and "Trigger" tabs in the input screen for configuring interface channels 1-64 (see "Alarms" on page 116 or "Trigger" on page 117).

Explanation of 5

The "Mdn_CtrlFinish()" (see Mdn_CtrlFinish()) callback function is called up if the internal processing of the measurement values reaches the control module. It is then possible to send additional commands via the interface by means of the "Mdn_SerialTx()" function to, for example, control the actuators.

Chapter 6 Storage, delivery and transport

6.1 Inspection of incoming deliveries

Check the shipment immediately upon receipt to ensure it is complete and intact. Immediately report any discovered transport damages to the delivering carrier. Also notify Microtronics Engineering GmbHin writing about this without delay. Report any incompleteness of the delivery to the responsible representative or directly to the company headquarters of the manufacturer within two weeks (see "Contact information" on page 237).

> *Note: Any claims received thereafter will not be accepted.*

6.2 Scope of supply

The standard scope of supply of the myDatalogMUC (206.229) includes:

- myDatalogMUC
- Antenna 900 SMA-M angled (300026)
- myDatanet Tool Pen (206.646)
- Type plate sticker to attach to the control cabinet
- 3x stickers with the type designation "myDatalogMUC "

Additional accessories such as assembly sets, antennas, power supplies, sensors, etc. depending on the order specifications. Please check this against the delivery slip.

6.3 Storage

The following storage conditions must be observed:

myDatalogMUC	Storage temperature	-20...+60°C
	Humidity	15...90%rH

Store the measurement technology so that it is protected against corrosive and organic solvent vapours, radioactive emissions and strong electromagnetic radiation.

6.4 Transport

The myDatalogMUC should not be subjected to heavy shocks, bumps, impacts or vibrations. The original packaging must always be used for transport.

6.5 Return

Every return must be accompanied by a fully field-out return form. This return form is available in the service area of the myDatanet server. An RMA number is mandatory for any returns and can be obtained from the Support & Service Centre (see "Contact information" on page 237). The return shipment of the myDatalogMUC must occur in the original packaging and with freight and insurance paid to Microtronics Engineering GmbH (see "Contact information" on page 237). Insufficiently cleared return shipments will otherwise not be accepted!

Chapter 7 Installation

Important note: To prevent any damage to the device, the work described in this section of the instructions must only be performed by qualified personnel.

7.1 Dimensions

Dimensions: Width and height

Dimensions: Depth

7.2 Installing the myDatalogMUC

Important note:

- *Ensure installation is completed correctly.*
- *Comply with existing legal and/or operational directives.*
- *Improper handling can cause injuries and/or damage to the devices.*

The installation site must be selected according to specific criteria. The following conditions must be avoided in any case:

- Direct sunlight
- Direct weather exposure (rain, snow, etc.)
- Objects that radiate intense heat (maximum ambient temperature: -20...+60°C)
- Objects with a strong electromagnetic field (frequency converter or similar)
- Corrosive chemicals or gases
- Mechanical impacts
- Direct installation on paths or roads
- Vibrations
- Radioactive emissions

Note: Approx. 2-5 cm of space must be left above and below the device for the cable connections. The antenna connection is located on the front of the device. The space required depends on the antenna used. Further information about the installation dimensions can be found in the relevant sub-chapter.

7.2.1 Top-hat rail assembly

Top-hat rail assembly

1	myDatalogMUC	2	Top-hat rail

1. Place the myDatalogMUC on to the top edge of the top-hat rail. Turn slightly around the horizontal axis so that the myDatalogMUC clicks into the top-hat rail (see Figure "Top-hat rail assembly" on page 54).

7.2.2 Assembly in a control cabinet

The Antenna 900 SMA-M angled (300026) included in the scope of supply is not suitable for assembly within a control cabinet as the GSM signal is shielded by the metal of the cabinet. In this case, the manufacturer recommends using the Flat antenna Smart Disc SMA-M 2,5m (206.816) or Flat Antenna Smart Disc US SMA-M 2,5m (206.818) that are available as accessories.

Important note: The installation of the Flat antenna Smart Disc SMA-M 2,5m or Flat Antenna Smart Disc US SMA-M 2,5m is only possible up to a wall thickness of 3mm .
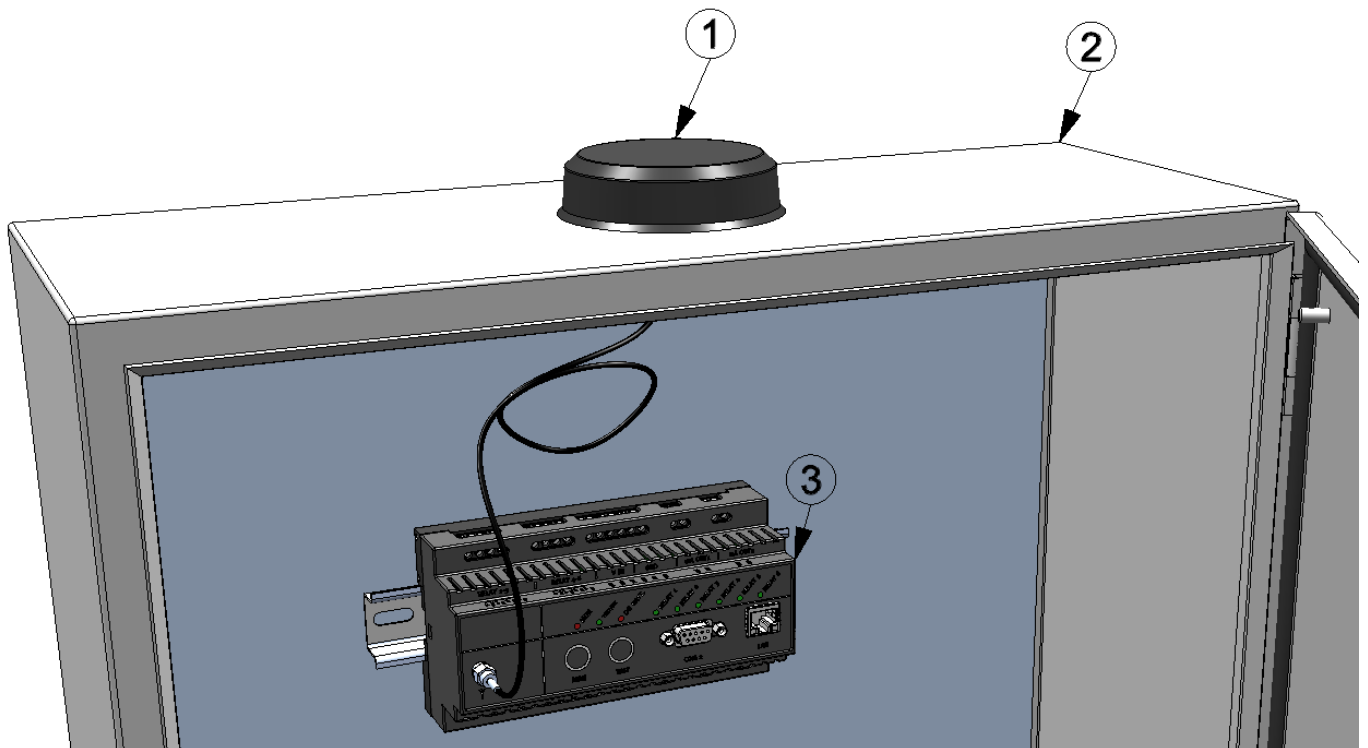
Control cabinet with mounted Flat antenna Smart Disc SMA-M 2,5m (206.816)

1	Flat antenna Smart Disc SMA-M 2,5m (206.816)	**3**	myDatalogMUC
2	Control cabinet		

7.3 Safety instructions for the cabling

Important note: To avoid any damage, always switch off the voltage supply to the device when performing electrical connections.

When connections are made to the myDatalogMUC , the following warnings and information must be observed, in addition to the warnings and information found in the individual chapters on the installation. Further safety information is included in "Safety instructions" on page 19.

7.3.1 Information on preventing electrostatic discharges (ESD)

Important note: Maintenance procedures that do not require the device to be connected to the power supply should only be performed once the device has been disconnected from the mains power supply to minimise hazards and ESD risks.

The sensitive electronic components inside the device can be damaged by static electricity, which can impair the device performance or even cause the device to fail. The manufacturer recommends the following steps to prevent any damage to the device caused by electrostatic discharges:

- Discharge any static electricity present on your body before handling the electronic components of the device (such as circuit boards and components attached thereto). To do this, you can touch a grounded metallic surface such as the housing frame of a device or a metal pipe.
- Avoid any unnecessary movements to prevent the build-up of static charges.
- Use antistatic containers or packaging to transport components that are sensitive to static.
- Wear an antistatic wristband that is grounded via a cable to discharge your body and keep it free of static electricity.
- Only touch components that are sensitive to electric charges in an antistatic working area. If possible, use antistatic mats and work pads.

7.4 Electrical installation

Important note: Only qualified personnel should undertake the installation described in this chapter of the operating instructions to avoid any damage to the device.

7.4.1 Connecting the sensors, actuators and power supply

Important note:

- *All cabling work must be performed in the de-energised state.*
- *Ensure installation is completed correctly.*
- *Comply with the existing legal and/or operational directives.*
- *Improper handling can cause injuries and/or damage to the devices.*
- *Run all data and power cables so that they do not pose a trip hazard and ensure that cables do not have any sharp bends.*
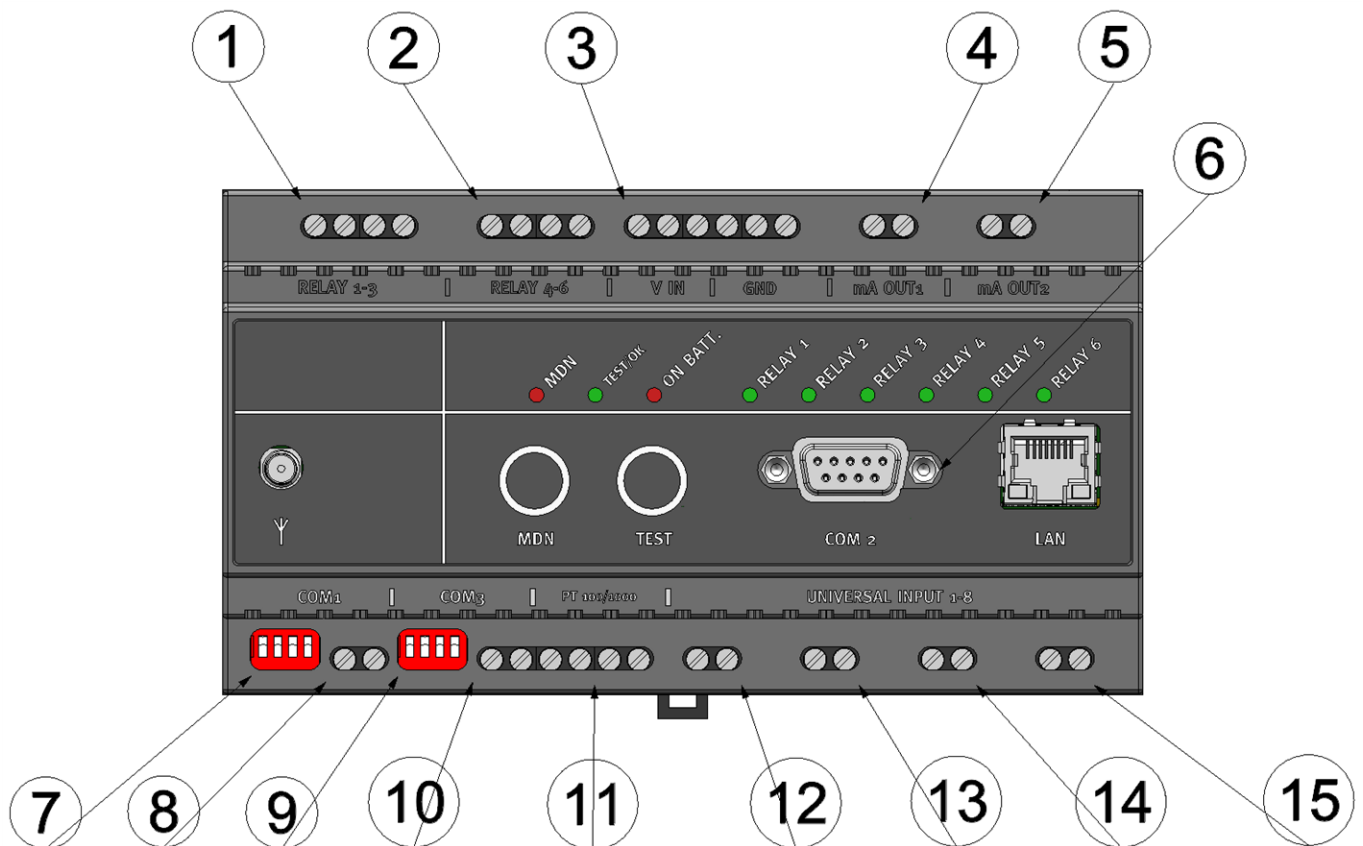
Connection of the sensors and power supply

1	Relay 1-3	**9**	Dip switch (dip switch 2) for activating/deactivating the load or clamp resistances for Com3 (RS485)
2	Relay 4-6	**10**	Com 3 (RS485)
3	Supply (V IN, GND)	**11**	Connector for the external temperature sensor (PT100/1000)
4	mA Out1	**12**	Universal input 1-2
5	mA Out2	**13**	Universal input 3-4
6	Com 2 (RS232)	**14**	Universal input 5-6
7	Dip switch (dip switch 1) for activating/deactivating the load or clamp resistances for Com1 (RS485)	**15**	Universal input 7-8
8	Com 1 (RS485)		

Relay 1-3

	CC1-3	Joint root for relay 1-3
	NO1	Operating contact of relay 1 (normally open)
	NO2	Operating contact of relay 2 (normally open)
	NO3	Operating contact of relay 3 (normally open)
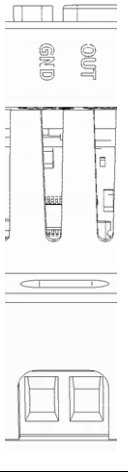
Relay 4-6

	CC4-6	Joint root for relay 4-6
	NO4	Operating contact of relay 4 (normally open)
	NO5	Operating contact of relay 5 (normally open)
	NO6	Operating contact of relay 6 (normally open)
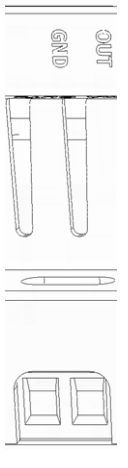
Supply (V IN, GND)

	GND	Mass
	V IN	Supply voltage: 12...30VDC (+/-10%), max. 3W
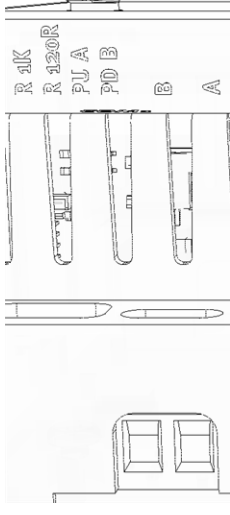
mA Out1

	GND	Mass
	OUT	0/4-20mA output signal

mA Out2

	GND	Mass
	OUT	0/4-20mA output signal

COM1

	R 1k	1k load resistance between RS485 A and B
	R 120 R	120Ω load resistance between RS485 A and B
	PU A	1k pull up to RS485 A
	PD B	1k pull down to RS485 B
	B	RS485 B
	A	RS485 A

COM3, PT 100/1000

	R 1k	1k load resistance between RS485 A and B
	R 120 R	120Ω load resistance between RS485 A and B
	PU A	1k pull up to RS485 A
	PD B	1k pull down to RS485 B
	B	RS485 B
	A	RS485 B
	Clamp for the cable shield of the external temperature sensor
	Clamp for the external temperature sensor (two or three wires)

UNIVERSAL INPUTS 1-8

	UI 1	Universal input 1
	UI 2	Universal input 2
	UI 3	Universal input 3
	UI 4	Universal input 4
	UI 5	Universal input 5
	UI 6	Universal input 6
	UI 7	Universal input 7
	UI 8	Universal input 8

1. Connect your sensors and actuators with the inputs and outputs. Ensure that no current is present! Ensure the supply cables for the myDatalogMUC are de-energised when being connected to the supply connectors.

2. Connect the antenna (see "Connection of the GSM antenna" on page 61).

3. Switch on the 12...30VDC supply voltage of the myDatalogMUC . The status LED should then start to flicker (see "Status LED" on page 76), indicating that a connection is being established.

The following step is not mandatory.

4. Check whether the connection to the myDatanet has worked correctly (see "Testing communication with the device" on page 72).

7.4.1.1 Connection examples
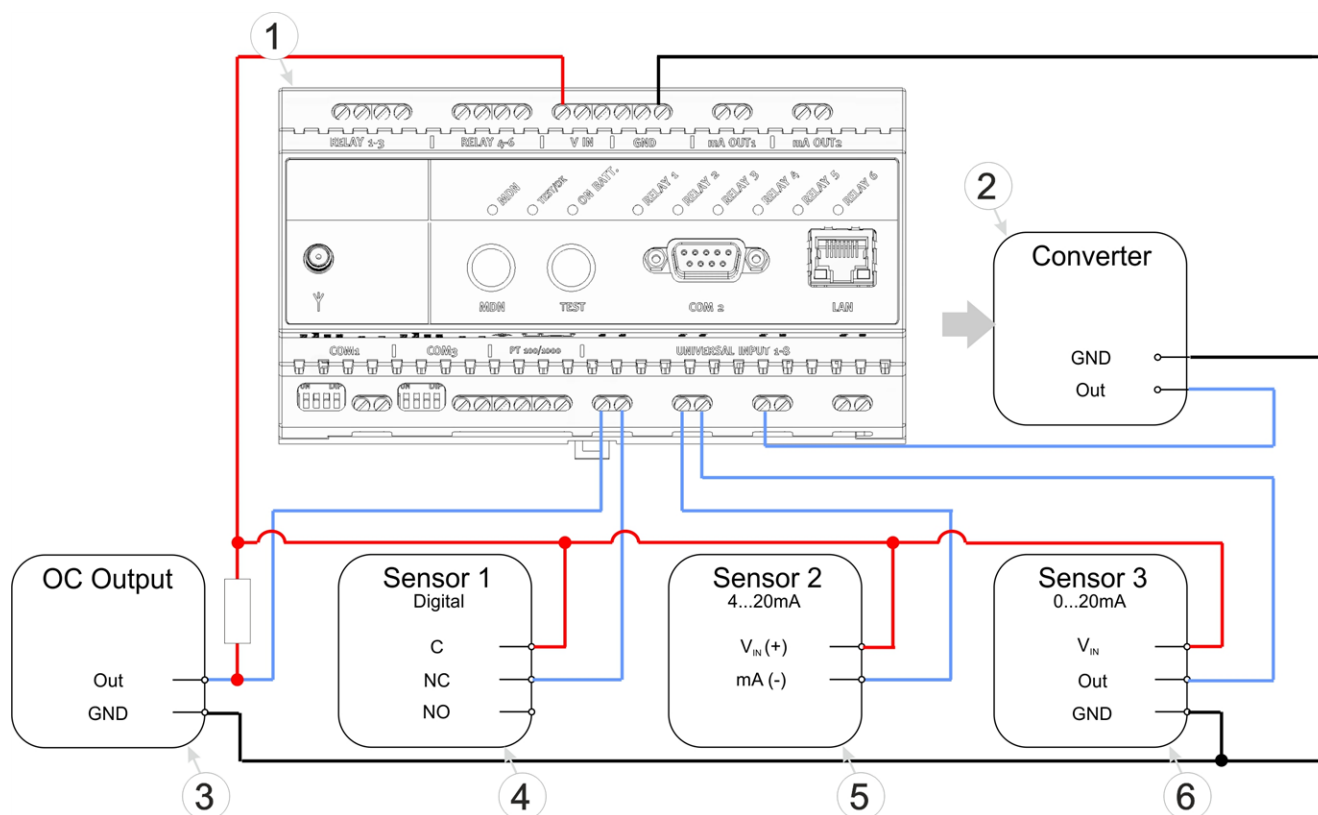
Connection examples

1	myDatalogMUC	4	Isolated relay contact
2	Signal converter, isolating transducer	5	2-wire mA sensor
3	Sensor with open collector output	6	3-wire mA sensor

7.4.2 Connection of the GSM antenna

> *Important note: To ensure the correct functionality, only use antennas that are supplied by the manufacturer.*

The standard antenna is directly attached to the antenna connector (see "Overview" on page 22) of the myDatalogMUC . In the event of a low radio signal strength, you can use the Flat antenna Smart Disc SMA-M 2,5m (206.816) or Flat Antenna Smart Disc US SMA-M 2,5m (206.818).

If the distance between the antenna position and the myDatalogMUC is too great, you can use a 2.5m Extension cable for antenna SMA-M/SMA-F 2,5m (206.807).

1. Ensure that the myDatalogMUC is de-energised.

2. If you need an antenna extension, connect this to the antenna first.

3. Connect the antenna extension or antenna directly to the antenna connector of the myDatalogMUC (see "Overview" on page 22).

> *Important note: Do not apply too much force when tightening the antenna. Do not use any tools to tighten the antenna or antenna extension; only tighten it manually.*

4. Switch the voltage supply of the myDatalogMUC back on.

The following step is not mandatory.

5. Check whether the connection to the myDatanet has worked correctly (see "Testing communication with the device" on page 72).

7.4.3 Technical details about the universal inputs

Note: The universal inputs are not galvanically isolated.

7.4.3.1 0/4 to 20mA mode

Note: Above 23,7mA, the relevant input becomes highly resistive (safety shutdown to prevent damage to the universal input).

Resolution	6,3µA
I_{max}	23,7mA
Load	96Ω

7.4.3.2 0 to 2V mode

Resolution	610µV
U_{max}	2,5V
Load	10k086

7.4.3.3 0 to 10V mode

Resolution	7,97mV
U_{max}	32V
Load	4k7

7.4.3.4 Standard digital modes (PWM, frequency, digital, day counter, impulse counter)

General	U_{max}	32V
	Low	<1,31V
	High	>2,61V
	Load	4k7
PWM	Measurement range	1...99%
	f_{max}	100Hz
	Minimum impulse length	1ms
Frequency	Measurement range	1...1000Hz
Day and impulse counter	Minimum impulse length	1ms

7.4.4 Technical details about the PT100/1000 interface

The interface for the external temperature sensor automatically detects whether a PT100 or PT1000 is being used. It is also possible to use three- or two-wire sensors. An additional link is required on two-wire sensors (see "PT100/PT1000 2-wire" on page 63).

PT100/PT1000 3-wire   PT100/PT1000 2-wire

7.4.5 Technical details about the modbus interfaces (Com1, Com3)

Note: Modbus interfaces Com1 and Com3 correspond to standard EIA-485.

The modbus interfaces Com1 and Com3 include an input common mode range that covers the full area specified for RS485 (-7V...+12V). Higher voltages damage the interface. Differential signals of more than +/-200mV within the specified input common mode range are detected correctly. In send mode, the output signal is in the range of1,5...3,3V .

Baud rate	300-115200
Stop bits	1,2
Parity	N, E, O
Data bits	7,8
Modes	RTU
	ASCII

Com1 and Com3 are RS485 interfaces that can be configured as a modbus master and a modbus slave (see "Interfaces" on page 80). In modbus master mode, one or more slaves can be connected to the myDatalogMUC . The load or clamp resistances can be connected via dip switch 1 (Com1) or dip switch 2 (Com3) (see "Connecting the sensors, actuators and power supply" on page 56).

S1	1k load resistance between RS485 A and B
S2	120Ω load resistance between RS485 A and B
S3	1k pull up to RS485 A
S4	1k pull down to RS485 B
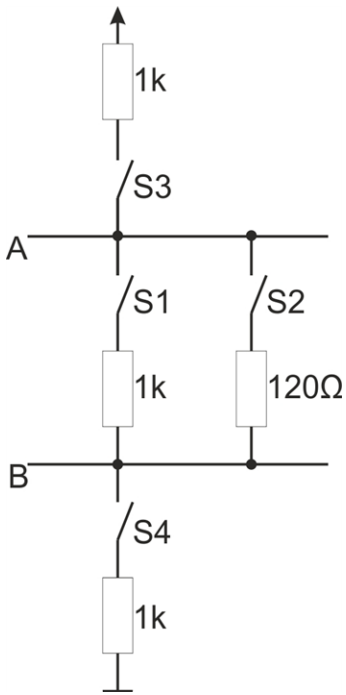
Schematic diagram of the switchable load resistances

Note: Additional explanation regarding the connection of two RS485 bus participants
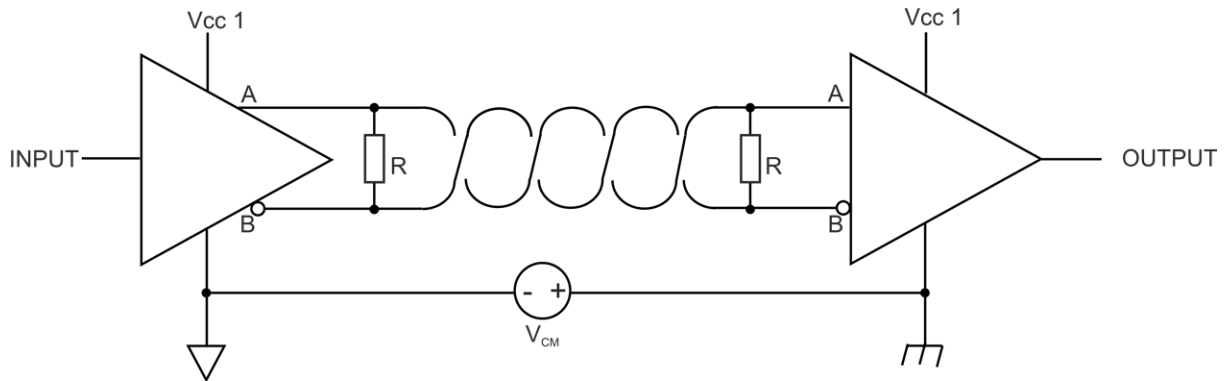
Schematic diagram: Connection of two RS485 bus participants

A problem occurs if there is no connection between the GND potentials of the sender and recipient. A common mode voltage (V_{CM}) occurs in this case. The GND potential difference must not exceed +/- 7V . Higher voltages will damage the interface. Temporary overvoltages (ESD, EFT and surge) are, however, absorbed by protective circuits.

Note: The common mode input voltage range of -7V...+12V specified for the RS485 is determined from the max. permissible GND potential difference (+/- 7V) and the max. permissible output voltage range of 0...5 V for RS485.

7.4.5.1 Modbus-Slave Modus

The following table details the possible access functions depending on the data type of the interface channel:

Modbus add.	Data type	Read function	Write function
0x0000 : 0x003F	Digital	Read coils (FC 01)	Write single coil (FC 05) Write multiple coils (FC 15)
0x0000 : 0x007F	Signed 16/32 bit Unsigned 16/32 bit Float	Read holding registers (FC 03)	Write single register (FC 06) Write multiple registers (FC 16)

The following table details the possible access functions depending on the data type of the interface output channel:

Modbus add.	Data type	Read function	Write function
0x0800 : 0x083F	Digital	Read discrete inputs (FC 02)	---
0x0800 : 0x087F	Signed 16/32 bit Unsigned 16/32 bit Float	Read input registers (FC 04)	---

7.4.6 Technical details about the RS232 interface (Com2)

> *Note: The modbus interface Com2 is compatible with standard TIA/EIA-232-F.*

> *Important note: The RS232 interface of the myDatalogMUC does not support hardware handshakes.*

The output drivers are protected against overloading and are not damaged by a short circuit to the GND or +/-15 V. The inputs are equipped with a 5 kΩ load resistance.

Baud rate	300-115200
Stop bits	1,2
Parity	N, E, O
Data bits	7,8
Modes	ASCII

The direction of the signals corresponds to that of a DCE (e.g. modem).
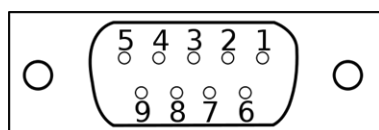
9-pin Sub-D(f)

Assignment of the Sub-D connector

Pin	Signal	Type
1	NC	
2	RXD	O (low: -5,7V; high: 6,2V)
3	TXD	I (low: <0,8V; high: >2,5V)
4	NC	
5	GND	
6	NC	
7	RTS[1]	I (low: <0,8V; high: >2,5V)
8	CTS[1]	O (low: -5,7V; high: 6,2V)
9	5V [2] max. 750mA	

[1] The current firmware does not support the hardware handeshake.

[2] Supply voltage (reserved for extensions)

If your sensor also comprises a SUB-D(f) connector, you can use the Gender changer 9-pin D-Sub male/male (206.684) provided as an accessory. If the connection characteristics (transmission direction of the individual signal lines) of your sensor also correspond to that of a DCE (e.g. modem), you can use the Null modem adapter 9-pin D-Sub female/male (206.686) that is available as an accessory.

7.4.6.1 Error codes of the RS232 interface

Log entry		Parameter		Description
Code	**Plain text**	**Code**	**Plain text**	
2000	LOG MODULE ERR	0	CHAR TIMEOUT	The time lag between two received characters is larger than the character timeout entered via the input screen for configuring the interface (see "Config" on page 90).
		1	FRAME TIMEOUT	The complete answer was not received within the frame timeout. This period of time is entered via the input screen for configuring the interface (see "Com2 (serial, RS232)" on page 87).
		2	INVALID FRAME	The answer frame of the sensor is invalid if, for example, the number of columns of the sensor does not correspond to the number entered via the configuration interface (see "Column" in chapter "Config." on page 112).

7.4.7 Technical details about the outputs

7.4.7.1 Analogue output 1-2 (OUT, GND)

> *Note: The analogue outputs of the myDatalogMUC are active, not galvanically isolated current outputs.*
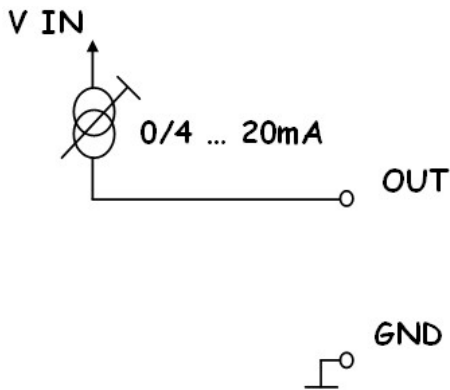
Replacement circuit diagram for the analogue output

Load voltage	Corresponds to the supply voltage (12...30VDC)
max. load (32V, 20mA)	1200Ω
$I_{out\ max}$	20mA

7.4.7.2 Relay 1-6

Every 3 relays are combined into a group with a joint root. In idle mode, the closed-circuit contacts of all of the relays are open (normally open).
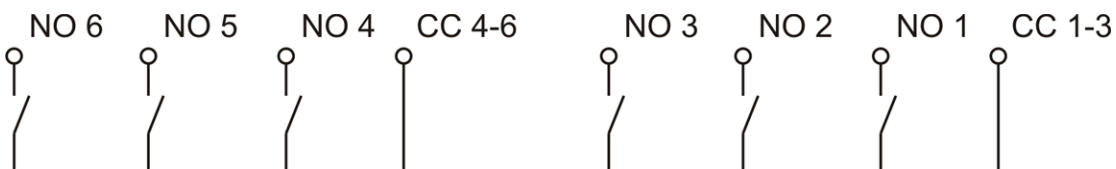
Replacement circuit diagram for the relays

U	240VAC
U_{max}	400VAC
I_{max}	3A[1]

[1]applies per relay. This means that up to 9A flows via CC.

7.4.8 Technical details about the integrated rechargeable buffer battery

The integrated rechargeable buffer battery enables a message to be issued in the event of a supply voltage failure. The alarm for the internal "voltage" measurement value (see "Internal channels" on page 131) must be configured for this purpose. The value that is entered for the "value low" alarm level is not important as the internal "voltage" measurement value is set to zero and the supply is switched to the rechargeable buffer battery as soon as the supply voltage falls below 9,5V . This initiates the malfunction message to be issued, as long as a time delay for issuing the malfunction message has not been entered via the "Delay for power off alarm" parameter in the "Basic settings" configuration section (see "Basic setting" on page 134). The switchover to the rechargeable buffer battery is definitely entered in the device log by the "BACKUP SUPPLY, 1" log entry. If a delay time has been configured, the myDatalogMUC , apart from the components that fail when supplied by the rechargeable buffer battery, continues to operate normally until the time delay has elapsed. If the supply voltage increases to above 10,5V again during the time delay, the supply is switched from the rechargeable buffer battery back to the supply voltage. This is recorded in the device log with the "BACKUP SUPPLY, 0" log entry. Issuing the malfunction message is initiated if the supply voltage is below 10,5V when the time delay elapses.

The following components fail as soon as the myDatalogMUC is only supplied by the rechargeable buffer battery:

- Analogue output 1-2
- LEDs to indicate the switching states of the relays
- Control of the relays (i.e. the closed-circuit contacts switch to "NO" idle mode)
- 5V supply to COM2

While the device establishes and maintains a connection to the server to issue the malfunction message, the LED lights up to indicate that the device is being supplied by the rechargeable buffer battery. Any outstanding data is also transmitted to the server during this process. The rechargeable buffer battery is then deactivated and the corresponding LED goes out. If the supply voltage has not broken down completely at this time, the supply voltage is only monitored on the device. All of the other operations are terminated. This state is maintained until the supply fails completely or increases above 10,5V . The myDatalogMUC resumes normal operation above 10,5V .

Devices that have informed the myDatanet server about the failure of their supply voltage are indicated in the site list with the symbol pictured below.


```
19.7.2012 15:02:41
    SER UTC+2,00
```

A hardware-regulated controller ensures that the rechargeable buffer battery is only charged when the ambient temperature is in the permitted range (0...+40°C).

V IN	Supply voltage: 12...30VDC (+/-10%)
$V_{recover}$	Threshold for switching to normal operation: 10,5V
V_{off}	Threshold for switching to supply via the rechargeable buffer battery: 9,5V
t_{delay}	Time delay for issuing the malfunction message. Is configured via the "Delay for power off alarm" parameter located in the "Basic settings" configuration section (see "Basic setting" on page 134).
Backup	Voltage of the rechargeable buffer battery

Supply of the myDatalogMUC

1	• The power supply is switched to the rechargeable buffer battery. • The "BACKUP SUPPLY, 1" log entry is created. • The device continues to operate normally (except for the components that fail when supplied by the rechargeable buffer battery).
2	• The time delay for issuing the malfunction message has elapsed. • The malfunction message is issued. The rechargeable buffer battery is then deactivated. The myDatalogMUC is subsequently switched off.
3	• The supply voltage exceeds the threshold for switching to normal operation again. • The "POWER ON" log entry is created. • The myDatalogMUC resumes normal operation again.
4	• The power supply is switched to the rechargeable buffer battery. • The "BACKUP SUPPLY, 1" log entry is created. • The device continues to operate normally (except for the components that fail when supplied by the rechargeable buffer battery).
5	• The supply voltage exceeds the threshold for switching to normal operation again. • The time delay for issuing the malfunction message has not elapsed yet. • The "BACKUP SUPPLY, 0" log entry is created. • NO malfunction message is issued.

7.4.9 Technical details regarding the energy supply
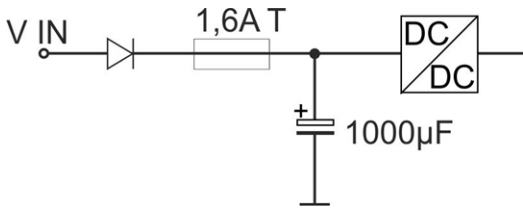
Schematic diagram of the energy supply

V IN	12...30VDC (+/-10%)
Power consumption [1] (without sensors)	typ. 1W max. 3W
Input capacity	1000µF
Fuse	1,6A T
Reverse voltage protection	Yes

[1] applies to ongoing operation. A current peak is caused by the input capacity at the time of activation.

The myDatalogMUC is equipped with a relatively large input capacity (1000µF) to ensure reliable switchover to the integrated rechargeable buffer battery in the event of a supply voltage failure. When selecting the power supply please ensure that it is able to supply the required initial current. A selection of compatible power supplies is included in the chapter "Power supply" on page 227. The supply voltage input is also equipped with a diode to protect against polarity reversal and a 1,6A T fuse.

Chapter 8 Initial Start-Up

8.1 User information

Before you connect the myDatalogMUC and place it into operation, you must observe and comply with the following user information!

This manual contains all information that is required for using the device.

Is intended for technically qualified personnel who have the relevant knowledge and experience in the area of measurement technology.

Read this manual carefully and completely in order to ensure the proper functioning of the myDatalogMUC .

Contact Microtronics Engineering GmbH(see "Contact information" on page 237) if anything is unclear or if you encounter difficulties with regard to installation, connection or configuration.

If necessary, also refer to the manuals of the accessory parts when commissioning the complete system. These manuals are included in the scope of delivery of the accessory parts.

8.2 General principles

The entire measurement system may only be placed into operation after completion and inspection of the installation. Study the manual thoroughly before placing into operation to prevent faulty or incorrect configuration.

Utilise the manual to familiarise yourself with the operation of the myDatalogMUC and the input screens of the myDatanet server before you begin with the configuration.

8.3 Placing the system into operation

> *Note: It is recommended that the myDatalogMUC is first placed into operation in the office before mounting the device permanently at the place of use. During this process, you should set a site for the later operation on the myDatanet server (see "myDatanet Server Manual " 805002) and determine a site configuration (see "Site configuration" on page 78). Take the opportunity to get to know the functions of the device in a stable environment. You can also use suitable test signals to simulate the sensors to establish the optimum configuration of the myDatalogMUC prior to its actual first use. This reduces the amount of time required for on-site installation to a minimum.*

1. Once you have completed all the steps described in the chapter "Connecting the sensors, actuators and power supply" on page 56, the myDatalogMUC is ready for operation and should have made its first connection to the myDatanet server.

2. Create a site for the operation on the myDatanet server (see "myDatanet Server Manual " 805002).

3. Configure the created site according to your requirements (see "Site configuration" on page 78).

4. Link the myDatalogMUC with the created site (see "Site" on page 79).

5. Activate the ALOHA transmission mode (see "ALOHA transmission mode" on page 44) so that the site configuration is transmitted to the myDatalogMUC .

8.4 Testing communication with the device

1. Create a site for the operation on the myDatanet server (see "myDatanet Server Manual " 805002).

2. Configure the created site according to your requirements (see "Site configuration" on page 78).

3. Link the myDatalogMUC with the created site (see "Site" on page 79).

4. Initiate the ALOHA transmission mode (see "ALOHA transmission mode" on page 44) so that the configuration of the site is transmitted to the myDatalogMUC .

5. Wait until it is indicated in the list of measurement instruments that the device is in ALOHA transmission mode. This is indicated by a speech bubble with the "Aloha" inscription.

The following steps are only required if you also wish to test the measurement value acquisition and data transmission.

6. Stop the ALOHA transmission mode by clicking the cross in the speech bubble with the "Aloha" inscription or wait for the duration of the ALOHA transmission mode. This period can be set in the basic settings (see "Basic setting" on page 134) of the site configuration. The default setting is 10 minutes.

7. Then wire up the sensors (see "Connecting the sensors, actuators and power supply" on page 56) and restart the ALOHA transmission mode.

   Important note: All wiring work must be performed in the de-energised state.

8. Check the incoming data in the ALOHA data window of the myDatanet server, which can be accessed by clicking on the speech bubble with the "Aloha" inscription (see "myDatanet Server Manual " 805002). Particular attention must be paid to the internal "GSM level" and "voltage" measurement values.

Note: Supplementary explanation on evaluating the "GSM level":

"GSM level"	
>-64dBm	
-64 to -73dBm	
-74 to -83dBm	
-84 to -93dBm	
-94 to -107dBm	
<= -108dBm	

Note: Additional explanation about evaluating the "voltage":

The displayed voltage should only deviate slightly from the supply voltage applied to the device.

Chapter 9 User interfaces

The configuration of the myDatalogMUC is carried out via the web interface on the myDatanet server (see "User interface on the myDatanet server" on page 78), which your responsible sales partner will provide to you.

9.1 User interface on the myDatalogMUC

9.1.1 Operating elements
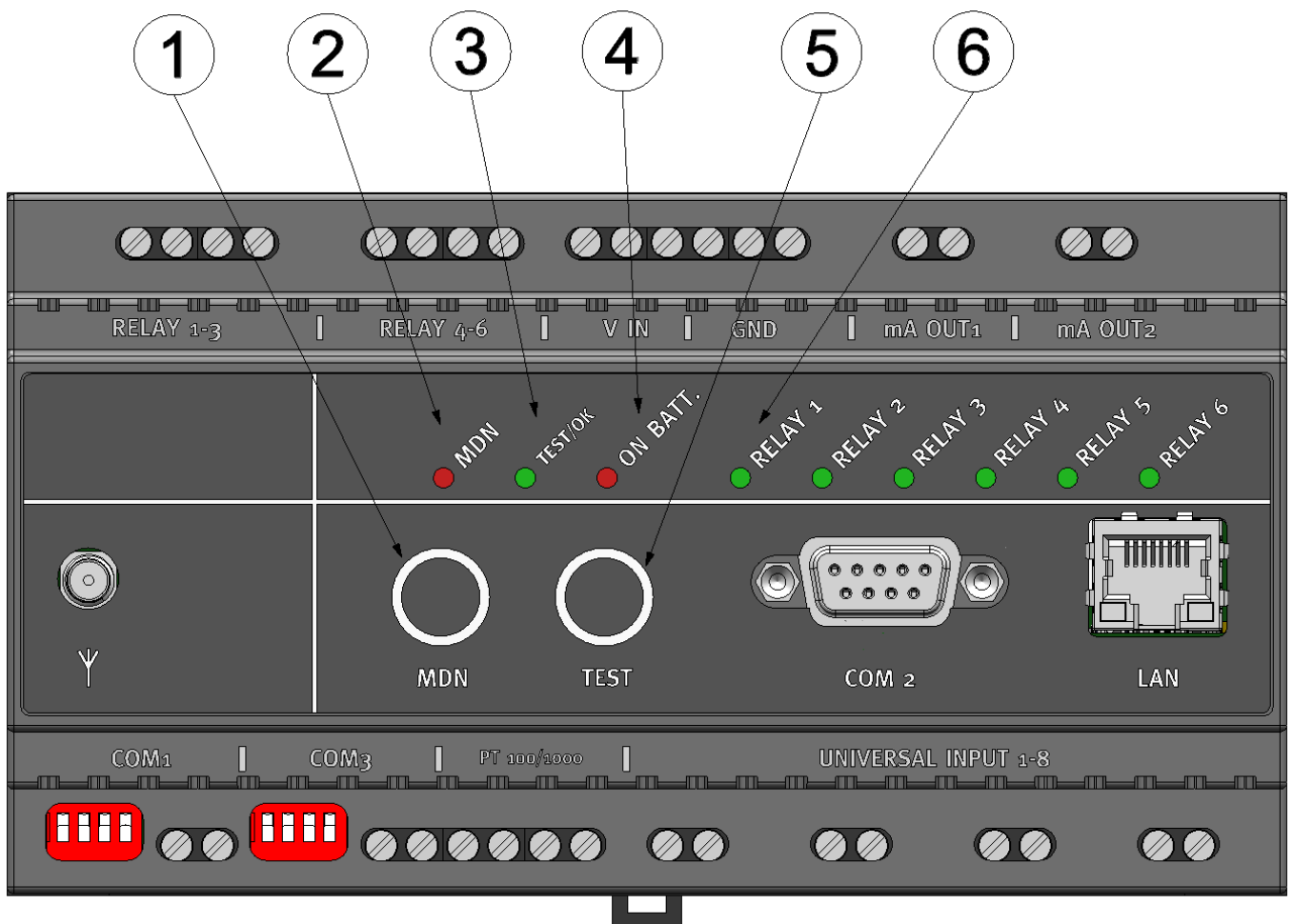
Operating elements

1	Button to initiate ALOHA transmission mode	4	Status display: buffer accu active
2	Status LED	5	Button to initiate self-testing
3	Status display: Self-testing	6	Status display: Switching states of the relays

9.1.1.1 Button to initiate ALOHA transmission mode

The button can be used to initiate the ALOHA transmission mode or to instruct the myDatalogMUC to immediately issue the error/status code.

User action	Device response	Operation after releasing the button
Press briefly for approx. one second	Status LED illuminates	Error/status code is displayed (see "Status LED" on page 76)
Press and hold for five seconds	Status LED flashes three times and then remains on	ALOHA transmission mode

9.1.1.2 Status LED

The status LED is used both to display the error/status codes and to indicate the current operating state. If the ALOHA transmission mode was activated or the power supply was provided (PowerOn), the status LED shows the current operating state for 10 minutes. During these 10 minutes, the error/status codes are issued every 3 seconds as long as there is no GPRS connection.

Error/status codes

Flashing code	Description	Solution/cause
0x	Transport lock (GPRS OFF, measurement OFF)	If the ALOHA transmission mode is initiated by button , the myDatalogMUC switches back into "RUN" mode (GPRS ON, measurement ON).
1x	Last connection OK	---
2x	Last transmission faulty	Try again later
4x	Standby (GPRS ON, measurement OFF)	see "transport lock"
6x	Offline (GPRS OFF, measurement ON)	see "transport lock"
7x	Network block/no matching provider	- Improve the position of the antenna - Check whether the device is located in the coverage area of one of the service providers supported by the integrated SIM chip (www.microtronics.com/footprint)
8x	No GSM network	- Try again later - Improve the position of the antenna
10x	No GPRS connection	Improve the position of the antenna
11x	No myDatanet server available	- Check whether port 51241 is enabled on the myDatanet server - Try again later
12x	Faulty SIM chip	Contact support

Operating states

Status LED	Description
Flickering	Establishing connection
Lights up	GPRS connection established or button actuated
Off	Normal measuring operation according to configuration until the next transmission

9.1.1.3 Status indication: Self-testing

This status display is designed to indicate that self-testing is in progress and also shows the result of the test. Self-testing is completed automatically when the power supply (PowerOn) is established. It can also be initiated by the operator at any time by pressing the "TEST" button (see "Button to initiate self-testing" on page 78). The analogue outputs and universal inputs are tested during this process. The sensor cables do not need to be removed.

TEST/OK	Description
Flashes	Self-testing in progress
Lights up	Self-testing successful, no hardware failure detected
Off	Hardware failure detected

A corresponding entry is generated in the device log if a hardware failure was detected. The cause of the problem can be specified more accurately with the help of the parameter saved with the error code. Instructions on evaluating the device log are included in the chapter "Evaluating the device log" (see "Evaluating the device log" on page 225).

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1020	ERROR SOD	## [1]	---	Hardware failure detected

[1] The following table provides an overview of the parameter coding:

Bit 0	Analogue output 1	Bit 4	Universal input 1
Bit 1	Analogue output 2	Bit 5	Universal input 2
Bit 2	External temperature sensor	Bit 6	Universal input 3
Bit 3	Reference voltage of the universal inputs	Bit 7	Universal input 4
		Bit 8	Universal input 5
		Bit 9	Universal input 6
		Bit 10	Universal input 7
		Bit 11	Universal input 8

Note: Example for identifying the cause of a hardware failure:

Assumption: The device log contains:

Log entry	Parameter
1020	4088

The parameter must be converted from the decimal notation (4088) to the binary notation (1111 1111 1000). It is now apparent that Bit3 to Bit11 are set. This means that in this example, the reference voltage and all of the universal inputs are defect.

9.1.1.4 Status display: buffer accu active

Important note: *Please note that when the device is being powered by the buffer accu, not all of the hardware components will be supplied (see "Technical details about the integrated rechargeable buffer battery" on page 68).*

ON ACCU	Description
Lights up	The myDatalogMUC is being powered by the buffer accu.
Off	The myDatalogMUC is supplied via the V IN and GND clamps.

9.1.1.5 Button to initiate self-testing

This button enables the operator to initiate self-testing at any time.

User action	Device response
Press and hold for five seconds	LED "TEST/OK" starts to flash and self-testing is started.

Further information on self-testing and diagnosing any problems is provided in "Status indication: Self-testing" on page 77.

9.1.1.6 Status display: Switching states of the relays

The myDatalogMUC is equipped with LEDs that display the switching state of each of the 6 relays.

RELAY x	Description
Lights up	The operating contact of the relevant relay is closed.
Off	The operating contact of the relevant relay is open (idle mode).

9.2 User interface on the myDatanet server

9.2.1 Site configuration

Note: *Several of the configuration fields in the following sub chapters may possibly be hidden depending on the respective user level. In this case, contact the myDatanet server administrator.*

Click on the name of the site in the list of sites to get to the input screen for configuring the site (see "myDatanet Server Manual " 805002).

9.2.1.1 Site

Customer
Specifies to which customer the site is assigned.

 symbol
Assign site to another customer

Name
Site designation (not relevant for the device or data assignment) [2-50 characters]

Device S/N
Serial number of the device that is linked to the site (device assignment!)

Application template
Name of the application template from which the site was created

Tags
List of tags that are already assigned to the site. This assignment can be cancelled by clicking on the cross next to the title of the tag.The input screen for assigning tags can be opened by clicking on the plus symbol.

9.2.1.2 Comments

Comments
Free comment field (is also displayed below the device type in the site list)

9.2.1.3 Control

Script type

Selection of the script type

Off	Script deactivated			
IL	Activates script processing and informs the myDatalogMUC that the script is an "Instruction list"			
	Script	Input window for editing the script that is loaded in the device		
Pawn	Activates script processing and informs the myDatalogMUC that the script is a "Pawn script" (see "Pawn script" on page 149)			
	Script source	Script	The script is directly entered and compiled via the server interface.	
			Script	Input window for editing the script that is loaded in the device
		Upload a compiled script	A script binary file (*.amx), that has already been compiled, should be uploaded.	
			File upload	Selection of the script binary file (*.amx) that is uploaded to the myDatanet server and is loaded into the myDatalogMUC during the next connection. The file path is only displayed as long as the input screen for configuring the site has not been closed.

9.2.1.4 Interfaces

9.2.1.4.1 Basis

Note: Modbus mode (master/slave) for interfaces Com1 and Com3 is selected in the "Config" tab (see "Config" on page 90).

9.2.1.4.1.1 Com1 and Com3 (modbus master, RS485, script parsing inactive)

Mode
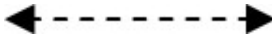
Off	Interface deactivated		
Modbus	activates the modbus functionality of the interface		
(1/2)	Baud rate	300	Selection of the required baud rate
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stop bits	1	Selection of the required stop bits
		2	
	Parity	None	Selection of the required parity
		Odd	
		Even	
	Data bits	7	Number of data bits to be used
		8	
	Response timeout	Time during which the modbus slave must react to the command from the device	

Modbus (2/2)	Hold	Hold the last valid measurement value for x measurement cycles	
		Off	Function deactivated
		1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
		On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
	Retry	In the event of a communication error, the corresponding command is repeated once. The error is only indicated if this second attempt also fails.	
Script parsing	Activates script parsing. This means that the interface can be accessed via the "Mdn_SerialEvent()", "Mdn_SerialRx()", "Mdn_SerialTx()" and "Mdn_SerialFinish()" PAWN script functions (see "Script parsing" on page 46). This deactivates the modbus functionality of the interface. Details on configuring the "Script Parsing" are provided in "Com1 and Com3 (RS485, script parsing active)" on page 84.		

Important note:

Example to explain the link between the "Response timeout", "Retry" and "Measurement cycle"

Basic setting	◀ - - - - - - ▶		Record interval	5min.
	◀━━━▶		Measurement cycle	1min.
Interfaces -> Basis	◀━▶	*First attempt*	Response timeout	4sec.
	◀━▶	*Retry*		
			Retry	Active

Explanation: In this example, seven channels are activated, however, the modbus slave does not answer. At the time of a measurement, an initial attempt is made to read the data for interface channel 1. As "Retry" is activated, another attempt is made to read the data for interface channel 1 from the modbus slave following expiry of the "Response timeout". Once the "Response timeout" has expired again, the error value "OL" (open loop) is set for interface channel 1 and the first attempt is made to read the data for interface channel 2.

Therefore, the number of activated channels (interface channels + interface output channels), the "Response timeout", "Retry" and measurement cycle must be selected as follows:

Retry not active: "Response timeout" * number of active channels < "Measurement cycle"

Retry active: "Response timeout" * 2 * number of active channels < "Measurement cycle"

9.2.1.4.1.2 Com1 and Com3 (modbus slave, RS485, script parsing inactive)

Mode

Off	Interface deactivated		
Modbus	activates the modbus functionality of the interface		
	Baud rate	300	Selection of the required baud rate
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stop bits	1	Selection of the required stop bits
		2	
	Parity	None	Selection of the required parity
		Odd	
		Even	
	Data bits	7	Number of data bits to be used
		8	
	Monitoring timeout	0...no monitoring Time during which the registers have to be renewed by the modbus master. The measurement values are labelled as "OL" (open loop) in the event of a timeout.	
	Hold	Hold the last valid measurement value for x measurement cycles	
		Off	Function deactivated
		1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
		On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

| Script parsing | Activates script parsing. This means that the interface can be accessed via the "Mdn_SerialEvent()", "Mdn_SerialRx()", "Mdn_SerialTx()" and "Mdn_SerialFinish()" PAWN script functions (see "Script parsing" on page 46). This deactivates the modbus functionality of the interface. |
| | Details on configuring the "Script Parsing" are provided in "Com1 and Com3 (RS485, script parsing active)" on page 84. |

9.2.1.4.1.3 Com1 and Com3 (RS485, script parsing active)

Mode

Off	Interface deactivated		
Modbus	activates the modbus functionality of the interface		
	Details on configuring "Modbus" mode are provided in "Basis" on page 80 or "Com1 and Com3 (modbus slave, RS485, script parsing inactive)" on page 83.		
Script parsing (1/2)	Activates script parsing. This means that the interface can be accessed via the "Mdn_SerialEvent()", "Mdn_SerialRx()", "Mdn_SerialTx()" and "Mdn_SerialFinish()" PAWN script functions (see "Script parsing" on page 46). This deactivates the modbus functionality of the interface.		
	Baud rate	300	Selection of the required baud rate
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stop bits	1	Selection of the required stop bits
		2	

Script parsing (2/2)	Parity	None	Selection of the required parity
		Odd	
		Even	
	Data bits	7	Number of data bits to be used
		8	
	Frame timeout	0...no timeout Time during which the connected digital sensor must send the full answer to a request. A communication error is detected if the time is exceeded.	
	Hold	Hold the last valid measurement value for x measurement cycles	
		Off	Function deactivated
		1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
		On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
	Retry	In the event of a communication error, the corresponding command is repeated once. The error is only indicated if this second attempt also fails.	

Important note:

Example to explain the link between the "Frame timeout", "Retry" and "Measurement cycle"

Basic setting	← – – – – – →	Record interval	5min.
	←→	Measurement cycle	1min.
Interfaces -> Basis	←→ First attempt	Frame timeout	12sec.
	←→ Retry		
		Retry	Active

Explanation: At the time of a measurement, the measurement data sequence is sent via the "Mdn_ SerialTx()" function (see "Script parsing" on page 46), although the sensor does not answer. As "Retry" is activated, the measurement data sequence is sent again once the "Frame timeout" has expired. If the "Frame timeout" expires again, the error value "OL" (open loop) is set for all of the interface channels that obtain their data via the relevant Com by means of the "Mdn_SetCh()" function (see Mdn_SetCh()). Afterwards, the data record that contains the setpoints of the interface output channels is sent via the interface by means of the "Mdn_SerialTx()" function.

The "Frame timeout", "Retry" and measurement cycle must therefore be selected as follows:

Retry not active: "Response timeout" < "Measurement cycle"

Retry active: "Response timeout" * 2 < "Measurement cycle"

9.2.1.4.1.4 Com2 (serial, RS232)

Mode

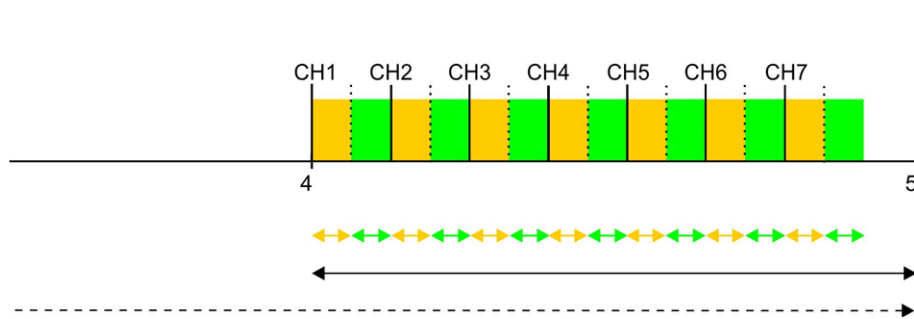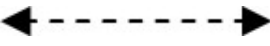| Off	Interface deactivated		
ASCII	Baud rate	300	Selection of the required baud rate
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stop bits	1	Selection of the required stop bits
		2	
	Parity	None	Selection of the required parity
		Odd	
		Even	
	Data bits	7	Number of data bits to be used
		8	
	Frame timeout	0...no timeout	
		Time during which the connected digital sensor must send the full answer to a request. A communication error is detected if the time is exceeded.	
	Hold	Hold the last valid measurement value for x measurement cycles	
		Off	Function deactivated
		1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
		On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
	Retry	In the event of a communication error, the corresponding command is repeated once. The error is only indicated if this second attempt also fails.	

Script parsing	Activates script parsing. This means that the interface can be accessed via the "Mdn_SerialEvent()", "Mdn_SerialRx()", "Mdn_SerialTx()" and "Mdn_SerialFinish()" PAWN script functions (see "Script parsing" on page 46). The option to determine via the "Config" tab of the "Interfaces" configuration section how the received characters should be evaluated is thus deactivated.		
	Baud rate	300	Selection of the required baud rate
		600	
		1200	
		2400	
		4800	
		9600	
		19200	
		38400	
		57600	
		115200	
	Stop bits	1	Selection of the required stop bits
		2	
	Parity	None	Selection of the required parity
		Odd	
		Even	
	Data bits	7	Number of data bits to be used
		8	
	Frame timeout	0...no timeout	
		Time during which the connected digital sensor must send the full answer to a request. A communication error is detected if the time is exceeded.	
	Hold	Off	Function deactivated
		1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
		On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
	Retry	In the event of a communication error, the corresponding command is repeated once. The error is only indicated if this second attempt also fails.	

Important note:

Example to explain the link between the "Frame timeout", "Retry" and "Measurement cycle"

Basic setting		Record interval	5min.
		Measurement cycle	1min.
Interfaces -> Basis	First attempt	Frame timeout	12sec.
	Retry		
		Retry	Active

Explanation: At the time of the measurement, the measurement data sequence is sent (see "Sequences for COM2" on page 91), although the sensor does not answer. As "Retry" is activated, the measurement data sequence is sent again once the "Frame timeout" has expired. If the "Frame timeout" expires again, the error value "OL" (open loop) is set for all of the interface channels that obtain their data via Com2. The data record that contains the setpoints of the interface output channels is then sent.

The "Frame timeout", "Retry" and measurement cycle must therefore be selected as follows:

Retry not active: "Response timeout" < "Measurement cycle"

Retry active: "Response timeout" * 2 < "Measurement cycle"

9.2.1.4.2 Config

9.2.1.4.2.1 Com1 and Com3

Mode

Off	Interface deactivated				
Modbus	Modbus mode	Master	Com mode	RTU	Data is transferred in binary format.
				ASCII	Data is transferred in ASCII format.
		Slave	Com mode	RTU	Data is transferred in binary format.
				ASCII	Data is transferred in ASCII format.
			Slave Ad.	Modbus slave address	
Script Parsing	---				

9.2.1.4.2.2 Com2

Mode

Off	Interface deactivated		
ASCII	Warmup time	Time from switching on the myDatalogMUC until the init sequence is sent.	
	Frame structure	...[CR][LF]	Each frame must be terminated with both "[CR][LF]" characters.
		...[ETX]	Each frame must be terminated with the "[ETX]" character.
		[STX]...[CR][LF]	Each frame must start with the "[STX]" character and terminate with both of the "[CR][LF]" characters.
		[STX]...[ETX]	Each frame must start with the "[STX]" character and terminate with the "[ETX]" character.
	Character timeout	0...no timeout Maximum permissible time interval between the full receipt of two characters. A communication error is detected if the time is exceeded. The monitoring process starts following receipt of the first character of an answer from the digital sensor.	
	Number delimiter	Divides the ASCII data message of the digital sensor into individual measurement values. Up to ten different delimiters can be entered via the input field. The following escape codes are also supported as delimiters: \a \b \f \n \r \t \v \\ \? \' \" \xhh The maximum number of delimiters is reduced when escape codes are used.	
	Number format	1,000.00	"," is used as the thousand separator "." is used as the decimal point
		1.000,00	"." is used as the thousand separator "," is used as the decimal point
Script Parsing	Warmup time	Time from switching on the myDatalogMUC until the init sequence is sent.	

9.2.1.5 Sequences for COM2

Note: This configuration section is only visible if "ASCII" mode is activated for the COM2 interface in the "Interfaces" configuration section (see "Basis" on page 80). If "Script parsing" mode is activated, the init sequence and measurement data sequence must be generated by the PAWN script.

Init sequence

Initialisation command that is sent to the digital sensor following expiry of the warmup time when the myDatalogMUC is switched on.

The following escape codes are supported:

\a \b \f \n \r \t \v \\ \? \' \" \xhh

Measurement data sequence

Command that is sent to the digital sensor at the time of every measurement [0-512 characters]

The following escape codes are supported:

\a \b \f \n \r \t \v \\ \? \' \" \xhh

9.2.1.6 Measurement channels

9.2.1.6.1 Basis

Title 1-8
Freely selectable channel title for the universal inputs

Title PT 100/1000
Freely selectable channel title for the external temperature sensor

Mode

Basic settings for the measurement channel

Universal inputs	Off	---	Measurement channel deactivated
(digital modes)	Digital	Invert	Inverts the input signal
	Cnt.Day	Impulse	Counted measurand of a pulse in the measurement unit
		Max	Defines the upper scale end of the pointer instruments
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	Cnt.Intrvl.	Impulse	Counted measurand of a pulse in the measurement unit
		Max	Defines the upper scale end of the pointer instruments
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	Freq	Factor	Factor by which the input signal is multiplied
		Min	Defines the lower scale end of the pointer instruments
		Max	Defines the upper scale end of the pointer instruments
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	PWM	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements

Universal inputs (analogue modes)	0-20 mA	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	4-20 mA	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	0-2V	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
	0-10V	0%	Start of the measurement range in the measurement unit
		100%	End of the measurement range in the measurement unit
		Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
		Decimal places	Number of decimal places that are used by all of the server display elements
Ext. temperature sensor	Off	---	Measurement channel deactivated
	On	Min	Defines the lower scale end of the pointer instruments
		Max	Defines the upper scale end of the pointer instruments
		Unit	Selection of the temperature unit used by all of the server display elements
		Decimal places	Number of decimal places that are used by all of the server display elements

9.2.1.6.2 Config

Universal inputs	Off	---	---	
(digital modes)	Digital	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			up	Minimum signal length for x seconds with a rising edge
			down	Minimum signal length for x seconds with a falling edge
			up&down	Minimum signal length for x seconds on both edges
		Time	Time x, that is used for decay modes "up", "down" and "up & down"	

Universal inputs (Counter modes)	Cnt.Day	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).	
		Reset at	Reset time of the day counter	
	Cnt.Intrvl.	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	

Note: Additional explanation regarding the difference between "Cnt.Day" and "Cnt.Intrvl."

Basic setting	◄- - - - - - - ►	Record interval	4 min.	Recorded value	red line
	◄——————►	Measurement cycle	1 min.	Measurement value	blue line
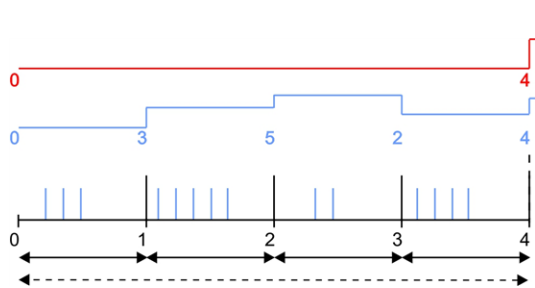
"Cnt.Intrvl." mode: The pulses are added up and reset every time a measurement value is generated.
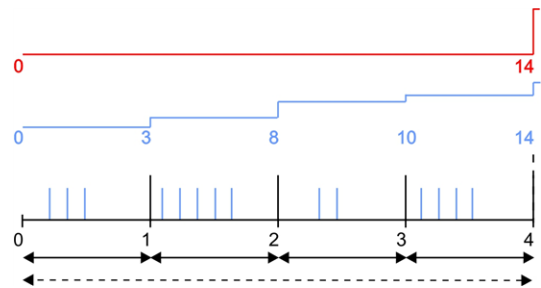
"Cnt.Day" mode: All of the pulses up to the reset time are added up.

Note on "Cnt.Intrvl." mode: *If the record interval is larger than the measurement cycle, only the number of pulses registered the last time the measurement value was generated is recorded. In this case, the individual measurement values can be summarised by means of the decay.*

Universal inputs (Frequency mode 1/2)	Freq (1/2)	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

Universal inputs (Frequency mode 2/2)	Freq (2/2)	Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	- If the measurement value is below 1Hz, the error value "UF" (underflow) is issued. - The error value "OF" (overflow) is issued, if the measurement value is above 1000Hz.
			NAMUR borders	- If the measurement value is below 1Hz, the error value "UF" (underflow) is issued. - The error value "OF" (overflow) is issued, if the measurement value is above 1000Hz.

Universal inputs (PWM mode 1/2)	PWM (1/2)	Filter time	Time in [ms] during which the signal must remain constant to initiate a level change. Used to suppress brief faults (debouncing).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

Universal inputs (PWM mode 2/2)	PWM (2/2)	Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	• If the measurement value is below 1%, the error value "UF" (underflow) is issued. • The error value "OF" (overflow) is issued, if the measurement value is above 99%.
			NAMUR borders	• If the measurement value is below 1%, the error value "UF" (underflow) is issued. • The error value "OF" (overflow) is issued, if the measurement value is above 99%.

Universal inputs (0-20mA mode 1/2)	0-20mA (1/2)	Filter time	Time in [ms] during which the analogue signal is averaged for signal smoothing. Used to suppress signal noise (also see "Example to explain the filter time in conjunction with the ext. warmup time" on page 131).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

Universal inputs (0-20mA mode 2/2)	0-20mA (2/2)	Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	• If the measurement value is above 20.1 mA, the error value "SC" (short circuit) is issued.
			NAMUR borders	• The error value "OF" (overflow) is issued, if the measurement value is above 20.1mA. • If the measurement value is above 21mA, the error value "SC" (short circuit) is issued.

Note: Additional explanation regarding the "Out of range" and "Namur borders" modes.

"Out of range" mode   *"Namur borders" mode*

Universal inputs (4-20mA mode 1/2)	4-20 mA (1/2)	Filter time	Time in [ms] during which the analogue signal is averaged for signal smoothing. Used to suppress signal noise (also see "Example to explain the filter time in conjunction with the ext. warmup time" on page 131).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

Universal inputs (4-20mA mode 2/2)	4-20mA (2/2)	Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	- The error value "OL" (open loop) is issued, if the measurement value is below 3.9mA. - If the measurement value is above 20.1 mA, the error value "SC" (short circuit) is issued.
			NAMUR borders	- The error value "OL" (open loop) is issued, if the measurement value is below 3.6mA. - The error value "UF" (underflow) is issued, if the measurement value is below 3.9mA. - The error value "OF" (overflow) is issued, if the measurement value is above 20.1mA. - If the measurement value is above 21mA, the error value "SC" (short circuit) is issued.

Note: Additional explanation regarding the "Out of range" and "Namur borders" modes.

"Out of range" mode

"Namur borders" mode

Universal inputs (0-2V mode)	0-2V	Filter time	Time in [ms] during which the analogue signal is averaged for signal smoothing. Used to suppress signal noise (also see "Example to explain the filter time in conjunction with the ext. warmup time" on page 131).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
		Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	The error value "OF" (overflow) is issued, if the measurement value is above 2.01V.
			NAMUR borders	The error value "OF" (overflow) is issued, if the measurement value is above 2.01V.

Note: Additional explanation regarding the "Out of range" and "Namur borders" modes.

"Out of range" or "Namur borders" mode

Universal inputs (0-10V mode)	0-10V	Filter time	Time in [ms] during which the analogue signal is averaged for signal smoothing. Used to suppress signal noise (also see "Example to explain the filter time in conjunction with the ext. warmup time" on page 131).	
		Decay	Temporal function in the measurement cycle	
			Off	Decay deactivated
			min	The minimum of the last x measurement values is recorded.
			max	The maximum of the last x measurement values is recorded.
			avg	The arithmetic mean of the last x measurement values is recorded.
			med	The median of the last x measurement values is recorded.
			rms	The root mean square of the last x measurement values is recorded.
		Time	Decay period. To calculate the number of considered measurement values see "Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay" on page 136. The "DECAY MEM ERR" error is entered in the device log if no temporary memory could be reserved to take another measurement value into consideration (see "Log entries and error codes " on page 216).	
		Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.
		Overflow	Procedure in the event of measurement range violations	
			Ignore	The measurement value is calculated beyond the range limits.
			Silent cutoff	The measurement value is truncated at the range limits.
			Out of range	The error value "OF" (overflow) is issued, if the measurement value is above 10.05V.
			NAMUR borders	The error value "OF" (overflow) is issued, if the measurement value is above 10.05V.

Note: Additional explanation regarding the "Out of range" and "Namur borders" modes.
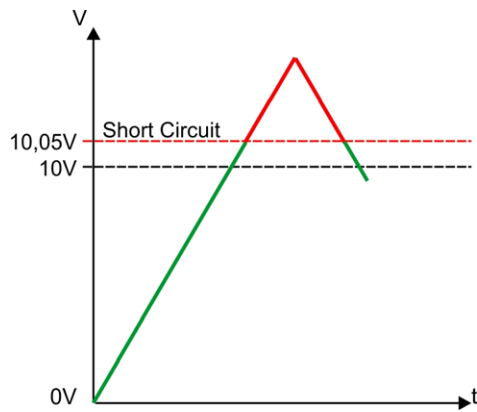
"Out of range" or "Namur borders" mode

Ext. temperature sensor	Off	---	Measurement channel deactivated	
	On	Hold	Hold the last valid measurement value for x measurement cycles	
			Off	Function deactivated
			1-5	Number of measurement cycles for which the measurement value is held until the error value is issued
			On	In the event of an error, the last valid measurement value is held until a new valid measurement value is present.

9.2.1.6.3 Alarms

"Digital" mode	WA	A "high" at the universal input triggers a "warning".	
	AL	A "high" at the universal input triggers an "alarm".	
	SW	A "high" at the universal input triggers a "fault warning".	
	SA	A "high" at the universal input triggers a "fault alarm".	
All other modes	Warning	Value low	A warning is triggered, if the measurement value drops to or below this value.
		Value high	A warning is triggered, if the measurement value meets or exceeds this value.
	Alarm	Value low	An alarm is triggered, if the measurement value drops to or below this value.
		Value high	An alarm is triggered, if the measurement value meets or exceeds this value.
	Hyst %	Hysteresis for the all-clear in the event of an alarm/warning (e.g. Hyst=5%, alarm or warning at 100 -> all-clear at 95) or reset the trigger (e.g. Hyst=5%, level = greater or equal, trigger at 100 -> reset at 95)	

9.2.1.6.4 Trigger

If an universal input is operated in digital mode or LP digital mode, there are two different types of triggers:

- Event trigger (MS, XM, I1-I4)

  In contrast to the level triggers, the relevant operation (e.g. initiate transmission) is only executed once when the trigger event occurs. With the help of the "Edge" configuration parameter, it is specified whether the rising, falling or both edges initiate the trigger event.

- Level trigger (QU, SL, ON)

  A "high" at the universal input initiates the trigger. A "low" at the universal input resets the trigger. The relevant operation (e.g. activate online mode) is executed as long as the trigger is active. The option selected via the "Edge" configuration parameter is not relevant to the level trigger. If it is necessary for the trigger to be initiated by a "low" at the universal input and for it to be reset by a "high", the input signal must then be inverted using the "Invert" configuration parameter located in the "Basis" tab.

These two types of triggers are also differentiated in other universal input modes:

- Event trigger (XM)

  The relevant operation (e.g. initiate transmission) is only executed once when the trigger event occurs.

- Level trigger (QU, SL, RO, RF, ON, I1-I4)

  The relevant operation (e.g. activate online mode) is executed as long as the trigger is active.

"Digital" mode	Event trigger	MS	Start measurement cycle immediately	
		XM	Initiate transmission	
		I1	Internal use with the control program	
		I2	Internal use with the control program	
		I3	Internal use with the control program	
		I4	Internal use with the control program	
		Edges	Selection of the edge at which the trigger should be initiated	
			rising	The rising edge initiates the trigger.
			falling	The falling edge initiates the trigger.
			both	Both edges initiate the trigger.
	Level trigger	QU	Fastrecording (record interval = record interval / factor)	
		SL	Slow recording (record interval = record interval * factor)	
		ON	Activate online mode	
		Edges	not relevant for the level triggers	

All other modes	Event trigger	XM	Initiate transmission	
		Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
			Greater or equal	The trigger is initiated, if the measurement value meets or exceeds this value.
			Less or equal	The trigger is initiated, if the measurement value drops to or below this value.
	Level trigger	QU	Fastrecording (record interval = record interval / factor)	
		SL	Slow recording (record interval = record interval * factor)	
		RO	Switch on recording	
		RF	Switch off recording	
		ON	Activate online mode	
		I1	Internal use with the control program	
		I2	Internal use with the control program	
		I3	Internal use with the control program	
		I4	Internal use with the control program	
		Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
			Greater or equal	The measurement value must be greater or equal to the level to initiate the trigger.
			Less or equal	The measurement value must be less or equal to the level to initiate the trigger.

9.2.1.7 Interface channels 1-32

Note: Some of the configuration parameters included in this configuration section also affect parameters that are not included in the tab that is currently open. In the following section, this is indicated by the ">" symbol, for example "Basis > Min" means that the "Min" configuration parameter is located in the "Basis" tab.

An interface channel is designed to record a single measurement value that is read from one of the 3 serial interfaces (2 x RS485, 1 x RS232). The data type that should be used and the interface, from which the data is read, can be determined for every interface channel independently of the other channels. The following chapters contain a description on configuring the interface channels.

9.2.1.7.1 Basis

Title 1-32
Freely selectable channel title for the interface channels [0-16 characters]

Interface

Basis settings for the measurement channel

Off	---		Measurement channel deactivated	
Com1 Com3	Scale > Scale (siehe "Scale" on page 115)	off	Min	Defines the lower scale end of the pointer instruments
			Max	Defines the upper scale end of the pointer instruments
		on	---	
	Config > Format (see "Config." on page 112)	Digital	---	
		Signed	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
		Unsigned	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
		Float	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
Com2	Min		Defines the lower scale end of the pointer instruments	
	Max		Defines the upper scale end of the pointer instruments	
	Unit		String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values	
	Decimal places		Number of decimal places that are used by all of the server display elements	

Script [1]	Min	Defines the lower scale end of the pointer instruments
	Max	Defines the upper scale end of the pointer instruments
	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
	Decimal places	Number of decimal places that are used by all of the server display elements

[1] *The measurement value is not read via one of the interfaces, but can be set by the script (see "Mdn_SetCh()" in chapter "Messkanäle" on page 154).*

Mode

Specifies how the interface channel should be handled by the evaluation elements of the server

Analogue/digital	Analogue or digital measurement value. This means that the system deals with each measurement value independently of the measurement values before or afterwards.
Cnt.Day	Day counter. This means that the system anticipates that the measurement value of the channel will continuously increase and is reset once per day.
Cnt.Intrvl.	Interval counter. This means that the system assumes that the counter reading is reset each time a measurement is recorded.
Cnt.Inf.	Infinite counter. This means that the system expects that the measurement value of the channel will continuously increase and never be reset.

9.2.1.7.2 Config.

Note: The following parameters are not available for an interface channel that is connected to an interface for which "Script parsing" was activated (see "Basis" on page 80).

9.2.1.7.2.1 Com1 and Com3 (modbus master, RS485)

Slave Ad.
Address of the modbus slave

Modbus Ad.
Address of the register that should be read

Format

Data type

Digital	A digital value should be read.			
	Function	Read coils (FC 01)		
		Read discrete inputs (FC 02)		
Signed	A signed integer value should be read.			
	Bit	16	16-bit integer	
		32	32-bit integer. Two registers must be read from the modbus slave for this purpose.	
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
	Function	Read holding registers (FC 03)		
		Read input registers (FC 04)		
Unsigned	The integer value to be read is not signed.			
	Bit	16	16-bit integer	
		32	32-bit integer. Two registers must be read from the modbus slave for this purpose.	
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
	Function	Read holding registers (FC 03)		
		Read input registers (FC 04)		
Float	A 32-bit float should be read.			
	Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address	
		LO-HI	LO word on the lower register address and the HI word on the higher register address	
	Function	Read holding registers (FC 03)		
		Read input registers (FC 04)		

9.2.1.7.2.2 Com1 and Com3 (modbus slave, RS485)

The following table details the possible access functions depending on the data type of the interface channel:

Modbus add.	Data type	Read function	Write function
0x0000 : 0x003F	Digital	Read coils (FC 01)	Write single coil (FC 05) Write multiple coils (FC 15)
0x0000 : 0x007F	Signed 16/32 bit Unsigned 16/32 bit Float	Read holding registers (FC 03)	Write single register (FC 06) Write multiple registers (FC 16)

Modbus Ad.
Address of the register that should be read

Format

Data type

Digital	*A digital value should be read.*			
Signed	*A signed integer value should be read.*			
	Bit	*16*	*16-bit integer*	
		32	*32-bit integer. Two registers must be read from the modbus slave for this purpose.*	
			Word order	*HI-LO*	*HI word on the lower register address and the LO word on the higher register address*
				LO-HI	*LO word on the lower register address and the HI word on the higher register address*
Unsigned	*The integer value to be read is not signed.*			
	Bit	*16*	*16-bit integer*	
		32	*32-bit integer. Two registers must be read from the modbus slave for this purpose.*	
			Word order	*HI-LO*	*HI word on the lower register address and the LO word on the higher register address*
				LO-HI	*LO word on the lower register address and the HI word on the higher register address*
Float	*A 32-bit float should be read.*			
	Word order	*HI-LO*	*HI word on the lower register address and the LO word on the higher register address*	
		LO-HI	*LO word on the lower register address and the HI word on the higher register address*	

9.2.1.7.2.3 Com2 (serial, RS232)

Column

> *The ASCII data message from the digital sensor is split into individual measurement values by the "number delimiter" (see "Interfaces" on page 80). The "column" configuration parameter specifies which of these measurement values is linked with/recorded on the interface channel.*

9.2.1.7.3 Scale

> *Note: The following parameters are not available for an interface channel that is connected to an interface for which "Script parsing" was activated (see "Basis" on page 80).*

9.2.1.7.3.1 Com1 and Com3 (modbus, RS485)

Scale	Off	---			
	On	0% modbus	Start of the measurement range in the unit of the modbus slave		
		100% modbus	End of the measurement range in the unit of the modbus slave		
		0% modbus	Start of the measurement range in the measurement unit		
		100% modbus	End of the measurement range in the measurement unit		
		Overflow	Procedure in the event of measurement range violations		
			Ignore	The measurement value is calculated beyond the range limits.	
			Silent cutoff	The measurement value is truncated at the range limits.	
			Overflow	• The error value "UF" (underflow) is issued, if the measurement value is below the lower limit. • The error value "OF" (overflow) is issued, if the measurement value is above the upper limit.	

Note: *Calculation of the measurement value during active scaling:*

Basis > min	4mA	D
Basis > max	20mA	C
Min modbus	0	A
Max modbus	65535	B

Measurement value in the unit of the modbus slave	32767	E
Scaled measurement value	11.9998mA	F

$$F = ((C - D)/(B - A)) * (E - A) + D$$
$$F = ((20mA - 4mA) /(65535 - 0)) * (32767 - 0) + 4mA$$
$$=11.9998 \text{ mA}$$
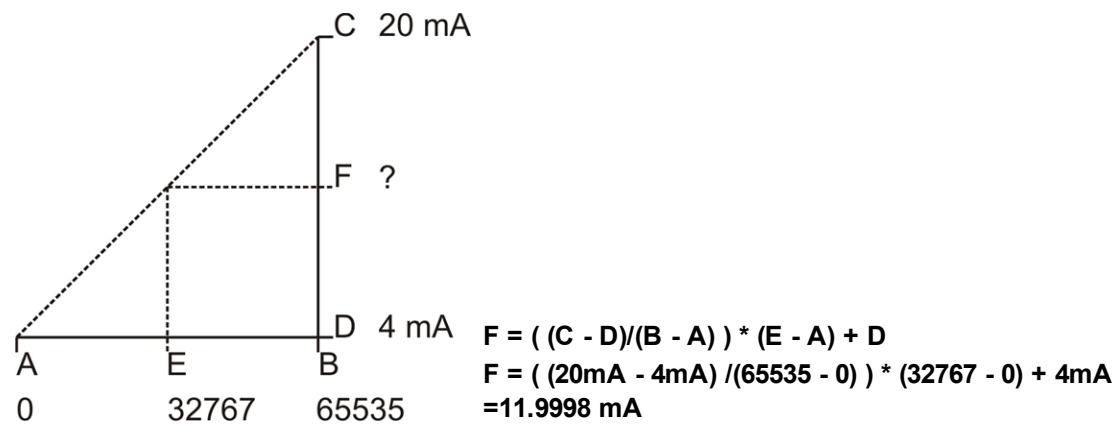
9.2.1.7.3.2 Com2 (serial, RS232)

The scaling is not available for the serial interface.

9.2.1.7.4 Alarms

Note: If "Script" is used as the basic setting for the measurement channel, alarms cannot be triggered automatically by the system, as the alarm/trigger module is executed before the control module in which script processing is completed (see "Internal processing of the measurement values" on page 31). However, the alarm configuration that can be entered via this tab can be accessed via the script and an alarm can be triggered via the script (see "Mdn_GetAlarmCfg()" and "Mdn_SetAlarm()" in chapter "Functions" on page 164).

Warning	Value low	A warning is triggered, if the measurement value drops to or below this value.
	Value high	A warning is triggered, if the measurement value meets or exceeds this value.
Alarm	Value low	An alarm is triggered, if the measurement value drops to or below this value.
	Value high	An alarm is triggered, if the measurement value meets or exceeds this value.
Hyst %	Hysteresis for the all-clear in the event of an alarm/warning (e.g. Hyst=5%, alarm or warning at 100 -> all-clear at 95) or reset the trigger (e.g. Hyst=5%, level = greater or equal, trigger at 100 -> reset at 95)	

9.2.1.7.5 Trigger

Note: If "Script" is used as the basic setting for the measurement channel, triggers cannot be initiated automatically by the system, as the alarm/trigger module is executed before the control module in which script processing is completed (see "Internal processing of the measurement values" on page 31). However, the trigger configuration that can be entered via this tab can be accessed via the script and a trigger can be initiated via the script (see "Mdn_GetTriggerCfg()" and "Mdn_SetTrigger()" in chapter "Functions" on page 164).

The following two type of triggers are differentiated:

- Event trigger (XM)

  The relevant operation (e.g. initiate transmission) is only executed once when the trigger event occurs.

- Level trigger (QU, SL, RO, RF, ON, I1-I4)

  The relevant operation (e.g. activate online mode) is executed as long as the trigger is active.

Event trigger	XM	Initiate transmission	
	Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
		Greater or equal	The trigger is initiated, if the measurement value meets or exceeds this value.
		Less or equal	The trigger is initiated, if the measurement value drops to or below this value.
Level trigger	QU	Fastrecording (record interval = record interval / factor)	
	SL	Slow recording (record interval = record interval * factor)	
	RO	Switch on recording	
	RF	Switch off recording	
	ON	Activate online mode	
	I1	Internal use with the control program	
	I2	Internal use with the control program	
	I3	Internal use with the control program	
	I4	Internal use with the control program	
	Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
		Greater or equal	The measurement value must be greater or equal to the level to initiate the trigger.
		Less or equal	The measurement value must be less or equal to the level to initiate the trigger.

9.2.1.8 Interface channels 33-64

9.2.1.8.1 Basis

Title 33-64
Freely selectable channel title for the interface channels

The significance of the remaining configuration parameters in this configuration section corresponds to the "Interface channels 1-32" configuration section (see "Interface channels 1-32" on page 110).

9.2.1.9 Interface output channels 1-32

Note: Some of the configuration parameters included in this configuration section also affect parameters that are not included in the tab that is currently open. In the following section, this is indicated by the ">" symbol. For example, "Basis > Min" means that the "Min" configuration parameter is located in the "Basis" tab.

An interface output channel is designed to receive a single setpoint that is transferred to a sensor or actuator via one of the 3 serial interfaces (2 x RS485, 1 x RS232). The data type that should be used and the interface, via which the data should be transferred, can be determined for every interface output channel independently of the other channels. The following chapters contain a description on configuring the interface output channels.

9.2.1.9.1 Basis

Title 1-32
Freely selectable channel title for the interface output channels [0-16 characters]

Interface

Basis settings for the output channel:

off	---	Output channel deactivated		
Com1 Com3	Scale > Scale (siehe "Scale" on page 123	off	Min	Defines the lower scale end of the pointer instruments
			Max	Defines the upper scale end of the pointer instruments
		on	---	
	Config > Format (see "Config" on page 120)	Digital	---	
		Signed	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
		Unsigned	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
		Float	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
			Decimal places	Number of decimal places that are used by all of the server display elements
	Setpoint	Output value in the measurement unit		
Com2	Min	Defines the lower scale end of the pointer instruments		
	Max	Defines the upper scale end of the pointer instruments		
	Setpoint	Output value in the measurement unit		

Script [1]	Min	Defines the lower scale end of the pointer instruments
	Max	Defines the upper scale end of the pointer instruments
	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values
	Decimal places	Number of decimal places that are used by all of the server display elements
	Setpoint	Output value in the measurement unit

[1] *Although the setpoint is not issued via one of the interfaces it can be used by a script. A possible application is to specify the setpoint via the server-PLC and to process it further via a script on the device.*

Mode

Specifies how the interface output channel should be handled by the evaluation elements of the server

Analogue/digital	Analogue or digital measurement value. This means that the system deals with each measurement value independently of the measurement values before or afterwards.
Cnt.Day	Day counter. This means that the system anticipates that the measurement value of the channel will continuously increase and is reset once per day.
Cnt.Intrvl.	Interval counter. This means that the system assumes that the counter reading is reset each time a measurement is recorded.
Cnt.Inf.	Infinite counter. This means that the system expects that the measurement value of the channel will continuously increase and never be reset.

9.2.1.9.2 Config

> *Note: The following parameters are not available for an interface output channel that is connected to an interface for which "Script parsing" was activated (see "Basis" on page 80).*

9.2.1.9.2.1 Com1 and Com3 (modbus master, RS485)

Slave add.
Address of the modbus slave

Modbus add.
Address of the register that should be written to

Format

Data type

Digital	A digital value must be written.			
	Function	Write single coils (FC 05)		
		Write multiple coils (FC 15)		
Signed	A signed integer value should to be written.			
	Bit	16	16-bit integer	
			Function	Write multiple registers (FC 16)
				Write single register (FC 06)
		32	32-bit integer. Two registers must be written in the modbus slave for this purpose.	
			Function	Write multiple registers (FC 16)
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
Unsigned	The integer value to be written is not signed.			
	Bit	16	16-bit integer	
			Function	Write multiple registers (FC 16)
				Write single register (FC 06)
		32	32-bit integer. Two registers must be written in the modbus slave for this purpose.	
			Function	Write multiple registers (FC 16)
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
Float	A 32-bit float should to be written.			
	Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address	
		LO-HI	LO word on the lower register address and the HI word on the higher register address	
	Function	Write multiple registers (FC 16)		

9.2.1.9.2.2 Com1 and Com3 (modbus slave, RS485)

The following table details the possible access functions depending on the data type of the interface output channel:

Modbus add.	Data type	Read function	Write function
0x0800 : 0x083F	Digital	Read discrete inputs (FC 02)	---
0x0800 : 0x087F	Signed 16/32 bit Unsigned 16/32 bit Float	Read input registers (FC 04)	---

Modbus add.
   Address of the register that should be written to

Format

Data type

Digital	A digital value must be written.			
Signed	A signed integer value should to be written.			
	Bit	16	16-bit integer	
		32	32-bit integer. Two registers must be written in the modbus slave for this purpose.	
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
Unsigned	The integer value to be written is not signed.			
	Bit	16	16-bit integer	
		32	32-bit integer. Two registers must be written in the modbus slave for this purpose.	
			Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address
				LO-HI	LO word on the lower register address and the HI word on the higher register address
Float	A 32-bit float should to be written.			
	Word order	HI-LO	HI word on the lower register address and the LO word on the higher register address	
		LO-HI	LO word on the lower register address and the HI word on the higher register address	

9.2.1.9.2.3 Com2 (serial, RS232)

Column

The ASCII output string is split into individual measurement values by the "number delimiter" (see "Interfaces" on page 80). The "column"configuration parameter specifies where the output value is placed in the ASCII output string.

9.2.1.9.2.4 Script

The configuration is not available for "script" mode.

9.2.1.9.3 Scale

Note: The following parameters are not available for an interface output channel that is connected to an interface for which "Script parsing" was activated (see "Basis" on page 80).

9.2.1.9.3.1 Com1 and Com3 (modbus, RS485)

Scale	off	--	
	on	0% Modbus	Start of the output range in the unit of the modbus slave
		100% Modbus	End of the output range in the unit of the modbus slave
		0%	Start of the output range in the measurement unit
		100%	End of the output range in the measurement unit

Note: *Calculation of the output value during active scaling:*

Basis > min	4mA	D
Basis > max	20mA	C
Min modbus	0	A
Max modbus	65535	B

Basis > setpoint	12mA	F
Scaled output value	32767	E
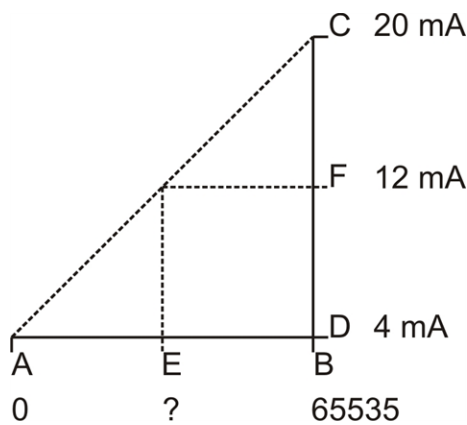
E = ((B - A)/(C - D)) * (F - D) + A

F = ((65535 - 0) /(20mA - 4mA)) * (12mA - 4mA) + 0 =32767

9.2.1.9.3.2 Com2 (serial, RS232)

The scaling is not available for the serial interface.

9.2.1.9.3.3 Script

The scaling is not available for "script" mode.

9.2.1.10 Interface output channels 33-64

Title 33-64
   Freely selectable channel title for the interface output channels

The significance of the remaining configuration parameters in this configuration section corresponds to the "Interface output channels 1-32" configuration section (see "Interface output channels 1-32" on page 118).

9.2.1.11 Calculated channels

   Note: *The values of the calculated channels are directly calculated for every data output (display on the myDatanet server or download from the myDatanet server). They are not saved in the server database.*

9.2.1.11.1 Basis

Title 1-5
   Freely selectable channel title for the calculated channels [0-16 characters]

Mode

Possible calculation modes for the calculated channels

Off	---	*Calculated channel deactivated*	
Table	*Min*	*Defines the lower scale end of the pointer instruments*	
	Max	*Defines the upper scale end of the pointer instruments*	
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values*	
	Decimal places	*Number of decimal places that are used by all of the server display elements*	
Digital	*Invert*	*Inverts the input signal*	
+, -, x, /	*Min*	*Defines the lower scale end of the pointer instruments*	
	Max	*Defines the upper scale end of the pointer instruments*	
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values*	
	Decimal places	*Number of decimal places that are used by all of the server display elements*	
Delta	*Determines the difference between two measurement values and divides the result by the time difference between the time stamps of the measurement values. The time unit (value/sec., value/min., ...) for the result can be selected via the "Time basis" parameter located in the "Calculation" tab. It is thus possible to convert the counter reading (m^3) of the source in to a flow rate (m^3/min.) (see "Additional explanation: Delta mode" on page 127).*		
	Min	*Defines the lower scale end of the pointer instruments*	
	Max	*Defines the upper scale end of the pointer instruments*	
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values*	
	Decimal places	*Number of decimal places that are used by all of the server display elements*	
clone	*Creates a clone of a measurement channel. This ensures it is possible to rename channels, to select different ends of the scale for the pointer instruments, to determine a new string as a measurement unit and to adjust the number of decimal places. The measurement values (numerical value without unit) correspond exactly to those of the source.*		
	Min	*Defines the lower scale end of the pointer instruments*	
	Max	*Defines the upper scale end of the pointer instruments*	
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]. It has no direct influence on the values*	
	Decimal places	*Number of decimal places that are used by all of the server display elements*	

▼

Shift element down

▲

Shift element up

9.2.1.11.2 Calculation

Off	---	Calculated channel deactivated
Table	Source	Selection of the channel from which the input data is used
	✎	Opens the screen for entering the values table (the table rows are interpolated linearly, values outside of the defined table are extrapolated linearly.)
Digital	Source	Selection of the channel from which the input data is used
	High level	Signal recognition level
+, -, x, /	Source	Selection of the channel from which the input data is used
	+, -, x, /	
	Source	Selection of the channel from which the input data is used
Delta	Source	Selection of the channel from which the input data is used
	Time basis	Specifies the desired time unit (value/sec., value/min., ...) for the result
	Offset	Offset that is added following multiplication with the "Factor" parameter. The result is converted in to the desired time unit before multiplication with the "Factor" parameter.
	Factor	Factor with which the result is multiplied once it has been converted into the desired time unit. The "Offset" parameter is then added.
Clone	Source	Selection of the channel that should be cloned

Note:

Additional explanation: Delta mode

Assumption: *The source channel contains the counter reading of an infinite counter in m^3. The calculated channel 1 should contain the flow rate in m^3/s and calculated channel 2 should contain the flow rate in l/h.*

Required configuration

Parameter	Value channel 1	Value channel 2
Basis -> mode	Delta	Delta
Basis -> unit	m^3/s	l/h
Calculation -> time basis	Seconds	Hours
Calculation -> offset	0	0
Calculation -> factor	1	1000

Source		Calculated channel 1	Calculated channel 2
Date/time	Infinite counter [m^3]	Flow rate [m^3/s]	Flow rate [l/h]
26.03.2013 12:50	900	0 [1]	0 [1]
26.03.2013 12:51	960	1	3,600,000
26.03.2013 12:52	990	0.5	1,800,000
26.03.2013 12:53	1005	0.25	900,000
26.03.2013 12:54	1065	1	3,600,000

[1]*Calculation not possible as there is no measurement value before 12:50.*

Explanation: *No values can be determined for the measurement at 12:50 for the calculated channels as there is no previous value and the difference between the counter readings cannot be determined. For the measurement at 12:51, the difference to the counter reading is 60m^3 and the time difference is 60sec.*

Result = { (value difference / time difference [sec.]) * time basis [sec] * factor } + offset

The result for calculated channel 1 (time basis "Seconds", offset "0" and factor "1") is calculated as follows:

Channel 1 = { (60m^3 / 60sec.) * 1 * 1 } + 0 = 1m^3/s

The result for calculated channel 2 (time basis "Hours", offset "0" and factor "1000") is calculated as follows:

Channel 2 = { (60m^3 / 60sec.) * 3600 * 1000 } + 0 = 3,600,000l/h

9.2.1.11.3 Alarms

Note: *The evaluation of the alarm thresholds for calculated channels can only occur if the device has transferred the measurement data to the myDatanet server.*

Alarm low	An alarm is triggered, if the measurement value drops to or below this value.
Alarm high	An alarm is triggered, if the measurement value meets or exceeds this value.
Hyst %	Hysteresis for the all-clear in the event of an alarm/warning (e.g. Hyst=5%, alarm or warning at 100 -> all-clear at 95) or reset the trigger (e.g. Hyst=5%, level = greater or equal, trigger at 100 -> reset at 95)

9.2.1.12 Output channels

9.2.1.12.1 Basis

Ext warmup time
Indicates the amount of time that an output channel is switched on in "Ext warmup time" mode before the measurement

mA OUT 1-2 title
Freely selectable channel title for the analogue outputs (not galvanically isolated) [0-16 characters]

Relay 1-6 title
Freely selectable channel title for the relays [0-16 characters] (every 3 relays with a joint root)

Mode
Basic settings for the analogue outputs

Off	*---*	*Output channel deactivated*
0-20 mA	*0%*	*Start of the output range in the measurement unit*
	100%	*End of the output range in the measurement unit*
	Setpoint	*Output value in the measurement unit*
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]*
	Decimal places	*Number of decimal places that are used by all of the server display elements*
4-20 mA	*0%*	*Start of the output range in the measurement unit*
	100%	*End of the output range in the measurement unit*
	Setpoint	*Output value in the measurement unit*
	Unit	*String that is used as a measurement unit by all of the server display elements [0-16 characters]*
	Decimal places	*Number of decimal places that are used by all of the server display elements*

Basic settings for the relays

Off	*---*	*Output channel deactivated*
Ext warmup time		*The output channel is switched on "Ext warmup time" seconds prior to the measurement. If the value is "0", the output channel is not switched on.*
Digital	*Invert*	*Inverts the level issued on the device(see "Supplementary explanation regarding "Digital" mode" on page 129)*
	Setpoint	*Setpoint (on/off) that should be issued (see "Supplementary explanation regarding "Digital" mode" on page 129)*
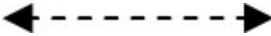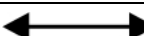
Note:

Supplementary explanation regarding "Digital" mode

Invert	Setpoint		Output on the device
Off	Off	=	Off (low)
Off	On	=	On (high)
On	Off	=	On (high)
On	On	=	Off (low)

Note:

Example to explain the burst interval in conjunction with the ext. warmup time (ext. warmup time < measurement cycle):

Basic setting		Record interval	5min.
		Burst interval	60 sec.
		Measurement cycle	15 sec.
Output channels		Ext warmup time	5 sec.
Measurement channels ->Config.		Decay	med
		Time	60 sec.
Output on the device		Sensor supply	
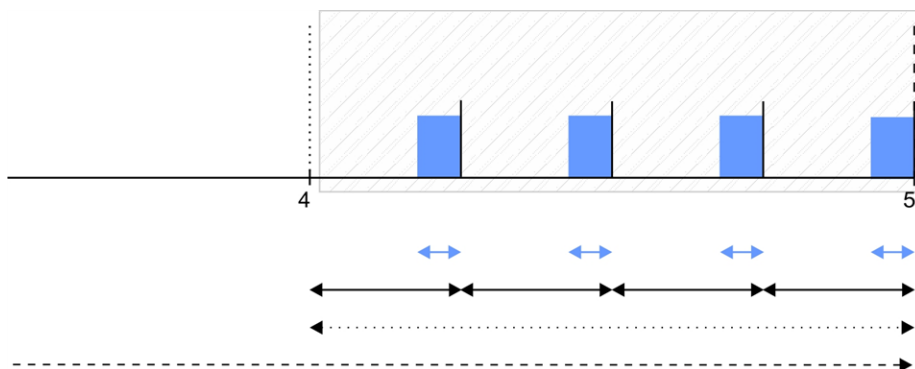
Formation of the measurement value: *In each case, the sensor supply is activated 5 sec before expiry of the measurement cycle. This results in four valid measurements that are used to determine a median and are recorded as the measurement value.*

Note:

Example to explain the burst interval in conjunction with the ext. warmup time (ext. warmup time > measurement cycle):

Basic setting		Record interval	5min.
		Burst interval	75 sec.
		Measurement cycle	15 sec.
Output channels		Ext warmup time	30 sec.
Measurement channels ->Config.		Decay	med
		Time	60 sec.
Output on the device		Sensor supply	
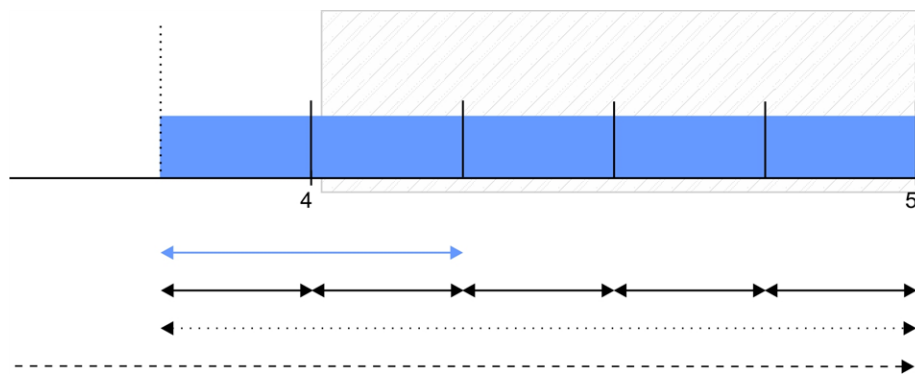
Explanation: *As the ext. warmup time is greater than the measurement cycle in this example, the sensor supply is activated right at the start of the burst interval and is only deactivated again when the recording starts. The first measurement is completed 15 sec. (measurement cycle) after the burst interval has started. At this time, the sensor supply had not yet been active for the duration of the ext. warmup time. This means that the measurement value of the connected sensor is probably not stable yet and the measurement is thus invalid. In this example, only the second measurement after the start of the burst interval is valid. To exclude the invalid measurements during the formation of the measurement value, the selected decay period must be appropriately smaller than the burst interval. The burst interval has to be increased from 60 sec. to 75 sec. to receive four valid measurements in exactly the same way as in the previous example (ext warmup time < measurement cycle).*

Formation of the measurement value: *The sensor supply is activated at the start of the burst interval. The first measurement is completed 15 sec. later. Five measurements are thus generated. The first measurement is invalid (ext. warmup time has not expired yet). As the decay period only lasts 60 sec., only the second to fifth measurement are used to determine the median and are recorded as the measurement value. The first invalid measurement is thus ignored.*

Note:

Example to explain the filter time in conjunction with the ext. warmup time

Basic setting		Record interval	5min.
		Measurement cycle	1min.
Output channels		Ext warmup time	1sec.
Measurement channels ->Config.		Filter time	500ms
Output on the device		Sensor supply	
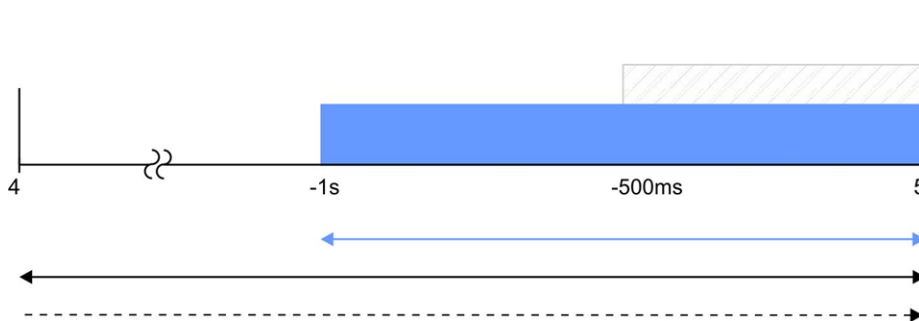
Explanation: *The sensor supply is always activated 1 sec before expiry of the measurement cycle. The filter time starts 500ms before expiry of the measurement cycle, which ensures that a valid value is available at the time of a measurement. However, this also means that the filter time must be taken into consideration when selecting the ext. warmup time. In the current example, the sensor connected to the input must at the latest supply stable values 500ms following activation of the sensor supply. Otherwise, invalid values would be taken into consideration while averaging during the filter time resulting in the measurement value being distorted.*

9.2.1.13 Internal channels

9.2.1.13.1 Basis

GSM level title	Freely selectable channel title for the GSM field level[0-16 characters]	
	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]
Voltage title	Freely selectable channel title for the supply voltage [0-16 characters]	
	Unit	String that is used as a measurement unit by all of the server display elements [0-16 characters]

9.2.1.13.2 Alarms

Warning	Value low	A warning is triggered, if the measurement value drops to or below this value.
	Value high	A warning is triggered, if the measurement value meets or exceeds this value.
Alarm	Value low	An alarm is triggered, if the measurement value drops to or below this value.
	Value high	An alarm is triggered, if the measurement value meets or exceeds this value.
Hyst %	Hysteresis for the all-clear in the event of an alarm/warning (e.g. Hyst=5%, alarm or warning at 100 -> all-clear at 95) or reset the trigger (e.g. Hyst=5%, level = greater or equal, trigger at 100 -> reset at 95)	

9.2.1.13.3 Trigger

The following two type of triggers are differentiated:

- Event trigger (XM)

  The relevant operation (e.g. initiate transmission) is only executed once when the trigger event occurs.

- Level trigger (QU, SL, RO, RF, ON, I1-I4)

  The relevant operation (e.g. activate online mode) is executed as long as the trigger is active.

Event trigger	XM	Initiate transmission	
	Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
		Greater or equal	The trigger is initiated, if the measurement value meets or exceeds this value.
		Less or equal	The trigger is initiated, if the measurement value drops to or below this value.
Level trigger	QU	Fastrecording (record interval = record interval / factor)	
	SL	Slow recording (record interval = record interval * factor)	
	RO	Switch on recording	
	RF	Switch off recording	
	ON	Activate online mode	
	I1	Internal use with the control program	
	I2	Internal use with the control program	
	I3	Internal use with the control program	
	I4	Internal use with the control program	
	Level	Levels for initiating the trigger. The hysteresis from the "Alarm" tab is used to determine the level to reset the trigger.	
		Greater or equal	The measurement value must be greater or equal to the level to initiate the trigger.
		Less or equal	The measurement value must be less or equal to the level to initiate the trigger.

9.2.1.14 Alarm settings

Acknowledgement	Standard	The global server setting is used to determine whether alarms must be acknowledged automatically or manually (see "myDatanet Server Manual " 805002).
	automatic	Alarms are acknowledged automatically as soon as all of the messages have been sent. If SMS that have a tariff with a delivery confirmation function have also been sent, acknowledgement is provided after delivery confirmation.
	manual	Alarms must be acknowledged by the user.
Offline alarm	Alarm in case the device does not respond for longer than the set number of transmission cycles. An additional tolerance of 10 min. is granted per transmission cycle to take any retries when establishing a connection into consideration. Example: Transmission cycle: 60 min; 3x transmission cycles -> Alarm after > 03:30	
Transfer volume	Standard	The setting for the transfer volume alarm is taken from the global server settings (see "myDatanet Server Manual " 805002).
	Off	The transfer volume alarm is deactivated.
	individual	The level at which the transfer volume alarm should be triggered can be entered in the adjacent field in KiB.
On alarm	A	The alarm is recorded in the alarm list.
	Ü	An immediate transmission is initiated.
On warning	A	The warning is recorded in the alarm list.
	Ü	An immediate transmission is initiated.
On fault alarm [1]	A	The fault alarm is recorded in the alarm list.
	Ü	An immediate transmission is initiated.
On fault warning [2]	A	The fault warning is recorded in the alarm list.
	Ü	An immediate transmission is initiated.

[1] A "fault alarm" is triggered if a measurement channel is configured in such a way that one of the error values NAN, OL (open loop), UF (underflow), OF (overflow) or SC (short circuit) is issued if the measurement range is breached and maintaining the last measurement value is not activated or the maximum number for holding the last valid measurement value has been reached.

[2] A "fault warning" is triggered if a measurement channel is configured in such a way that one of the error values NAN, OL (open loop), UF (underflow), OF (overflow) or SC (short circuit) is issued if the measurement range is breached and the maximum number for holding the last valid measurement value has not yet been reached.

9.2.1.15 Basic setting

Connection type	Interval	The device connects in the transmission cycle.
	Interval & wakeup	The device connects to the server in the transmission cycle and can be placed into ALOHA transmission mode via the server.
	online	The device does not disconnect the connection and continuously transmits the measurement data. However, every 7 days, the connection is temporarily interrupted in order to verify the server assignment.
Communication channel	GPRS only	Only the GPRS interface is used for establishing a connection to the myDatanet server. A retry is completed in the event of an error (see "Procedure in case of connection aborts" on page 43).
	GPRS and LAN (GPRS preferred)	An attempt is made to establish a GPRS connection. A LAN connection is established if this does not work.
	GPRS and LAN (LAN preferred)	Initially, an attempt is made to establish a LAN connection. A GPRS connection is established in the event of an error.
Online at backup connection	If the "online" connection type and one of the two "GPRS and LAN" modes were selected as the communication channel, an online connection is established via the backup channel if the online connection via the preferred channel is terminated. If this checkbox was not selected, the data is synchronised via the backup channel in the transmission cycle if the online connection via the preferred channel is terminated.	
Aloha/Wakeup duration	Duration of the Aloha/wakeup connection	
Transmission cycle	Time between transmissions	
Record interval	Time between measurement data recordings	
Trigger measurement on wakeup	A measurement is also started if the device is placed in ALOHA transmission mode by the server.	
Measure quick divisor	Record interval = record interval / factor (from triggering)	
Measure slow factor	Record interval = record interval * factor (from triggering)	
Measurement cycle	Time between measurements (00:00 same as record interval)	
Burst interval	Period of time, during which measurement intervals are used for measuring, before recording begins (00:00 measurement cycle is permanently activated)	
Time zone	Regional settings (not relevant for raw measurement data as this is stored in UTC)	

Delay for power off alarm	A relevant alarm is only triggered if the supply voltage has failed for longer than the time configured here. This prevents alarms from being triggered in the event of brief supply voltage failures. However, an entry is always added to the device log.	
Daylight saving time	Configuration for automatic time adjustment	
	Standard	The configuration for the time adjustment is adopted by the global server setting (see "myDatanet Server Manual " 805002).
	Off	Automatic time adjustment deactivated
	USA	Predefined setting for the American area
	EU	Predefined setting for the European area
Position cycle	Position update interval (00:00 positioning is completed with every connection)	
Default report	Selection of the report that is loaded by clicking on the device link in the maps	
	Off	The default graphic is loaded.
	"Name of a report"	The selected report is loaded.
Report template	Selection of whether the default graphic or a report template to display the data is used when clicking on the symbol to display the measurement data located in the site/application list. Only the report templates in which the site/application type of the first wild card is compatible with the site/application that is currently being edited are displayed in the dropdown list.	
	(not assigned)	The default graphic is used to display the measurement data.
	"Name of a report template"	Name of the report template used to display the measurement data

Example to clarify the record interval, measurement cycle and burst interval in conjunction with the decay

As only the universal inputs have a decay module (see "Internal processing of the measurement values" on page 31), it is only wise to use the burst interval for these measurement channels. An explanation of the sequences, if an ext. warmup time also has to be used, is provided in chapter "Output channels" on page 128.

Basic setting	◄ – – – – – ►	Record interval	5min.
	◄ ·············· ►	Burst interval	1min.
	◄►	Measurement cycle	15 sec.
Measurement channels ->Config.		Decay	med
	▨▨▨▨	Time	60 sec.
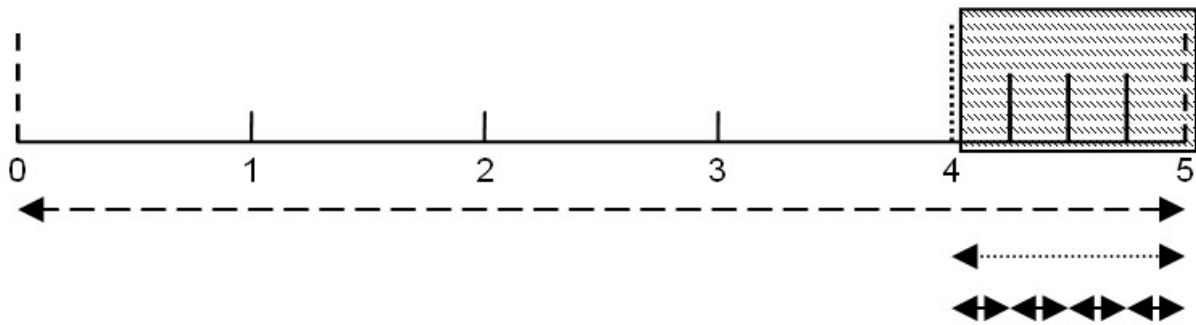
Formation of the measurement value: The last four measurement values are used to determine the median and recorded as the measurement value.

Note: *Additional explanation about the connection types*

Connection type	Energy consumption	Data volumes	Response time
online	🔋🔋🔋🔋	💾💾💾💾	⧖
Interval & wakeup	🔋🔋🔋🔋	💾	⧖ ⧖
Interval	🔋🔋🔋🔋	💾	⧖ ⧖ ⧖

Note: Additional explanation about the effects of the measurement cycle and transmission cycle on the monthly data volume

Measurement cycle/transmission cycle	Data volume per month
Only the 8 universal inputs are active (only minor change to the measurement signal)	
2min/10min	*13,0MB*
2min/2h	*2,0MB*
1min/2h	*2,5MB*
1min/4h	*2,0MB*
5min/2h	*1,2MB*
5min/online	*4,0MB*
Only the 64 interface channels are active (values change significantly (random values))	
1min/4h	*18,6MB*
1min/2h	*19,0MB*

9.2.1.16 LAN settings

Note: This configuration section is only visible, if one of the modes that the LAN interface uses is active as the communication channel in the "Basic settings" configuration section (see "Basic setting" on page 134).

IP configuration	DHCP	There is a DHCP server in the network and the myDatalogMUC is automatically assigned the settings for the IP address, subnet mask and gateway.
	Static	The settings for the IP address, subnet mask and gateway must be entered manually.
IP address	If DHCP is active, the IP address received from the DHCP server is displayed. Otherwise the IP address must be entered manually.	
Subnet mask	If DHCP is active, the subnet mask received from the DHCP server is displayed. Otherwise the subnet mask must be entered manually.	
Gateway	If DHCP is active, the IP address of the gateway received from the DHCP server is displayed. Otherwise the IP address of the gateway must be entered manually.	

9.2.1.17 FTP export settings

> *Note: This configuration section is only visible if the "FTP Agent Extended" licence for the myDatanet server has been enabled.*

FTP export profile	off	FTP export deactivated
	"Name of an FTP export profile"	List with the FTP export profiles that were created on the myDatanet server (for creating an FTP export profile, see "myDatanet Server Manual " 805002).
Settings of the selected profile	Shows an overview of the most important parameters of the selected FTP export profile	
FTP directory	Makes overwriting the standard directory of the selected FTP export profile possible [0-100 characters]	
Last export	Time stamp of the last FTP export	

9.2.2 Device configuration

> *Note: Several of the configuration fields in the following sub chapters may possibly be hidden depending on the respective user level. In this case, contact the myDatanet server administrator.*

You can reach the input screen for configuring the device by clicking on the serial number in the site list (see "myDatanet Server Manual " 805002) or by clicking on the device name in the device name list (see "myDatanet Server Manual " 805002).

9.2.2.1 Comments

Comments
> *Free comment field (is also displayed below the site name in the measurement instrument list)*

9.2.2.2 Measurement instrument

Customer	Name of the customer to whom the measurement instrument is assigned
Tags	List of the tags that are already assigned to the measurement instrument. This assignment can be cancelled by clicking on the cross next to the title of the tag. The input screen for assigning the tags is opened by clicking on the plus symbol. This enables existing tags to be assigned and new tags to be created.
Serial number	Serial number of the instrument
Instrument class	The instrument class of the site and instrument must match for an instrument to be able to be connected to a site. Once the instrument has been created via the server interface, the instrument class can only be changed up until the first connection of the instrument to the server. If an instrument class, that does not match the actual class of the instrument, is selected when the instrument is created it is automatically corrected during the first connection.
Telephone number	Telephone number of the SIM card. The control SMS messages (e.g. wakeup) are sent to this number. Format: +43555837465
Instrument flags	Additional information regarding the instrument class (for internal use)

Firmware version	Current software version installed on the measurement controller	
Modem version	Current software version installed on the modem controller	
OS version	OS version of the modem	
Last connection	In each case, the last time stamp of the affected operation	
Last wakeup		
Last disconnection		
Last transmission error		
Last Aloha connection		
Wakeup SMS count	Number of wakeup SMS sent to this device since the last connection. This counter is reset at/during each successfully established connection.	
Firmware update	Off	Firmware update is deactivated.
	On	As soon as a new version of the selected firmware type is available, this is installed immediately.
	Even if tag is missing	Firmware is also transferred to the device if the device has not transmitted the current firmware version to the server (NOT RECOMMENDED!).
	Allow downgrade	Facilitates the installation of an older firmware version than the one on the device (NOT RECOMMENDED!)
	Once	Performs a single firmware update. If no new firmware is available or the firmware was installed successfully, the firmware update is automatically switched to "OFF".
	Ignore	The firmware update is deactivated and no information is provided about available firmware updates.
Firmware type	Released	Only firmware versions that have successfully undergone internal and field testing are installed (this practically eliminates malfunctions).
	Release candidate	Only firmware versions that have successfully undergone internal testing are installed (malfunctions cannot be excluded).
	Beta release	Even firmware versions that have not successfully undergone all of the internal tests are installed (malfunctions may occur).
Hardware version	Hardware version of the myDatalogMUC	

9.2.2.3 Device-specific settings

Operating type	hold	Measurement: OFF, transmission: ON
	run	Measurement: ON, transmission: ON
	transport	Measurement: OFF, transmission: OFF
	offline	Measurement: ON, transmission: OFF

9.2.2.4 GPRS

SIM tariff
Selected SIM tariff

Chapter 10 myDatanet server

Note: All of the screenshots show version 44.1 of the myDatanet server using the standard colour scheme. Newer versions may include minor changes to the appearance of the server.

10.1 Overview
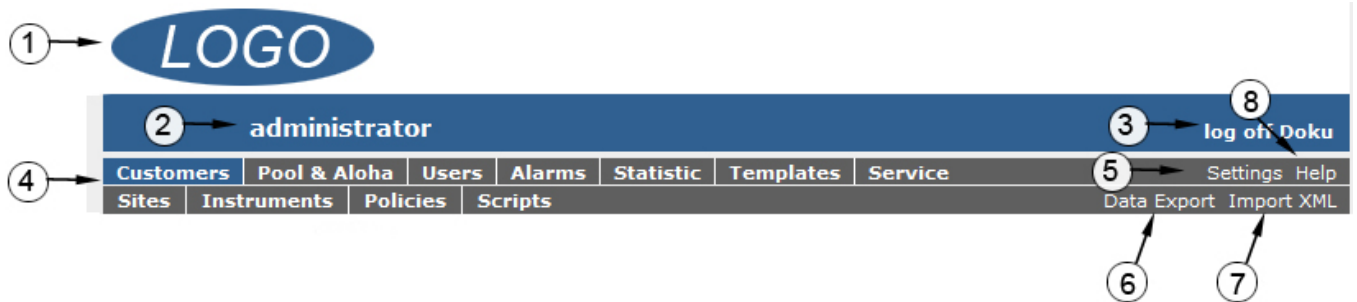
Overview of the myDatanet server

1	Freely selectable logo	**5**	Opens the screen to input the global settings for the server
2	User group to which the user who is currently logged in is assigned	**6**	Switches to the "Data exports" area to configure the data export. This tab is only visible if at least the licence for one export variant is available.
3	Logout the active user	**7**	Opens the input screen to upload a XML file. This tab is only visible if the licence for the XML import is available.
4	Tabs to switch between the individual server areas	**8**	Call up the help menu

10.1.1 Explanation of the symbols

Adds a new entry to the current list (reports, sites, users, etc.).

Deletes the adjacent element (reports, sites, users, etc.) from the list.

Calls up the input screen to edit the adjacent element (reports, sites, users, etc.).

10.2 "Customer" area
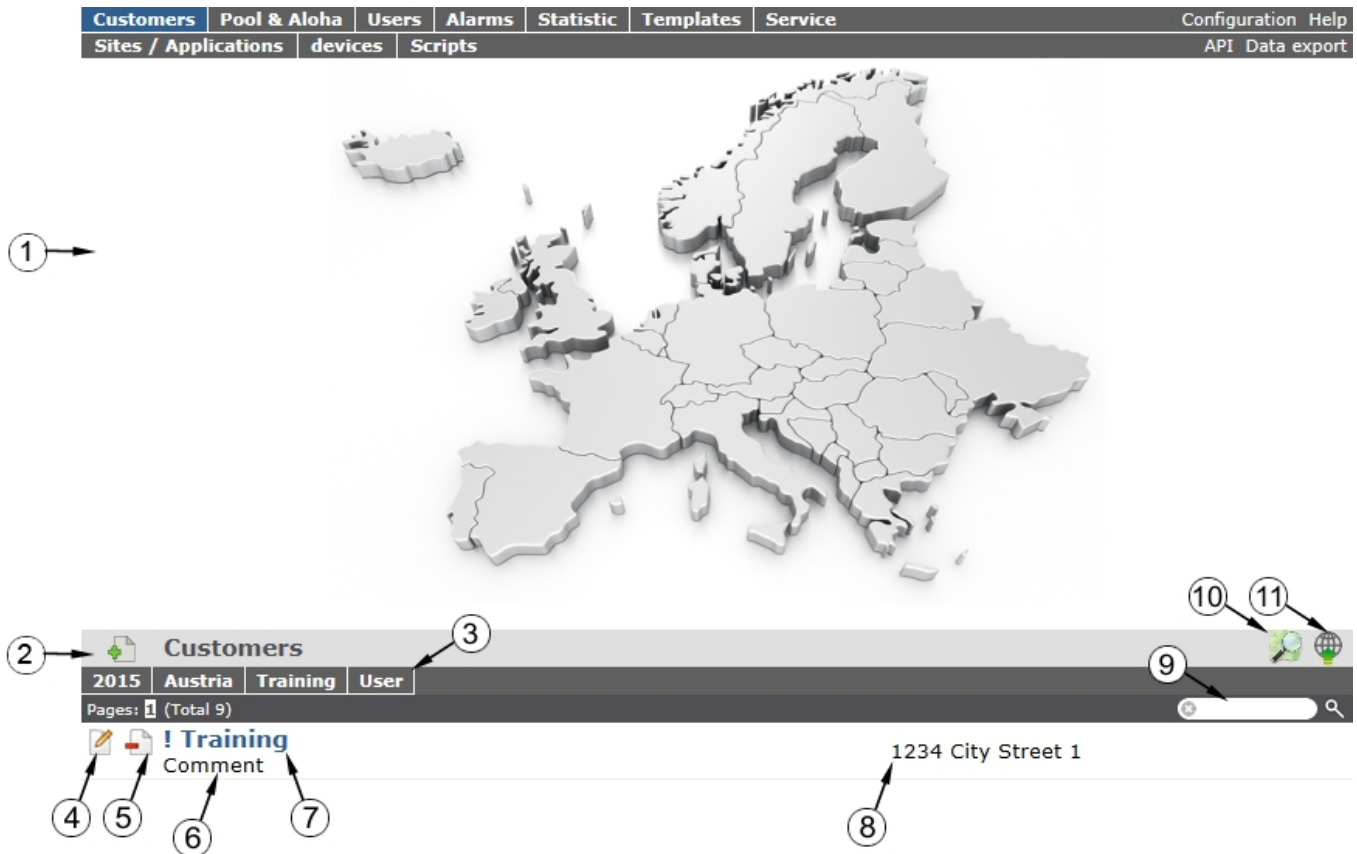
Overview of the "Customer" area

1	Area where an image file can be displayed as a "Map" and/or the OpenStreetMaps map can be displayed The sites can be manually placed on the image file used as a "map". In the OpenStreetMaps map, the sites are only displayed once GPS coordinates have been assigned to the site.
2	Adds a new customer
3	List of tags that are assigned to at least one of the customers displayed in the list of customers. If the list of customers was limited by the search field or selection of a tag, this is taken into consideration when creating the list of tags. A cross is added to the end of the list of tags as soon as the list of customers is limited by the selection of a tag. Clicking on this cross will reset the selection of all tags and the restriction is cancelled. By clicking on one of the tags with the left mouse button only those customers who have been assigned the corresponding tag are displayed in the list of customers and the selected tag is highlighted in colour. By clicking on one of the tags with the right mouse button all of the customers who have been assigned the corresponding tag are hidden, the selected tag is highlighted in colour and the title of the tag is crossed out. Clicking the same mouse button again will remove the restriction.

4	Opens the input screen for configuring the customer
5	Deletes the customer
6	Comment that can be entered in the configuration of the customer
7	If a default report was defined, the default report is accessed by clicking on the name of the customer. Otherwise the "Sites" area at customer level is opened by clicking on the name of the customer (see ""Site" area at customer level" on page 144 or "Reports" on page 145).
8	Customer's address that can be entered via the input screen for configuring the customer
	Search field to filter the customer list
10	Symbol via which an image file can be loaded on to the server as an "Overview map" To remove the "Map" again, open the upload dialogue again and click on "Submit" without selecting an image file beforehand.
11	Symbol via which a OpenStreetMaps map, on which the sites are displayed, can be loaded. (see "Map view" on page 145)

10.3 "Site" area at customer level

Overview of the "Sites" area at customer level

1	Area where an image file can be displayed as a "Map" and/or the OpenStreetMaps map can be displayed

The sites can be manually placed on the image file used as a "map".

In the OpenStreetMaps map, the sites are only displayed once GPS coordinates have been assigned to the site. |
2	List of reports (see "Reports" on page 145)
3	List of sites/applications (see "Site" on page 79)
4	List of groups (see "Groups" on page 145)
5	Symbol that represents a site on the "Map"
6	Symbol via which a OpenStreetMaps map, on which the sites are displayed, can be loaded. (see "Map view" on page 145)

7	Symbol via which an image file can be loaded on to the server as a "Map"
	To remove the "Map" again, open the upload dialogue again and click on "Submit" without selecting an image file beforehand.

10.3.1 Reports

The reports provide a variety of options to display graphs of the data on the web interface of the myDatanetserver or to download the data from the myDatanet server. Detailed instructions on creating and handling the reports is provided in myDatanet Server Manual (805002).

10.3.2 Groups

The groups offer a variety of options for consolidating and summarising the measurement data of different sites. Detailed instructions on creating and handling the groups is provided in myDatanet Server Manual (805002).

10.3.3 Map view

The map view provides an overview of the geographic position of the sites. Detailed instructions on operating and configuring map view are provided in myDatanet Server Manual (805002).

10.4 Recommended procedure

10.4.1 Creating the site

> *Note: Some of the fields mentioned in the following chapters may be hidden depending on the respective user level. In this case, please contact the administrator of the myDatanet server.*

Detailed instructions on creating a new site are provided in myDatanet Server Manual (805002).

1. Log in via the web interface on the myDatanet server. You will receive the web address from your responsible sales partner.

Log in with your user name and password:

user name	
password	

Log in

Login form of the myDatanet server

2. Click on the "Customer" menu item of the myDatanet server to call up the list of available customers. Select an existing customer or create a new customer.
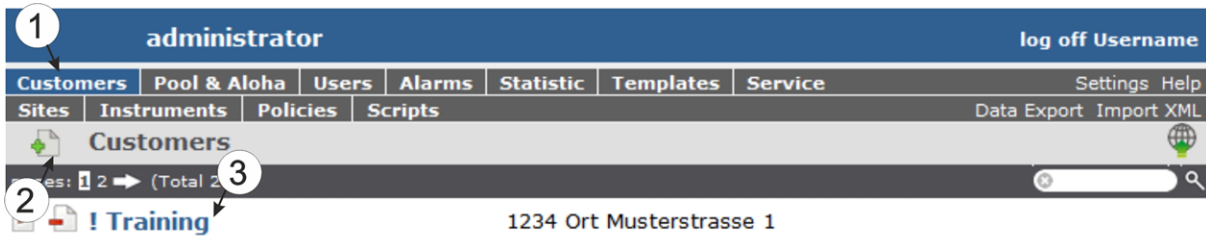
Selecting the customer

1	Menu item to call up the list of customers	3	List of available customers
2	Creating a new customer		

3. Click on the "Sites / Applications" menu item of the myDatanetserver to call up the list of available application templates and sites. Create a new "myDatalogMUC " type site or a site using an application template that is compatible with the "myDatalogMUC " site type.
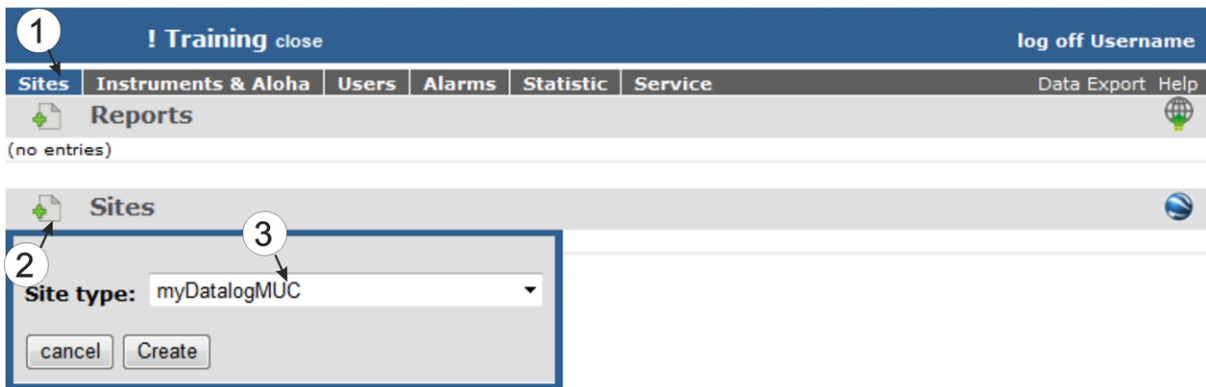
Creating the site

1	Menu item to call up the list of sites	3	Selection of the type of site
2	Creating a new site		

4. Link the site/application with the myDatalogMUC by selecting the serial number. If the serial number of your device is not included in the list, you must first assign the device to the customer (see "Assigning a device to the customer" on page 147) or the device is already assigned to a different site. More detailed information on this is provided in myDatanet Server Manual (805002).

Linking the device and site

1	Device assignment	2	List of devices that have not yet been linked with a site

10.4.2 Assigning a device to the customer

Note: Some of the fields mentioned in the following chapters may be hidden depending on the respective user level. In this case, please contact the administrator of the myDatanet server.

Detailed instructions on shifting the devices are provided in myDatanet Server Manual (805002).

1. Log in via the web interface on the myDatanet server. You will receive the web address from your responsible sales partner.

Login form of the myDatanet server

2. Click on the "Pool and Aloha" menu item of the myDatanet server to access the list of devices that are in ALOHA transmission mode.
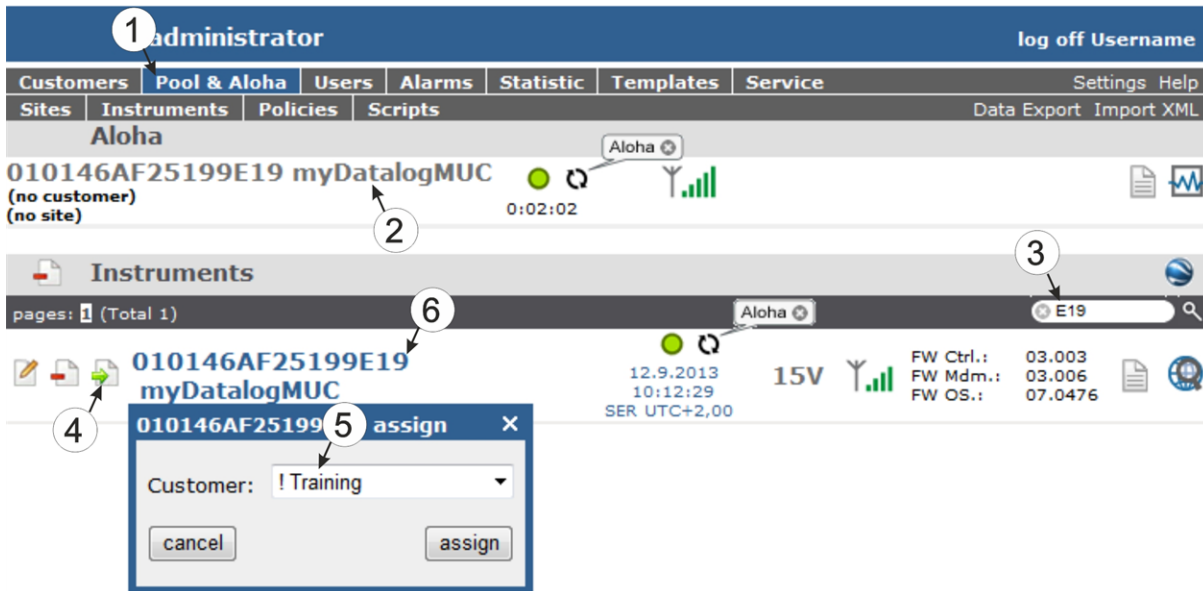
Assigning a device to the customer

1	Menu item to call up the list of devices that are in ALOHA transmission mode	**4**	Opens the dialogue to select the customer to whom the device should be assigned
2	List of devices in ALOHA transmission mode	**5**	Selection of the customer to whom the device should be assigned
3	Filter for the list of measurement instruments	**6**	List of measurement instruments

3. Initiate ALOHA transmission mode on the device. Instructions on this are provided in "ALOHA transmission mode" on page 44.

4. Use the filter to find the required device in the list of measurement instruments. Click on the symbol to open the dialogue to select the customer (see "Assigning a device to the customer" on page 148) and assign the device to the selected customer.

Chapter 11 Pawn script

11.1 General

The following chapter describes the functionality of the pawn script. PAWN (previously SMALL) is a C-similar scripting language that is used on embedded systems.

Additional and more detailed information is provided on the developer's website:
http://www.compuphase.com/pawn/pawn.htm

There are two ways to create a pawn script for the myDatalogMUC :

- Direct entry in the "Script" input field in the "Control" configuration section
- Use of a script template that was created on the myDatanet server

11.1.1 Direct input of a pawn script

The pawn script is entered via the "Control" configuration section (see "Control" on page 79) of the input screen for configuring the site. "Pawn" must be selected as the "script type" so that myDatalogMUC interprets the commands entered in the "script" as a pawn script.

11.1.2 Uploading a binary file

If the "Upload a compiled script" entry was selected via the "Script source" list selection in the "Control" configuration section (see "Control" on page 79) of the input screen for configuring the site, a binary file that was previously created can be uploaded to the myDatanet server. This is then loaded into the myDatalogMUC during the next connection. When using this method, "Pawn" must also be selected as the "script type" so that the myDatalogMUC interprets the commands as a pawn script.

> *Note: If a pawn script is used to write directly to an output, the setpoint entered via the input screen of the myDatanet server is overwritten with the determined value.*

11.2 Compiler options

Compressing the pawn program code

```
// The parameter is used to specify which of the sections should be
// compressed
// 0: no compression (default)
// 1: DATA
// 2: DATA and CODE
// 3: DATA, CODE and TABLES

#pragma amxcompress <0-3>
```

11.3 API

11.3.1 Constants

Return codes for general purposes

```
OK    = 0
ERROR = -1
```

11.3.2 System

main();

> This function is executed during a PowerOn and when the PAWN script is exchanged. It should include all initialisations that only have to executed once during program start-up.

forward public Mdn_CtrlFinish();

> This function is called up at the time of every measurement once all of the measurement values have been generated. It should comprise all of the calculations and functions that should be performed cyclically.

11.3.3 Date & Time

native Mdn_GetTime(&hour=0, &minute=0, &second=0, timestamp=0);

> If no time stamp was transferred (timestamp=0), the current system time (in local time) is converted to hours/minutes/seconds. Alternatively, the transferred time stamp is converted to hours/minutes/seconds.

Parameter	Explanation
hour	Variable to store the hours - OPTIONAL
minute	Variable to store the minutes - OPTIONAL
second	Variable to store the seconds - OPTIONAL
timestamp	Time stamp that should be converted = 0: The current system time (in local time) is converted. > 0: The transferred time stamp is converted. (The time stamp must be specified in seconds since 31.12.1999.)

	Explanation
Return value	• timestamp = 0: Seconds since 31.12.1999 (current system time in local time) • timestamp > 0: The transferred time stamp is returned.

native Mdn_GetDate(&year=0, &month=0, &day=0, timestamp=0);

If no time stamp was transferred (timestamp=0), the date (year, month, day) is determined for the current system time (in local time). Alternatively, the date (year, month, day) is determined for the transferred time stamp.

Parameter	Explanation
year	Variable to store the year - OPTIONAL **Note:** *The year specified relates to the 21st century, i.e. the value 14 refers to the year 2014.*
month	Variable to store the month - OPTIONAL
day	Variable to store the day - OPTIONAL
timestamp	Time stamp for which the date should be determined = 0: The date for the current system time (in local time) is determined. > 0: The date for the transferred time stamp is determined. (The time stamp must be specified in seconds since 31.12.1999.)

	Explanation
Return value	• *timestamp = 0: Seconds since 31.12.1999 (current system time in local time)* • *timestamp > 0: The transferred time stamp is returned.*

native Mdn_GetTimezoneOffset();

Returns the difference (in seconds) between UTC and local time configured for the site on the myDatanet server. This can be used to determine the UTC in the script by subtracting the difference from the system time (local time). The offset value is determined by the myDatanet server in accordance with the set time zone (including summer/winter time) and is synchronised during every connection to the device.

Example: Central European time (CET = UTC+1) is used for the site -> Offset = 3600 sec.

	Explanation
Return value	Offset value in seconds

native Mdn_DoW(timestamp);

Calculates the weekday from a given timestamp

Parameter	Explanation
timestamp	Timestamp of the day in question

	Explanation
Return value	Weekday, 0=Monday ... 6=Sunday

11.3.4 Encoding

native Mdn_SetPacked(data{}, pos, &{Float,Fixed,_}:value, size=4, bool:bigendian=false);
Writes the transferred value to a specified position in an array

Parameter	Explanation
data	Array that should contain the data
pos	Byte offset within the array to determine the position where the value should be written
value	Value that should be written in the array
size	Number of bytes that should be used for the value to be written
bigendian	Settings for the byte order that should be used when writing the value: true: "Big endian" is used false: "Little endian" is used

	Explanation
Return value	• OK, if successful • ERROR

Note: Additional explanation on the byte order:

In the following example, the whole number 439.041.101 is saved as a 32-bit integer value from memory address 10000.

Addresses	Big endian			Little endian		
	Hex	Dez	Binary	Hex	Dez	Binary
10000	1A	26	00011010	4D	77	01001101
10001	2B	43	00101011	3C	60	00111100
10002	3C	60	00111100	2B	43	00101011
10003	4D	77	01001101	1A	26	00011010

native Mdn_SetPackedB(data{}, pos, const block{}, size);
Writes the transferred data block to the specified position in an array

Parameter	Explanation
data	Array that should contain the data
pos	Byte offset within the array to determine the position where the data block should be written
block	Data block that should be written in the array
size	Number of bytes that should be written in the array by the data block

	Explanation
Return value	• OK, if successful • ERROR

native Mdn_GetPacked(const data{}, pos, &{Float,Fixed,_}:value, size=4, bool:bigendian=false);
Returns the value that is located at the specified position in an array

Parameter	Explanation
data	Array that contains the data
pos	Byte offset within the array to determine the position from which the data should be read
value	Variable to store the data to be read
size	Number of bytes that should be read
bigendian	Specifies how the packed data must be interpreted: true: The data is saved in "Big endian" format in the array. false: The data is saved in "Little endian" format in the array.

	Explanation
Return value	• OK, if successful • ERROR

Note: Additional explanation on the byte order:

In the following example, the whole number 439.041.101 is saved as a 32-bit integer value from memory address 10000.

	Big endian			Little endian		
Addresses	Hex	Dez	Binary	Hex	Dez	Binary
10000	1A	26	00011010	4D	77	01001101
10001	2B	43	00101011	3C	60	00111100
10002	3C	60	00111100	2B	43	00101011
10003	4D	77	01001101	1A	26	00011010

native Mdn_GetPackedB(const data{}, pos, block{}, size);
Reads a data block that is located at the specified position in an array

Parameter	Explanation
data	Array that contains the data
pos	Byte offset within the array to determine the position from which the data should be read
block	Array to store the data to be read
size	Number of bytes that should be read

	Explanation
Return value	• OK, if successful • ERROR

11.3.5 Messkanäle

11.3.5.1 Constants

Numbers of the channels

```
MDN_CH_UI1              // Universal input 1
:
MDN_CH_UI8              // Universal input 8
MDN_CH_EXTTEMP          // Ext. temperature sensor
MDN_CH_GSM              // Internal measurement channel "GSM level"
MDN_CH_VIN              // Internal measurement channel "Voltage"
MDN_CH_IOUT1            // Analogue output 1
MDN_CH_IOUT2            // Analogue output 2
MDN_CH_REL1             // Relay 1
:
MDN_CH_REL6             // Relay 6
MDN_CH_IN1              // Interface channel 1
:
MDN_CH_IN64             // Interface channel 64
MDN_CH_OUT1             // Interface output channel 1
:
MDN_CH_OUT64            // Interface output channel 64

//Extension module 1
MDN_CH_MUCE1_UI1        // Universal input 1
:
MDN_CH_MUCE1_UI8        // Universal input 8
MDN_CH_MUCE1_EXTTEMP    // Ext. temperature sensor
MDN_CH_MUCE1_IOUT1      // Analogue output 1
MDN_CH_MUCE1_IOUT2      // Analogue output 2
MDN_CH_MUCE1_REL1       // Relay 1
:
MDN_CH_MUCE1_REL6       // Relay 6

//Extension module 2
MDN_CH_MUCE2_UI1        // Universal input 1
:
MDN_CH_MUCE2_UI8        // Universal input 8
MDN_CH_MUCE2_EXTTEMP    // Ext. temperature sensor
MDN_CH_MUCE2_IOUT1      // Analogue output 1
MDN_CH_MUCE2_IOUT2      // Analogue output 2
MDN_CH_MUCE2_REL1       // Relay 1
:
MDN_CH_MUCE2_REL6       // Relay 6

//Extension module 3
MDN_CH_MUCE3_UI1        // Universal input 1
:
MDN_CH_MUCE3_UI8        // Universal input 8
MDN_CH_MUCE3_EXTTEMP    // Ext. temperature sensor
MDN_CH_MUCE3_IOUT1      // Analogue output 1
MDN_CH_MUCE3_IOUT2      // Analogue output 2
MDN_CH_MUCE3_REL1       // Relay 1
:
MDN_CH_MUCE3_REL6       // Relay 6
```

```
// Information regarding the status of the communication with the respective
extension module
MDN_CH_MUCE1_COM_STATE
MDN_CH_MUCE2_COM_STATE
MDN_CH_MUCE3_COM_STATE


// Number of channels
MDN_NUM_IN_CHANNELS   // Total number of interface channels
MDN_NUM_OUT_CHANNELS  // Total number of interface output channels
MDN_NUM_CHANNELS      // Total number of all channels (all of the in- and
                      // outputs)
```

Status of the measurement value/measurement channel
Coding to indicate various error statuses

```
MDN_STATUS_OK  = 0,   // Value is OK.
MDN_STATUS_NAN = 1,   // Invalid measurement value / undefined error
MDN_STATUS_OF  = 2,   // The measurement value is above the upper limit
(overflow).
MDN_STATUS_UF  = 3,   // The measurement value is below the lower limit
(underflow).
MDN_STATUS_OL  = 4,   // Cable break was detected or no sensor was connected.
MDN_STATUS_SC  = 5,   // Short circuit was detected.
```

Channel modes of the universal inputs

```
MDN_MODE_IN_NONE      = 0,        // Channel deactivated
MDN_MODE_IN_DIGITAL   = 1,        // Digital
MDN_MODE_IN_DCTRDAY   = 2,        // Day counter
MDN_MODE_IN_DCTRCONT  = 3,        // Interval counter
MDN_MODE_IN_DFREQ     = 4,        // Frequency
MDN_MODE_IN_DPWM      = 5,        // PWM
MDN_MODE_IN_A420MA    = 6,        // 4-20mA
MDN_MODE_IN_A020MA    = 7,        // 0-20mA
MDN_MODE_IN_A002V     = 8,        // 0-2V
MDN_MODE_IN_A010V     = 9,        // 0-10V
```

Channel modes of the output channels

```
MDN_MODE_OUT_NONE     = 100,      // Channel deactivated
MDN_MODE_OUT_EXTWARMUP = 101,     // Ext. warmup time
MDN_MODE_OUT_DIGITAL  = 102,      // Digital output
MDN_MODE_OUT_DFREQ    = 103,      // Frequency output
MDN_MODE_OUT_DPWM     = 104,      // PWM
MDN_MODE_OUT_DIMPULS  = 105,      // Pulse output
MDN_MODE_OUT_A420MA   = 106,      // 4-20mA
MDN_MODE_OUT_A020MA   = 107,      // 0-20mA
```

11.3.5.2 Functions

native Mdn_GetCh(ch, &Float:value, &Mdn_ValueStatus:status = MDN_STATUS_OK);
> *Reads out the current value and status of a channel*

Parameter	Explanation
ch	*Number of the channel (see "Numbers of the channels" in chapter "Constants" on page 154)*
value	*Value of the channel*
status	*Status of the channel (see "Status of the measurement value/measurement channel" in the chapter "Constants" on page 154) - OPTIONAL*

	Explanation
Return value	• *OK, if successful* • *ERROR, if no valid channel number was transferred*

native Mdn_GetChScale(ch, &Float:min, &Float:max, &Float:scale);
Reads the parameters of channel scaling and saves it in the transferred variables

Parameter	Explanation
ch	*Number of the channel (see "Numbers of the channels" in chapter "Constants" on page 154)*
min	

Channel type	Mode	Explanation
Universal inputs	*Digital Cnt.Day Cnt.Intervl. Freq*	*Not relevant*
	PWM 4-20 mA 0-20 mA 0-2 V 0-10 V	*Start of the measurement range in the measurement unit*
Ext. temperature sensor	*---*	*---*
Internal measurement channel	*---*	*---*
Analogue outputs	*All modes*	*Start of the output range in the measurement unit*
Relay	*---*	*---*
Interface channels	*If scaling is "on"*	*Start of the measurement range in the measurement unit*
Interface output channels	*If scaling is "on"*	*Start of the output range in the measurement unit*

Parameter	Explanation		
max			

Channel type	Mode	Explanation
Universal inputs	Digital Cnt.Day Cnt.Intervl. Freq	Not relevant
	PWM 4-20 mA 0-20 mA 0-2 V 0-10 V	End of the measurement range in the measurement unit
Ext. temperature sensor	---	---
Internal measurement channel	---	---
Analogue outputs	All modes	End of the output range in the measurement unit
Relay	---	---
Interface channels	If scaling is "on"	End of the measurement range in the measurement unit
Interface output channels	If scaling is "on"	End of the output range in the measurement unit

Parameter	Explanation			
scale				
	Channel type	Mode	Explanation	
	Universal inputs	Digital	1: Inverting "off" -1: Inverting "on"	
		Cnt.Day Cnt.Intervl.	Metered measurand of a pulse in the measurement unit	
		Freq	Factor by which the input signal is multiplied	
		PWM 4-20 mA 0-20 mA 0-2 V 0-10 V	Not relevant	
	Ext. temperature sensor	---	---	
	Internal measurement channel	---	---	
	Analogue outputs	All modes	Not relevant	
	Relay	---	---	
	Interface channels	---	Not relevant	
	Interface output channels	---	Not relevant	

	Explanation
Return value	• OK, if successful • ERROR, if no valid channel number was transferred

native Mdn_GetChMode(ch);
Returns the set mode of the channel

Parameter	Explanation
ch	Number of the channel (see "Numbers of the channels" in chapter "Constants" on page 154)

	Explanation
Return value	• Set channel mode (see "Channel modes of the universal inputs" or "Channel modes of the output channels" in chapter "Constants" on page 154) • ERROR, if no valid channel number was transferred

native Mdn_SetCh(ch, Float:value, Mdn_ValueStatus:status = MDN_STATUS_OK);

Sets the value and status of a channel. This function can also be used to change the value of an input before it is saved. This is possible, as the script is processed once the measurement value has been acquired but before it is recorded (see "Internal processing of the measurement values" on page 31).

Parameter	Explanation
ch	*Number of the channel (see "Numbers of the channels" in chapter "Constants" on page 154)*
value	*Value of the channel*
status	*Status of the channel (see "Status of the measurement value/measurement channel" in the chapter "Constants" on page 154) - OPTIONAL*

	Explanation
Return value	• *OK, if successful* • *ERROR, if no valid channel number was transferred*

11.3.6 Serial interfaces

11.3.6.1 Constants

Number of the interfaces

```
MUC_ITF_COM1 = 1,      // COM1 (RS485)
MUC_ITF_COM2,          // COM2 (RS232)
MUC_ITF_COM3,          // COM3 (RS485)
```

Serial events

Events that have to be processed by the Mdn_SerialEvent() function

```
MDN_SERIAL_EVENT_INIT    = 0,    // Script should send the init sequence.
                                 // (e.g. following activation of the sensor)
MDN_SERIAL_EVENT_MEASURE = 1,    // Script should send the measurement command.
MDN_SERIAL_EVENT_TIMEOUT = 2,    // Timeout upon receipt of the answer to the
                                 // measurement command
```

11.3.6.2 Callback functions

forward public Mdn_SerialEvent(com, event);

This function is the entry point for evaluating the serial events. This function must be provided by the user if script parsing is to be used.

Parameter	Explanation
com	Number of the interface (see "Number of the interfaces" in chapter "Constants" on page 160)
event	Event that caused this function to be called up • MDN_SERIAL_EVENT_INIT (Script should send the init sequence.) • MDN_SERIAL_EVENT_MEASURE (Script should send the measurement command.) • MDN_SERIAL_EVENT_TIMEOUT (Timeout upon receipt of the answer to the measurement command)

forward public Mdn_SerialRx(com, const data{}, len);

This function is the entry point for evaluating the received characters. This function must be provided by the user if script parsing is to be used.

Parameter	Explanation
com	Number of the interface (see "Number of the interfaces" in chapter "Constants" on page 160)
data	Array that contains the received characters
len	Number of characters that were received (max. 256)

11.3.6.3 Functions

native Mdn_SerialTx(com, const data{}, len);

Sends the transferred data block via the specified interface

Parameter	Explanation
com	Number of the interface (see "Number of the interfaces" in chapter "Constants" on page 160)
data	Array that contains the data to be sent
len	Number of bytes that should be sent

	Explanation
Return value	• OK, if successful • ERROR

native Mdn_SerialFinish(com);

Terminates the receipt of data via the specified interface. This function must always be called up once all of the required data has been received or the timeout event has occurred.

Parameter	Explanation
com	Number of the interface (see "Number of the interfaces" in chapter "Constants" on page 160)

	Explanation
Return value	• OK, if successful • ERROR

11.3.7 Alarm & Trigger

11.3.7.1 Arrays with symbolic indices

Mdn_AlarmCfg

Alarm configuration of a measurement channel

```
//Relevant for all channel modes except for "Digital"
// WarnValueLow    A warning is triggered,
//                 if the measurement value drops to or below this value.

// WarnValueHigh   A warning is triggered,
//                  if the measurement value meets or exceeds this value.
// AlarmValueLow   An alarm is triggered,
//                 if the measurement value drops to or below this value.
// AlarmValueHigh  An alarm is triggered,
//                 if the measurement value meets or exceeds this value.
// Hyst            Hysteresis for all-clear in event of alarm/warning

//Relevant for the "Digital" channel mode
// Flags           MDN_FLG_WARNING:   "High" triggers a "warning".
//                 MDN_FLG_ALARM:     "High" triggers an "alarm".
//                 MDN_FLG_FAULTLOW:  "High" triggers a "fault warning".
//                 MDN_FLG_FAULTHIGH: "High" triggers a "fault alarm".

#define Mdn_AlarmCfg[ Float:.WarnValueLow, Float:.WarnValueHigh,
                      Float:.AlarmValueLow, Float:.AlarmValueHigh,
                      Float:.Hyst, .Flags                        ]
```

Mdn_TriggerCfg

Trigger configuration of a measurement channel

```
// Mode      Trigger mode (see "Trigger mode" in chapter
//           "Constants" on page 163)
// Flags     Indicates which actions should be initiated when the trigger
//           is active (see "Trigger flags" in chapter
//           "Constants" on page 163)
// Value     Levels for initiating the trigger

#define Mdn_TriggerCfg[.Mode, .Flags, Float:.Value]
```

11.3.7.2 Constants

Alarm flags

Flags to specify the alarm type to be set for the Mdn_SetAlarm() function or to determine the alarm type supplied by the Mdn_GetAlarmCfg() function.

```
MDN_FLG_WARNING   = 0b00000001,  // Warning
MDN_FLG_ALARM     = 0b00000010,  // Alarm
MDN_FLG_FAULTLOW  = 0b00000100,  // Technical error of low priority
MDN_FLG_FAULTHIGH = 0b00001000,  // Technical error of high priority
MDN_FLG_UNDERFLOW = 0b10000000,  // Bit set: Alarm/warning, as
                                 //          value <= level
                                 // bit deleted: Alarm/warning, as
                                 //                value >= level
```

Trigger flags

Trigger flags for the Mdn_SetTrigger() function, to evaluate the return value of the Mdn_GetTrigger() function and to evaluate the trigger configuration of a channel read by the Mdn_GetTriggerCfg() function

```
MDN_TRG_INTERN1    = 0b0000000000010000, // Device-specific
MDN_TRG_INTERN2    = 0b0000000000100000, // Device-specific
MDN_TRG_INTERN3    = 0b0000000001000000, // Device-specific
MDN_TRG_INTERN4    = 0b0000000010000000, // Device-specific
MDN_TRG_RECQUICK   = 0b0000000100000000, // Request fast recording.
MDN_TRG_RECSLOW    = 0b0000001000000000, // Request slow recording.
MDN_TRG_MEASSTART  = 0b0000010000000000, // Starts a new measurement
MDN_TRG_RECORD_ON  = 0b0000100000000000, // Start recording the current
                                         // measurement
MDN_TRG_RECORD_OFF = 0b0001000000000000, // Prevent current measurement from
                                         // being recorded
MDN_TRG_TXSTART    = 0b0010000000000000, // Request a single connection to the
                                         // server
MDN_TRG_CONTINOUS  = 0b1000000000000000, // Request an online connection to the
                                         // server
```

Trigger mode
To evaluate the trigger configuration of a channel read by the Mdn_GetTriggerCfg() function

```
MDN_TRG_MODE_NONE              = 0, // Channel deactivated, no trigger

//Relevant for all channel modes except for "Digital"
MDN_TRG_MODE_LESS_OR_EQUAL     = 1, // Initiation of the trigger, if
                                    // value <= level
MDN_TRG_MODE_GREATER_OR_EQUAL  = 2, // Initiation of the trigger, if
                                    // value >= level

//Relevant for the "Digital" channel mode
MDN_TRG_MODE_DI_RISING_EDGE    = 3, // Rising edge initiates the trigger.
MDN_TRG_MODE_DI_FALLING_EDGE   = 4, // Falling edge initiates the trigger.
MDN_TRG_MODE_DI_BOTH_EDGES     = 8, // Both edges initiate the trigger.
```

11.3.7.3 Functions

native Mdn_GetAlarm(ch);
Returns the alarm status of a channel

Parameter	Explanation
ch	Number of the channel (see "Constants" on page 154)

	Explanation
Return value	Positive, if successful (see "Alarm flags" in the chapter "Constants" on page 163), ERROR, if no valid channel was specified

native Mdn_GetAlarmCfg(ch, Config[Mdn_AlarmCfg]);
Returns the alarm configuration of a channel

Parameter	Explanation
ch	Number of the channel (see "Constants" on page 154)
Config	Structure for storing the alarm configuration (see "Mdn_AlarmCfg" in chapter "Arrays with symbolic indices" on page 162)

	Explanation
Return value	• OK, if successful • ERROR, if no valid channel number was transferred

native Mdn_SetAlarm(ch, alarm, Float:value, Float:level = 0.0);

Sets the alarm status of a channel. The "MDN_FLG_UNDERFLOW" alarm flag must be set to indicate that the alarm/warning was triggered because the alarm/warning level was undercut. The "MDN_FLG_ UNDERFLOW" alarm flag remains clear to indicate that the alarm/warning was triggered because the alarm/warning level was exceeded. The alarm detection of the system is executed before the script is processed (see "Internal processing of the measurement values" on page 31).

Parameter	Explanation
ch	Number of the channel (see "Constants" on page 154)
alarm	Alarm flags (see "Alarm flags" in chapter "Constants" on page 163). The value "0" clears the alarm status of the channel.
value	Value of the channel used for the alarm calculation
level	Contains the alarm threshold - OPTIONAL

	Explanation
Return value	• OK, if successful • ERROR, if no valid channel number was transferred

native Mdn_SetTrigger(trigger);

Sets global trigger events

Parameter	Explanation
trigger	Trigger flags (see "Trigger flags" in chapter "Constants" on page 163)

	Explanation
Return value	OK, if successful

native Mdn_GetTrigger();

Returns the global trigger flags

	Explanation
Return value	Global trigger flags (see "Trigger flags" in chapter "Constants" on page 163)

native Mdn_GetTriggerCfg(ch, Config[Mdn_TriggerCfg]);
  Returns the trigger configuration of a channel

Parameter	Explanation
ch	Number of the channel (see "Constants" on page 154)
Config	Structure for storing the trigger configuration (see "Mdn_TriggerCfg" in chapter "Arrays with symbolic indices" on page 162)

	Explanation
Return value	• OK, if successful • ERROR, if no valid channel number was transferred

11.3.8 Math

Helpful constants

Definition	Value	Description
M_E	2.7182818284590452354	e
M_LOG2E	1.4426950408889634074	$\log_2 e$
M_LOG10E	0.43429448190325182765	$\log_{10} e$
M_LN2	0.69314718055994530942	ln 2
M_LN10	2.30258509299404568402	ln 10
M_PI	3.14159265358979323846	π
M_PI_2	1.57079632679489661923	π/2
M_PI_4	0.78539816339744830962	π/4
M_1_PI	0.31830988618379067154	1/π
M_2_PI	0.63661977236758134308	2/π
M_2_SQRTPI	1.12837916709551257390	2/sqrt(π)
M_SQRT2	1.41421356237309504880	sqrt(2)
M_SQRT1_2	0.70710678118654752440	1/sqrt(2)

native fround(Float:x);
  Commercially rounds the transferred float

Parameter	Explanation
x	Float that should be rounded

	Explanation
Return value	Commercially rounded integral value

native min(value1, value2);
Supplies the smaller of the two transferred values

Parameter	Explanation
value1	Two values of which the smaller one is to be determined
value2	

	Explanation
Return value	The smaller of the two transferred values

native max(value1, value2);
Supplies the larger of the two transferred values

Parameter	Explanation
value1	Two values of which the larger one is to be determined
value1	

	Explanation
Return value	The larger of the two transferred values

native clamp(value, min=cellmin, max=cellmax);
Checks whether the transferred value is between "min" and "max"

Parameter	Explanation
value	Value that is to be checked
min	Lower limit
max	Upper limit

	Explanation
Return value	• "value" if the value is between "min" and "max" • "min" is the value is less than "min" • "max", if the value is greater than "max"

native swapchars(c);
Swaps the order of the bytes

Parameter	Explanation
c	Value for which the bytes should be swapped over

	Explanation
Return value	Value for which the bytes in parameter "c" are swapped over (the lowest byte becomes the highest byte)

The mode of operation of the following functions corresponds to that of the standard ANSI-C implementation:

native Float:sin(Float:x);
   Sine of x

native Float:cos(Float:x);
   Cosine of x

native Float:tan(Float:x);
   Tangent of x

native Float:asin(Float:x);
   Arcsine(x) in the range [-π/2, π/2], x element of [-1, 1]

native Float:acos(Float:x);
   Arccosine(x) in the range [0, π], x element of [-1, 1]

native Float:atan(Float:x);
   Arctangent(x) in the range [-π/2, π/2]

native Float:atan2(Float:y, Float:x);
   Arctangent(y/x) in the range [-π, π]

native Float:sinh(Float:x);
   Hyperbolic sine of x

native Float:cosh(Float:x);
   Hyperbolic cosine of x

native Float:tanh(Float:x);
   Hyperbolic tangent of x

native Float:exp(Float:x);
   Exponential function e^x

native Float:log(Float:x);
   Natural logarithm ln(x), x > 0

native Float:log10(Float:x);
   Logarithm as the basis 10 $log_{10}(x)$, x > 0

native Float:pow(Float:x, Float:y);
   x^y. An argument error has occurred if x = 0 and y <= 0, or if x < 0 and y is not a whole number.

native Float:sqrt(Float:x);
   Root x, x >= 0

native Float:ceil(Float:x);
   Smallest whole number that is not smaller than x

native Float:floor(Float:x);
   Largest whole number that is not larger than x

native Float:fabs(Float:x);
   Absolute value | x |

native Float:ldexp(Float:x, n);
   $x*2^n$

native Float:frexp(Float:x, &n);
   Breaks down x into a normalised mantissa in the range [1/2, 1] that is supplied as the result, and a potency of 2 that is filed in n. If x is zero, both parts of the result are zero.

native Float:modf(Float:x, &Float:ip);
   Breaks down x into an integral and residual part that both have the same prefix as x. The integral part is filed in ip, while the residual part is the result.

native Float:fmod(Float:x, Float:y);
> *Residual floating point of x/y with the same prefix as x. The result is dependent on the implementation, if y is zero.*

native isnan(Float:x);
> *Returns a value that is not equal to zero, if x is not a number*

11.3.9 Char & String

> **Note:** *You require the following include file to be able to use the functions in this chapter:* `#include <string>`

The mode of operation of the following functions essentially corresponds to that of the standard ANSI-C implementation:

native strlen(const string[]);
> *Supplies the length of the string (without '\0')*

Parameter	Explanation
string	*Character string for which the length has to be determined*

	Explanation
Return value	*Number of characters without the final '\0'*

native sprintf(dest[], maxlength=sizeof dest, const format[], {Float,Fixed,_}:...);
> *Saves the transferred format string in the array dest. The mode of operation of the functions corresponds to that of the "snprintf" function of the standard ANSI-C implementation.*

Parameter	Explanation
dest	*Array to store the formatted result*
maxlength	*Maximum number of characters that the array dest can store*
format	*The format character string to be used*

	Explanation
Return value	• *-1 in the event of a fault* • *Number of characters that would have been written if the array dest had been long enough (without '\0').* *The array dest is always assigned a final zero. The length of the array dest cannot be exceeded.*

native strcpy(dest[], const source[], maxlength=sizeof dest);

Copies the source character string to the array dest (including '\0').

Parameter	Explanation
dest	Array to store the character string that should be copied
source	Character string that should be copied
maxlength	Size (in Cells) of the array to store the string to be copied - OPTIONAL

	Explanation
Return value	Number of copied characters

native strcat(dest[], const source[], maxlength=sizeof dest);

Adds the source character string to the dest character string (including '\0')

Parameter	Explanation
dest	Array to store the result. This array already comprises one character string to which the source character string should be added.
source	Character string that should be added to the character string included in the array dest
maxlength	Size (in Cells) of the array to store the result - OPTIONAL

	Explanation
Return value	Number of added characters

native strcmp(const string1[], const string2[], length=cellmax);

Compares character string1 and string2

Parameter	Explanation
string1	The two character strings that are to be compared
string2	
length	The maximum number of characters that should be taken into consideration during the comparison - OPTIONAL

	Explanation
Return value	1: string1 > string 2 0: both of the character strings are the same (at least the length that is taken into account) -1: string1 < string 2

native strchr(const string[], char);

Searches for a character (first occurrence) in a character string

Parameter	Explanation
string	*Character string that should be searched*
char	*Character that the search is looking for*

	Explanation
Return value	• *-1, if the character that the search is looking for is not included in the character string* • *Array index for the character that the search is looking for (first character occurring in the character string)*

native strrchr(const string[], char);

Searches for a character (last occurrence) in a character string

Parameter	Explanation
string	*Character string that should be searched*
char	*Character that the search is looking for*

	Explanation
Return value	• *-1, if the character that the search is looking for is not included in the character string* • *Array index for the character that the search is looking for (last character occurring in the character string)*

native strspn(const string1[], const string2[]);

*Searches for the position of the first character in string1 that is **not**included in the character string of permitted characters (string2)*

Parameter	Explanation
string1	*Character string that should be searched*
string2	*Character string of permitted characters*

	Explanation
Return value	• *Length of string1 if no permitted characters are found* • *Position of the first character in the character string that should be searched that is not included in the character string of permitted characters*

native strcspn(const string1[], const string2[]);

Searches for the position of the first character in string1 that is also included in the character string of permitted characters (string2)

Parameter	Explanation
string1	Character string that should be searched
string2	Character string of permitted characters

	Explanation
Return value	• Length of string1 if no permitted character has been found • Position of the first character in the character string that should be searched that is also included in the character string of permitted characters

native strpbrk(const string1[], const string2[]);

Searches the array index of the first character that is also included in the character string of permitted characters

Parameter	Explanation
string1	Character string that should be searched
string2	Character string of permitted characters

	Explanation
Return value	• -1, if the character that the search is looking for is not included in the character string • Array index of the first character in the character string that should be searched that is also included in the character string of permitted characters

native strstr(const string1[], const string2[]);

Searches character string2 in character string1

Parameter	Explanation
string1	Character string that should be searched
string2	Character string that the search should be for

	Explanation
Return value	• -1, if character string2 that is being searched for is not included in string1 • Array index where character string2 that is being searched for starts in string1

native strtol(const string[], base);

Converts a character string into a value

Parameter	Explanation
string	*Character string that is to be converted*

base	*Specifies the basis that must be used for the conversion*
	
	2-36: The specified basis is used
	0: 8, 10 or 16 is used as the basis, depending on the
	character string that must be converted
	  Basis 8: with a leading 0
	  Basis 16: with 0x or 0X

	Explanation
Return value	*Value that corresponds to the character string*

native Float: atof(const string[]);

Converts a character string into a float

Parameter	Explanation
string	*Character string that is to be converted*

	Explanation
Return value	*Float for which the numerical value corresponds to the character string*

native tolower(c);

Converts a character into lower case

Parameter	Explanation
c	*Character that should be converted to lower case*

	Explanation
Return value	*The lower case variant of the transferred character, if available, or the unchanged character code of "c" if the letter "c" does not have a lower case equivalent.*

native toupper(c);
Converts a character into upper case

Parameter	Explanation
c	Character that should be converted to upper case

	Explanation
Return value	The upper case variant of the transferred character, if available, or the unchanged character code of "c" if the letter "c" does not have a upper case equivalent.

11.3.10 Various

11.3.10.1 Arrays with symbolic indices

Mdn_TablePoint
Two-column reference point table, float data type

```
// key     Column that is searched
// value   Column with the result values that need to be returned

#define Mdn_TablePoint[Float:.key, Float:.value]
```

11.3.10.2 Constants

Error codes for the "Mdn_TablePoint" function

```
MDN_TAB_ERR_FLOOR = -1, // searched value lower than the first table entry
MDN_TAB_ERR_CEIL  = -2, // searched value higher than the last table entry
```

11.3.10.3 Functions

native getapilevel();
Issues the implemented API level of the script engine

	Explanation
Return value	Implemented API level of the script engine

native CRC16(data{}, len);
Returns the calculated modbus CRC16 of the transferred data

Parameter	Explanation
data	Array that contains the data for which the CRC16 must be calculated
len	Number of bytes that must be taken into consideration during the calculation

	Explanation
Return value	Calculated CRC16

native CRC32(data{}, len);
Returns the calculated Ethernet CRC32 of the transferred data

Parameter	Explanation
data	Array that contains the data for which the CRC32 must be calculated
len	Number of bytes that must be taken into consideration during the calculation
initial	Initial value for calculating the CRC32 - OPTIONAL

	Explanation
Return value	Calculated CRC32

native LRC(data[], len);
Returns the calculated modbus LRC of the transferred data

Parameter	Explanation
data	Array that contains the data for which the LRC must be calculated
len	Number of bytes that must be taken into consideration during the calculation

	Explanation
Return value	Calculated LRC

native Mdn_CalcTable(Float:key, &Float:value, const table[][Mdn_TablePoint], size = sizeof table);

Searches for a certain value in the "key" column of the transferred reference point table and supplies the relevant value from the "value" column in the table. If the searched value is between two reference points, the returned value is interpolated linearly between the two adjacent values in the "value" column (linear equation: y = k*x + d). Non-linear characteristic curves (e.g. connection between ADC value -> temperature) can be reproduced with this function.

Parameter	Explanation
key	Value that is used for the search
value	Includes the result of the calculation by the function
table	The table that is searched must be a "Mdn_TablePoint" type table.
size	Number of elements in the table

	Explanation
Return value	• *OK, if the relevant value was found* • *MDN_TAB_ERR_FLOOR, if the key is smaller than the first table entry. "value" contains the first table entry.* • *MDN_TAB_ERR_FLOOR, if the key is larger than the last table entry. "value" contains the last table entry.*

Note: *Additional explanation on the "`table`" reference point table*

The rows of the table can be displayed in an x/y coordinate system. The values in the "key" column are displayed on the X axis and the associated values in the "value" column are displayed on the Y axis.
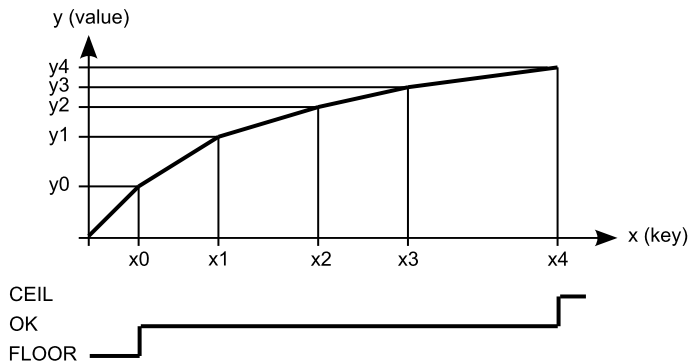
Display of the reference point table as an x/y coordinate system

native Mdn_WriteLog(log, param);
Generates an entry in the device log

Parameter	Explanation
log	Log entry to be generated (valid range 0...999). Some error codes have already been predefined in chapter "Predefined log entries" on page 179. If possible, you should use these for the described error situations. *Important note:* The log entries are mapped in the "MODULE ERR" (2000-2999) area for display on the server.
param	Additional parameter for the exact specification of the log entry (valid range -32768 .. 32767)

	Explanation
Return value	OK

native heapspace();
Supplies the free memory capacity to the heap

	Explanation
Return value	The free memory capacity to the heap. The stack and the heap have a joint memory area, so that this value specifies the number of bytes that remain for the stack or the heap.

native funcidx(const name[]);
Supplies the index of a public function

Parameter	Explanation
name	Name of the public function

	Explanation
Return value	• -1, if there is no function with the transferred name • Index of the public function

native numargs();
Returns the number of arguments transferred to a function

	Explanation
Return value	The number of arguments that have been transferred to a function. numargs is useful within functions with a variable list of arguments.

native getarg(arg, index=0);
Supplies the value of the argument

Parameter	Explanation
arg	The sequence number of the argument. Use 0 for the first argument.
index	Index if "arg" refers to an array

	Explanation
Return value	This function supplies an argument from a variable argument list. If the argument is an array, the "index" specifies the index of the required array element. The return value is the value of the argument.

native setarg(arg, index=0, value);
Sets the value of the argument

Parameter	Explanation
arg	The sequence number of the argument. Use 0 for the first argument.
index	Index if "arg" refers to an array
value	Value to which the argument should be set

	Explanation
Return value	• true, if the value could be set • false, if the argument or index are invalid This function sets an argument in a variable argument list. If the argument is an array, the "index" specifies the index of the required array element.

11.3.11 Console functions

Important note: The Com2 (115200 8N1) interface is used as the standard output on the myDatalogMUC if it is not being used otherwise.

native print(const string[]);
Prints the specified string to the standard output

Parameter	Explanation
string	The character string to be issued. This can include escape sequences.

	Explanation
Return value	OK

native printf(const format[], {Float,Fixed,_}:...);
> *Prints the transferred format string to the standard output. The mode of operation of the functions corresponds to that of the standard ANSI-C implementation.*

Parameter	Explanation
format[]	*The format character string to be used*

	Explanation
Return value	• *Number of printed characters* • *ERROR, if not successful*

> *Important note: Although the following function will still be supported for the purpose of downward compatibility, it should no longer be used for devices with modem version 03v006 or higher.*

native printi(value);
> *Prints the specified value on the standard output*

Parameter	Explanation
value	*The integer type value to be issued*

11.4 Predefined log entries

> *Important note: The log entries are mapped in the "MODULE ERR" (2000-2999) area for display on the server.*

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
0	LOG_COM2_ERROR	0	CHAR TIMEOUT	Com2: The time lag between two received characters is larger than the permissible character timeout.
		1	FRAME TIMEOUT	Com2: The complete answer was not received within the frame timeout.
		2	INVALID FRAME	Com2: The answer frame of the sensor is invalid.
1	LOG_CHAR_TIMEOUT	##	---	The time lag between two received characters is larger than the permissible character timeout.
2	LOG_FRAME_TIMEOUT	##	---	The complete answer was not received within the frame timeout.
3	LOG_FRAME_INVALID	##	---	The answer frame of the sensor is invalid.

11.5 Pawn script error codes

The PAWN script is run through at the time of every measurement once all of the measurement values have been generated. If an error occurs during this process, the script execution is stopped and deactivated. The relevant error code is also entered in the device log and a connection to the server is established. The parameter for all log entries except "SCRIPT_ERR" contains the 32-bit instruction pointer of the PAWN abstract machine (AMX). Two entries are generated in the device log as only 16-bit values can be saved in the parameter of a log entry. The first entry contains Bit31-Bit16 and the second entry contains Bit15-Bit0 of 32-bit instruction pointer. Instructions on evaluating the device log are included in the chapter "Evaluating the device log" (see "Evaluating the device log" on page 225).

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
3000	SCRIPT_ERR	0	NO SCRIPT	No valid script available
		1	SCRIPT UPDATE	New script received
3001	AMX_ERR_EXIT	##	---	Abortion e.g. max. number of PAWN commands (100,000) per run reached
3002	AMX_ERR_ ASSERT	##	---	Assertion failed
3003	AMX_ERR_ STACKERR	##	---	Stack/heap collision (insufficient stack size)
3004	AMX_ERR_ BOUNDS	##	---	Array index outside the valid range
3005	AMX_ERR_ MEMACCESS	##	---	Invalid memory access e.g. mix-up between cell (32-bit element) access [] and byte access {}
3006	AMX_ERR_ INVINSTR	##	---	Invalid statement
3007	AMX_ERR_ STACKLOW	##	---	Stack underflow
3008	AMX_ERR_ HEAPLOW	##	---	Heap underflow
3009	AMX_ERR_ CALLBACK	##	---	No (invalid) native callback function
3010	AMX_ERR_NATIVE	##	---	Native function failed
3011	AMX_ERR_DIVIDE	##	---	Divide by zero
3012	AMX_ERR_SLEEP	##	---	Sleep mode
3013	AMX_ERR_ INVSTATE	##	---	Invalid state
3014	reserved			

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
3015	reserved			
3016	AMX_ERR_ MEMORY	##	---	Out of memory
3017	AMX_ERR_ FORMAT	##	---	P-code file format is invalid/not supported
3018	AMX_ERR_ VERSION	##	---	File is for a newer version of AMX
3019	AMX_ERR_ NOTFOUND	##	---	File or function not found
3020	AMX_ERR_INDEX	##	---	Invalid index parameter (invalid entry point)
3021	AMX_ERR_DEBUG	##	---	Debugger cannot be executed
3022	AMX_ERR_INIT	##	---	AMX not initialised (or initialised twice)
3023	AMX_ERR_ USERDATA	##	---	User data field cannot be set (table full)
3024	AMX_ERR_INIT_ JIT	##	---	JIT cannot be initialised.
3025	AMX_ERR_ PARAMS	##	---	Faulty parameter
3026	AMX_ERR_ DOMAIN	##	---	Domain error. The result of the expression is not in the valid range.
3027	AMX_ERR_ GENERAL	##	---	General error (invalid or non-specific error)
3028	AMX_ERR_ OVERLAY	##	---	Overlays are not supported (JIT) or are not initialised.

11.6 Syntax

11.6.1 General syntax

11.6.1.1 Format

Identifiers, numbers and characters are separated by spaces, tabs, line breaks and "form feed". A series of one or more of these separators is recognised as an empty space.

11.6.1.2 Optional semicolons

Semicolons (used to finish a statement) are optional if they are at the end of a line. Semicolons are required to separate several statements in a line. An expression can be split across several lines, though the postfix operators must be on the same line as the operand.

11.6.1.3 Comments

Text between the /* and */ symbols (both symbols can be on the same or different lines) and text following // (to the end of the line) are comments. Comments must not be nested. The compiler considers comments to be blank space. A documentation comment is a comment that starts with "/** " (two stars and space after the second star) and ends with "*/". A comment that starts with "/// " (three forward slashes and a space after the third slash) is also a documentation comment. The parser can support the documentation comment in different ways, for example, by using it to generate online help.

11.6.1.4 Identifier

Names of variables, functions and constants. Identifier comprises the characters a...z, A...Z, 0...9, _ or @. The first character must not be a number. The characters @ and _ on their own are not valid identifiers, e.g. "_Up" is a valid identifier but "_" is not. Pawn distinguishes between upper and lower case. The parser cuts identifiers off after a certain length. By default, only the first 16 characters are referenced for distinguishing purposes.

11.6.1.5 Reserved keywords

Statements	Operator	Directives	Others
assert	defined	defined	defined
break	sizeof	sizeof	sizeof
case	state	state	state
continue	tagof	tagof	tagof
default			
do			
else			
exit			
for			
goto			
if			
return			
sleep			
state			
switch			
while			

11.6.1.6 Numerical constants

11.6.1.6.1 Numerical integer constants

Binary:
> *0b followed by a series of 0 and 1*

Decimal:
> *A series of numbers between 0 and 9*

Hexadecimal:
> *0x followed by a series of numbers between 0 and 9 and the letters a to f*

11.6.1.6.2 Numerical floating-point constants

A floating-point number is a number with numbers after the decimal point. A floating-point number starts with one or several numbers, includes a decimal point and has at least one number after the decimal point, e.g. "12.0" and "0.75" are valid floating-point numbers. An exponent can optionally be added. The notation is the letter "e" (lower case) followed by an integer numerical constant. For example, "3.12e4" or "12.3e-3" are valid floating-point numbers with an exponent.

11.6.2 Variables

11.6.2.1 Declaration

The keyword "new" declares a new variable. For special declarations, the keyword "new" is replaced with "static" (see "Static local declaration" on page 183). The value of the new variable is zero, provided that is not initialised explicitly.

A variable declaration can appear

- At every position at which an expression is valid - local variable
- At every position at which a function declaration or the implementation of the function is valid - global variables
- In the first expression of a "for" loop (see "For (expression 1; expression 2; expression 3) statement " on page 193) - local variable

Example:

```
new a;      // without initialisation (value is 0)
new b = 3; // with initialisation (value is 3)
```

11.6.2.2 Local declaration

A local declaration appears within a statement block. A variable can only be accessed within this block and the blocks that it comprises. A declaration within the first expression of a loop instruction is also a local declaration.

11.6.2.3 Global declaration

A global declaration appears outside of a function and a global variable can be used in any function. Global variables can only be initialised with constant expressions.

11.6.2.4 Static local declaration

A local variable is destroyed if the execution leaves the block in which the variable was created. Local variables in a function only exist during the operating time of the specified function. Each new call up of the function creates and initialises new local variables. The variable will also remain in the memory at the end of the function, if a local variable is declared with the keyword "static" instead of "new". This means that static, local variables provide permanent private storage that can only be accessed by a single function (or block). Static local variables can only be initialised with constant expressions, the same way as global variables.

11.6.2.5 Static global declaration

A static global variable acts in the same way as a global variable with the difference that the variable is only valid in the file in which it was declared. Replace the keyword "new" with "static" to declare a global variable as static.

11.6.2.6 Floating point values

The pawn supports floating point values. These can be added at every point at which a variable declaration is valid.

Example:

```
new Float:a;        // without initialisation (value is 0.0)
new Float:b = 3.0; // with initialisation (value is 3.0)
```

11.6.3 Constant variables

It is sometimes necessary to create a variable that is initialised once and is then not meant to be changed again. Such a variable acts in a similar way to a symbolic constant although it is still a variable. To declare a constant variable, place the keyword "const" between the keyword that starts the variable declaration ("new", "static") and the name of the variables.

Example:

```
new const address[4] = { 192, 0, 168, 66 }
static const status /* initialised to zero */
```

Typical situations in which you could use a constant variable, include:

- To create an "array" constant. Symbolic constants cannot be accessed via the index.
- It is a special case when array arguments are marked as "const" in a function. Arrays arguments are always transferred via a reference. If the arguments are declared to be "const", they are protected against unwanted changes. See examples of "const function arguments" in the chapter "Function arguments ("call-by-value" versus "call-by-reference")" on page 196.

11.6.4 Array variables

11.6.4.1 One-dimensional arrays

The name[constant] syntax declares the name as an array of "constant" elements, where each element is an entry. "name" is a placeholder for the name of the variable and "constant" is a positive value not equal to zero. "constant" is optional and can be omitted. If there is no value between the brackets, the number of elements is equal to the number of initial values. The array index area is "zero-based", which means that the first element is "name[0]" and the last element is "name[constant-1]".

11.6.4.2 Initialisation

Data objects can be initialised during their declaration. The initialised value from global data objects must be a constant value. Global or local arrays must also be initialised with constant values. Data that is not initialised are zero by default.

Example:

List: valid declaration

```
new i = 1
new j                            /* j is 0 */
new k = 'a'                      /* k has the character code of 'a' */
new a[] = [1,4,9,16,25]          /* a has 5 elements */
new s1[20] = ['a','b']           /* the remaining 18 elements are 0 */
new s2[] = ''Hello world...''    /* an unpacked string */
```

List: invalid declaration

```
new c[3] = 4                     /* An array cannot be set to an individual
                                    value */
new i = "Good-bye"               /* only an array can hold a string */
new q[]                          /* Unknown size for an array */
new p[2] = { i + j, k - 3 }      /* Array initialisers must be constants */
```

11.6.4.3 Progressive initialisation for arrays

The point operator continues the initialisation of the arrays based on the last two initialised values. The point operator (three points, "...") initialises the array up to the array limit.

Example: List: array initialisers

```
new a[10] = { 1, ... }           // sets all of the elements to 1
new b[10] = { 1, 2, ... }        // b = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
new c[8]  = { 1, 2, 40, 50, ... } // c = 1, 2, 40, 50, 60, 70, 80, 90
new d[10] = { 10, 9, ... }       // d = 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
```

11.6.4.4 Multi-dimensional arrays

(Only arrays with up to three dimensions are supported)

Multi-dimensional arrays are arrays that include references to other sub-arrays. For example, a two-dimensional array is an "array on one-dimensional arrays".

Example for the declaration of two-dimensional arrays:

```
new a[4][3]
new b[3][2] = [ [ 1, 2 ], [ 3, 4 ], [ 5, 6 ] ]
new c[3][3] = [ [ 1 ], [ 2, ...], [ 3, 4, ... ] ]
new d[2]{10} = [ "agreement", "dispute" ]
new e[2][] = [ ''OK'', ''Cancel'' ]
new f[][] = [ ''OK'', ''Cancel'' ]
```

As the last two declarations (variables "e" and "f") illustrates, the last dimension has an unspecified length. In this case, the length of the sub-array is detected by the associated initialiser. Each sub-array is a different length. In this specific example, "e[1][5]" includes the letter "l" of the word "Cancel". However, "e[0][5]" is invalid as the sub-array e[0] only comprises three entries (the letters "O", "K" and the zero terminator). The difference between the declarations of the "e" and "f" arrays is that we enable the compiler to determine the

number of higher dimensions for "f". "sizeof f" and "sizeof e" are both 2 (see "Arrays and the "sizeof" operator" on page 186).

11.6.4.5 Arrays and the "sizeof" operator

The "sizeof" operator returns the number of elements of a variable. The "sizeof" result of a simple (non array) variable is always 1.

An array with one dimension comprises a number of elements and the "sizeof" operator returns this quantity. The code section below would therefore issue "5", as the array comprises four characters and the zero terminator.

```
new msg[] = ''Help''
printf(''%d'', sizeof msg);
```

The "sizeof" operator always returns the number of entries even for a "packed" array. The code section below also issues "5", as the variable comprises five entries even though it requires less memory space.

```
new msg{} = "Help"
printf(''%d'', sizeof msg);
```

For multi-dimensional arrays, the "sizeof" operator can return the number of elements for every dimension. An element in the last (lowest) dimension is a single entry, while it is a sub-array in the highest dimension. Please note that in the following code section, the "sizeof matrix" syntax returns the number of elements of the higher dimension and that the "sizeof matrix[]" syntax issues the lower dimension of the two-dimensional array. The code section issues three (higher dimension) and two (lower dimension).

```
new matrix[3][2] = { { 1, 2 }, { 3, 4 }, { 5, 6 } }
printf(''%d %d'', sizeof matrix, sizeof matrix[]);
```

The application of the "sizeof" operator on multi-dimensional arrays is particularly practical when it is used as a standard value for function arguments.

11.6.5 Operators and expressions

11.6.5.1 Notational conventions

The use of some operators is dependent on the relevant type of operand. The following notations are therefore used in this chapter:

e

Any expression

v

Any expression that can be assigned a value ("lvalue" expression - variable)

a

An array

f

A function

s

A symbol - this can be a variable, a constant or a function

11.6.5.2 Expressions

An expression consists of one or several operands with an operator. The operand can be a variable, a constant or another expression. An expression followed by a semicolon is a statement.

Examples of expressions:

```
v++ f(a1, a2)
v = (ia1 * ia2) / ia3
```

11.6.5.3 Arithmetic

Operator	Example	Explanation
+	e1 + e2	Result of adding e1 and e2
-	e1 - e2	Result of subtracting e2 from e1
	-e	Result of the arithmetic negation of e (two's complement)
*	e1 * e2	Result of multiplying e1 with e2
/	e1 / e2	Result of dividing e1 by e2. The result is truncated to the closest whole number that is less or equal to the quotient. Positive and negative values are rounded down (negative infinity).
%	e1 % e2	Result is the remainder of the division of e1 by e2. The prefix is the same as that of e2
++	v++	Increases v by 1. The result of the expression is the value before the increase.
	++v	Increases v by 1. The result of the expression is the value following the increase.
--	v--	Decreases v by 1. The result of the expression is the value before the decrease.
	--v	Decreases v by 1. The result of the expression is the value following the decrease.

Note: The unary + is not defined in the pawn. The operators ++ and -- change the operand. The operand must be a "lvalue".

11.6.5.4 Bit manipulation

Operator	Example	Explanation
~	~e	The result is the one's complement of e.
>>	e1 >> e2	The result of the arithmetic shift to the right of e1 by e2 bits. The shift is signed: The bit on the far left is copied to the free bits of the result.
>>>	e1 >>> e2	The result of the logical shift to the right of e1 by e2 bits. The shift is unsigned. The free bits of the result are filled with 0.
<<	e1 << e2	Result: Shift to the left of e1 by e2 bits. The free bits of the result are filled with 0. There is no difference between an arithmetic and a logical shift to the left.
&	e1 & e2	The result is the bitwise logical "and" of e1 and e2.
|	e1 | e2	The result is the bitwise logical "or" of e1 and e2.
^	e1 ^ e2	The result is the bitwise logical "exclusive or" of e1 and e2.

11.6.5.5 Assignment

The result of an assignment expression is the value of the operand following the assignment.

Operator	Example	Explanation
=	v = e	Assigns the value of e to the variable v
	v = a	Assigns the array a to variable v. v must be an array of the same size and with the same dimensions as a. a can be a character string or an array.

Note: The following operators combine an assignment with an arithmetic or bitwise operation. The result of the expression is the value of the left operand following the arithmetic or bitwise operation.

Operator	Example	Explanation
+=	v += e	Increases v by e
-=	v -= e	Decreases v by e
*=	v *= e	Multiplies v with e
/=	v /= e	Divides v by e
%=	v %= e	Assigns v the remainder of the division of v and e
>>=	v >>= e	Arithmetically shifts v to the right by e bits
>>>=	v >>>= e	Logically shifts v to the right by e bits
<<=	v <<= e	Shifts v to the left by e bits
&=	v &= e	Executes a bitwise "and" from v and e and assigns the result to v
|=	v |= e	Executes a bitwise "or" from v and e and assigns the result to v
^=	v ^= e	Executes a bitwise "exclusive or" from v and e and assigns the result to v

11.6.5.6 Comparative operators

A logical "false" is represented by an integer value of 0; a logical "true" is represented by a value that is not 0. Results of a comparative expression are either 0 or 1 and the "tag" is set to "bool".

Operator	Example	Explanation
==	e1 == e2	The result is "true" if e1 and e2 are the same.
!=	e1 != e2	The result is "true" if e1 and e2 are not the same.

Note: The following operators can be linked, the same as in the expression "e1 <= e2 <= e3". This means that the result is "1" if every single comparison is true and "0" if at least one comparison is false.

Operator	Example	Explanation
<	e1 < e2	The result is a logical "true" if e1 is less than e2.
<=	e1 <= e2	The result is a logical "true" if e1 is less or equal to e2.
>	e1 > e2	The result is a logical "true" if e1 is greater than e2.
>=	e1 >= e2	The result is a logical "true" if e1 is greater or equal to e2.

11.6.5.7 Boolean

A logical "false" is represented by an integer value of 0; a logical "true" is represented by a value that is not 0. Results of a comparative expression are either 0 or 1 and the "tag" is set to "bool".

Operator	Example	Explanation
!	!e	The result is a logical "true", if e is logical "false".
||	e1 || e2	The result is "true", if either e1 or e2 (or both) are logical "true". The expression e2 is only evaluated if e1 is logical "false".
&&	e1 && e2	The result is "true" if e1 and e2 are logical "true". The expression e2 is only evaluated if e1 is logical "true".

11.6.5.8 Other

Operator	Example	Explanation
[ ]	a[e]	Array index: The result is the entry at position e of array a.
{ }	a{e}	Array index: The result is the index at position e of "packed" array a:
( )	f(e1, e2, ... eN)	The result is the value that is returned by function f. The function is called up with parameters e1, e2, ... eN. The sequence of the evaluation of the parameters is not defined. (The implementation of the script engine may evaluate the parameters in reverse order.)
? :	e1 ? e2 : e3	The result is either e2 or e3, depending on the value of e1. The conditional expression is a composite expression with a two-part operator, "?" and ":". The expression e2 is evaluated if e1 is logical "true"; e3 is evaluated if e1 is logical "false".
:	tagname: e	"Tag" overwritten: The value of the expression does not change, although the "tag" does change.
defined	defined s	Result is "1" if the symbol was defined. The symbol can be a constant or a global or local variable. The "tag" of the expression is "bool"
sizeof	sizeof s	The result is the number of elements of the specified variable. An element is an entry for simple variables and for one dimensional arrays. For multi-dimensional arrays, the result is the number of elements (sub-arrays) in the highest dimension. Add [] to the name of the array to specify a lower dimension. The result is 0 if the size of the variable is not known. If this operator is used in a "default" value of a function, the expression is executed at the time that the function was called up and not at the time that the definition was completed.
tagof	tagof s	The result is a unique number that represents the "tag" of the variables, the constants, the return value of a function or the name of the "tag" title. If this operator is used in a "default" value of a function, the expression is executed at the time that the function was called up and not at the time that the definition was completed.

11.6.5.9 Priority of the operators

The following table groups the operators with the same priority, starting with the highest priority at the top of the table.

If the evaluation of an expression is not explicitly justified with brackets, it is categorised by the association rules. For example: a*b/c is equal to (a*b)/c based on the left to right association, and a=b=c is equal to a=(b=c).

Operator	Explanation	Reading order
( ) [ ] { }	Function call array index (element) array index (character)	left-to-right
! ~ - ++ -- : defined sizeof tagof	logical not one's complement two's complement (unary minus) increase decrease "tag" overwritten symbol definition status symbol size in "elements" unique number of the tag	right-to-left
* / %	multiplication division modulo	left-to-right
+ -	addition subtraction	left-to-right
>> >>> <<	arithmetic shift to the right logical shift to the right shift to the left	left-to-right
&	bitwise "and"	left-to-right
^	bitwise "exclusive or"	left-to-right
|	bitwise "or"	left-to-right
< <= > >=	less than less or equal to greater than greater or equal to	left-to-right
== !=	equal unequal	left-to-right
&&	logical "and"	left-to-right
||	logical "or"	left-to-right
? :	conditional execution	right-to-left
=	Assignment *= /= %= += -= >>= >>>= <<= &= ^= |=	right-to-left
,	comma	left-to-right

11.6.6 Statements

A statement can comprise one or several lines. A line can comprises two or more statements.

Statements for the sequence control (if, if-else, for, while, do-while and switch) can be nested.

11.6.6.1 Statement label

A label consists of an identifier followed by a ":". A label is a "Jump target" of a "goto" statement.

Each statement can be marked with a label. The label must be followed by a statement, which can also be an "empty statement".

The scope of a label is the function in which it was declared, i.e. a "goto" statement cannot jump from the current function to another function.

11.6.6.2 Composite statements

A composite statement (also known as a block) is a series of zero or several statements that is enclosed by brackets ("{" and "}"). The closing bracket ("}") must not be followed by a semicolon. Each statement can be replaced by a block. A composite statement that does not comprise any statements is a special case and is known as an "empty statement".

11.6.6.3 Expression statement

Each expression becomes a statement when a semicolon (";") is added. An expression also becomes a statement, if the expression is only followed by blank spaces to the end of the line, and the expression is not continued in the next line.

11.6.6.4 Empty statement

An empty statement does not execute any statements and consists of a block statement without statements, i.e. it consists of the "{ }" symbol. Empty statements are implemented in control flow statements without actions (e.g. "while (!iskey()) {}") or if a label is defined exactly before the closing bracket of a block statement. An empty statement does not end with a semicolon.

11.6.6.5 Assert expression

The program is aborted with a runtime error if the expression is logical "false"

> *Note: This expression protects against "impossible" or invalid conditions. In the following example, a negative fibonacci number is invalid. The assert statement marks this error as a programming error. Assert statements should only ever highlight programmer errors and never user inputs.*

Example:

```
fibonacci(n)
{
  assert n > 0

  new a = 0, b = 1
  for (new i = 2; i < n; i++)
  {
    new c = a + b
    a = b
    b = c
  }
  return a + b
}
```

11.6.6.6 Break

Terminates and leaves the smallest, encircling "do", "for" or "while" statement at any point in the loop. The "break" statement moves the program flow to the next statement outside the loop.

Example:

```
example(n)
{
  new a = 0

  for(new i = 0; i < n ; i++ )
  {
    a += i

    if(i>10)
      break

    a += 1
  }
  return a
}
```

11.6.6.7 Continue

Terminates the current iteration of the smallest encircling "do", "for" or "while" statement and moves the program control to the conditional part of the loop.

Example

```
example(n)
{
  new a = 0

  for(new i = 0; i < n ; i++ )
  {
    a += i

    if(i>10)
      continue

    a += 1
  }
  return a
}
```

11.6.6.8 Do statement while (expression)

Executes a statement before the conditional part (the "while" condition) is evaluated. The statement is repeated as long as the condition is logical "true". The statement is executed at least once.

Example:

```
example(n)
{
  new a = 0

  do
  {
    a++
  }
  while(n >= 0)

  return a
}
```

11.6.6.9 Exit expression

Cancels the program. The expression is optional, however, if present it must start and end on the same line as the "exit" statement. The exit statement returns the expression value or zero to the main application, if no expression is specified.

11.6.6.10 For (expression 1; expression 2; expression 3) statement

All three of the expressions are optional.

Expression 1:
> Is only evaluated once before entering the loop. This expression can be used to initiate a variable. This expression also includes the variable declaration by means of the "new" syntax. A variable that is declared at this stage is only valid in the loop. It is not possible to combine an expression (with existing variables) and a declaration of new variables in this field. All of the variables must either already exist in this field, or they must all be declared in this area.

Expression 2:
> This expression is executed before every run of the loop and terminates the loop if the expression logical "false" is returned. If this expression is omitted, it is assumed that the result of expression 2 is logical "true".

Expression 3:
> This expression is executed each time the statement is completed. The program control moves from expression 3 to expression 2 for the next (conditional) iteration of the loop.

Example:

```
example(n)
{
  new a = 0

  for(new i = 0; i < n; i++)
  {
    a++
  }

  return a
}
```

The "for (; ;)" statement is the same as the "while (true)" statement.

11.6.6.11 Goto label

Moves the program control (unconditionally) to the statement that follows the specified label. The label must be within the same function as the "goto"-statement (a "goto"-statement cannot jump out of a function).

11.6.6.12 If (expression) statement 1 else statement 2

Executes statement 1 if the results of the expression is logical "true". The "else"clause of the "if" statement is optional. If the result of the expression is logical "false" and there is an "else" clause, the statement that is associated with the "else" clause (statement 2) is executed.

Example:

```
example(n)
{
  if(n < 0)
    return -1
  else if (n == 0)
    return 0
  else
    return 1
}
```

11.6.6.13 Return expression

Terminates the current function and moves the program control to the next statement following the function call. The expression value is returned as the function result. The expression can be an array or a character string. The expression is optional, however, if present it must start on the same line as the "return" statement. Zero is returned if no expression is specified.

11.6.6.14 switch (expression) {case list}

Transfers the sequence control to the various statements within the "switch", depending on the value of the "switch" expression. The main part of the "switch" statement is a composite statement that comprises a series of "case" clauses. Each "case" clause starts with the keyword "case" followed by a list of constants and a statement. The list of constants is a series of expressions separated by commas, each of which is evaluated as a constant value. This list ends with a colon. To specify an area in this list, separate the lower and upper limit of the area with a double point (".."). An example for an area is: "case 1..9:".

The "switch" statement shifts the sequence control to a "case" clause if a value from the list corresponds to the value of the "switch" expression.

The "default" clause consists of the "default" keyword and a double point. The "default" clause is optional, however, if it is specified it must be included as the last entry in the "case" list. The "switch" statement shifts the sequence control to the "default" clause if none of the "case" clauses comply with the "switch" expression.

Example:

```
example(n)
{
  new a = 0

  switch (n)
  {
    case 0..3:
      a = 0
    case 4,6,8,10:
      a = 1
    case 5,7:
      a = 2
    case 9:
      a = 3
    default:
      a = -1
  }

  return a
}
```

11.6.6.15 While (expression) statement

Evaluates the expression and executes the statement if the result of the expression is logical "true". The program control returns to the expression again once the statement has been executed. The statement is therefore executed as long as the expression is logical "true".

Example:

```
example(n)
{
  new a = 0

  while(n >= 0)
  {
    a++
  }

  return a
}
```

11.6.7 Functions

A function declaration specifies the name of the function and the formal parameters enclosed in brackets. A function can also return a value. A function must be defined globally, i.e. declared outside of another function and is globally available.

If the function declaration is followed by a semicolon (instead of a statement), this is a forward declaration of a function.

The "return" statement sets the return value of the function. For example, the return value of the "sum" function (see below) is the sum of both parameters. The "return" expression is optional.

```
sum(a, b)
{
  return a + b
}
```

The arguments of a function are (declared implicitly) local variables for this function. The function call specifies the values of the arguments. Another example of a complete definition of a function is "leap year" that indicates "true" or "false" for the relevant year.

```
leapyear(y)
{
  return y % 4 == 0 && y % 100 != 0 || y % 400 == 0
}
```

Details of the statements used in this example are provided in the chapter "Operators and expressions" on page 186.

Generally, functions include local variable declarations and consist of a block statement.

> *Note: In the next example, the "assert" statement prevents negative values for the exponent.*

```
power(x, y)
{
  /* returns x^y */
  assert y >= 0

  new r = 1
  for (new i = 0; i < y; i++)
    r *= x

  return r
}
```

A function can comprise several "return" statements, for example, one is used to quickly terminate a function if invalid parameters are transferred, or when it becomes apparent that the function has nothing to do. If a function returns an array, all of the "return" statements must return an array with the same number of entries.

11.6.7.1 Function arguments ("call-by-value" versus "call-by-reference")

The "faulty" function in the next example has a parameter that is used in the loop to calculate the factorial of this number. It must be noted that the function modifies the argument.

```
main()
{
  new v = 5
  new f = faculty(v)
}

faculty(n)
{
  assert n >= 0

  new result = 1
  while (n > 0)
    result *= n--

  return result
}
```

Regardless of what (positive) value the "n" variable has at the start of the "while" loop, "n" will equal zero at the end of the function. In the "faculty" function, for example, the parameter is transferred as a value ("by value"), which means that changes to the "n" variable are only valid locally in the "faculty" function. In other words, the "v" variable in the "main()" function has the same value before and after the function is called up.

Arguments can be transferred as a value ("by value") or as a reference ("by reference"). A function argument that is to be transferred as a reference must have the "&" prefix preceding the name. The arguments are transferred to the function as a value by default.

Example:

```
swap(&a, &b)
{
  new temp = b
  b = a
  a = temp
}
```

To transfer an array to a function, add a pair of brackets ("[]") to the name of the argument. The number of entries can also be specified. This improves the error detection of the compiler's parser.

Example:

```
addvector(a[], const b[], size)
{
  for (new i = 0; i < size; i++)
    a[i] += b[i]
}
```

Arrays are always transferred as a reference.

> **Note:** *The "b" array in the above-mentioned example is not changed in the function. This function argument was declared as a "const" to make this explicit. In addition to the improved error detection, it also enables the compiler to generate a more efficient code.*

The following code example calls up the "addvector" function and adds five to each element of the "vect" variables:

```
new vect[3] = [ 1, 2, 3 ]

addvector(vect, [5, 5, 5], 3)

/* vect[] now comprises the values 6, 7 and 8 */
```

11.6.7.2 Named parameters versus fixed parameters

In the previous examples, the order of the parameters in a function call were important as each parameter was copied to the same position of the function parameter. For example, in the "weekday" function (defined below), the expression "weekday(12 ,31, 1999)" would be used to get the weekday of the last day of the last century.

```
weekday(month, day, year)
{
  /* returns the day of the week: 0=Saturday, 1=Sunday, etc. */
  if (month <= 2)
    month += 12, --year

  new j = year % 100
  new e = year / 100
  return (day + (month+1)*26/10 + j + j/4 + e/4 - 2*e) % 7
}
```

The date format changes depending on the culture and country, while the USA use the month/day/year format, European countries frequently use the day/month/year format and technical publications use the year/month/day (ISO/IEC 8824) format. In other words, the sequence of the parameters is not "standardised" or "normal". For this reason, there is an alternative way of transferring parameters to a function, by using "named parameters". These are illustrated in the next example (the function was declared in the same way as the previous example).

```
new wkday1 = weekday( .month = 12, .day = 31, .year = 1999)
new wkday2 = weekday( .day = 31, .month = 12, .year = 1999)
new wkday3 = weekday( .year = 1999, .month = 12, .day = 31)
```

In "named parameters", a dot (".") precedes the name of the argument. The argument of the function can be set to any expression that is valid for the argument. In the event of a named parameter, the equals sign ("=") does not refer to an allocation but instead links the expression with a function argument.

Fixed and named parameters can be mixed together, although the fixed parameters must be specified before the named parameters.

11.6.7.3 Standard values of function arguments

A function argument can have a standard value. The standard value of a function argument must be a constant. To specify a standard value, add an equals sign ("=") and the value to the name of the parameter.

The standard value is adopted if a placeholder is specified instead of a valid function parameter during a function call. The placeholder is the underscore character ("_"). The argument placeholder is only valid for parameters with a standard value.

The right argument placeholders can be removed from the list of arguments.

For example, if the "increment" function is defined as follows:

```
increment(&value, incr=1)
{
  value += incr
}
```

The following function calls are all the same:

```
increment(a)
increment(a, _)
increment(a, 1)
```

Standard values for arguments that are transferred as a reference are helpful in making these parameters optional. For example, if the "divmod" function was written to return the quotient and the rest as a parameter.

```
divmod(a, b, &quotient=0, &remainder=0)
{
  quotient = a / b
  remainder = a % b
}
```

Based on the previous definition of the "divmod" function, the following function calls are all valid:

```
new p, q

divmod(10, 3, p, q)
divmod(10, 3, p, _)
divmod(10, 3, _, q)
divmod(10, 3, p)
divmod 10, 3, p, q
```

The next example adds the value of an array to another one. The values of the array are increased by one if only one parameter is specified:

```
addvector(a[], const b[] = {1, 1, 1}, size = 3)
{
  for (new i = 0; i < size; i++)
    a[i] += b[i]
}
```

11.7 Example

11.7.1 Saw-tooth generator

This example generates a saw-tooth on the output of a device.

```
static Float:fCurrentValue;            // Static declaration of the fCurrentValue variable of
                                       // the float type. This variable is not reset between
                                       // two measurement cycles

/*
  This code is only executed once when
  - starting the device
  - changing a script.
  This can be used to initialise
  variables.
*/
main()
{
  fCurrentValue = 4.0;                 // Initialise fCurrentValue with the value 4.0
}


/*
  This code is executed at the end of every
  measurement cycle.
*/
public Mdn_CtrlFinish()
{
  if(fCurrentValue <= 0.0)             // If the variable is less or equal to 0.0,
    fCurrentValue = 4.0;               // the variable is set to 4.0.

  fCurrentValue -= 0.1;                // Subtract 0.1 from the variable.

  Mdn_SetCh(MDN_CH_IOUT1, fCurrentValue); // Set output 1 to the value of the fCurrentValue
                                       // variable
}
```
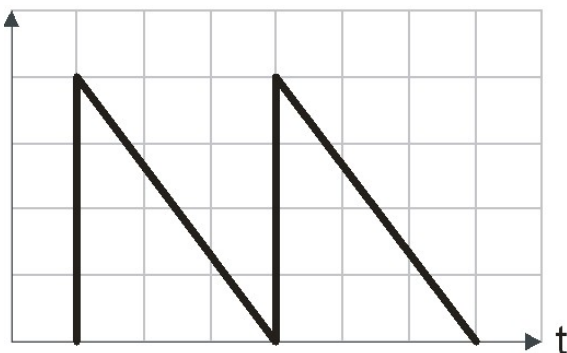

Saw-tooth generated by means of pawn

11.7.2 Weekday designation

This example determines the weekday from the current time stamp of a device. An interim value is then calculated by dividing the time difference between the cycles of the script by six. With a measurement cycle of 2 minutes this equates to a value of 20. This interim value is then also divided by the determined weekday and the result is issued at output 0.

```
static iLastStamp;                          // Static declaration of the iLastStamp
                                            // variable of the integer type (is
                                            // automatically pre-initialised with 0)
                                            // This variable is not reset between two
                                            // measurement cycles.
/*
  This code is executed at the end of every
  measurement cycle.
*/
public Mdn_CtrlFinish()
{
  new iDayOfWeek;                           // Declaration of the iDayOfWeek variable of
                                            // the integer type
  new iCurrentStamp;                        // Declaration of the iCurrentStamp variable
                                            // of the integer type
  new Float:fResult;                        // Declaration of the fResult variable of the
                                            // Float type
                                            // These variables are reset with every
                                            // measurement cycle

  printf("iLastStamp=%d\r\n", iLastStamp);  // Issue last time of a measurement via the
                                            // standard output

  if(iLastStamp != 0)                       // When the content of the iLastStamp variable
  {                                         // has already been set, the content of the
                                            // brackets is executed.

    iDayOfWeek = Mdn_DoW(Mdn_GetDate());    // Save week day in the iDayOfWeek variable

    iDayOfWeek += 1;                        // +1 so that the subsequent division is not
                                            // divided by 0 (Monday)

    printf("iDayOfWeek=%d\r\n", iDayOfWeek); // Issue the weekday via the standard output.


    iCurrentStamp = Mdn_GetTime();          // Read out the current time and save in the
                                            // iCurrentStamp variable.

    printf("iCurrentStamp=%d\r\n", iCurrentStamp); // Issue the current time via the standard
                                            // output.


    fResult = (iCurrentStamp - iLastStamp) / 6.0; // Determine the time difference between the
                                            // current and last time of a measurement
                                            // (measurement cycle) and divide by 6.

    printf("fResult=%f\r\n", fResult);      // Issue the time difference divided by 6 via
                                            // the standard output.

    fResult = fResult / iDayOfWeek;         // Divide the result by the weekday.
    printf("fResult=%f\r\n", fResult);      // Issue the result via the standard output.

    Mdn_SetCh(MDN_CH_IOUT1, fResult);       // Write the result on the output.
  }

  iLastStamp = Mdn_GetTime();               // Temporarily save the current time for
                                            // processing until the next time of a
                                            // measurement
}
```
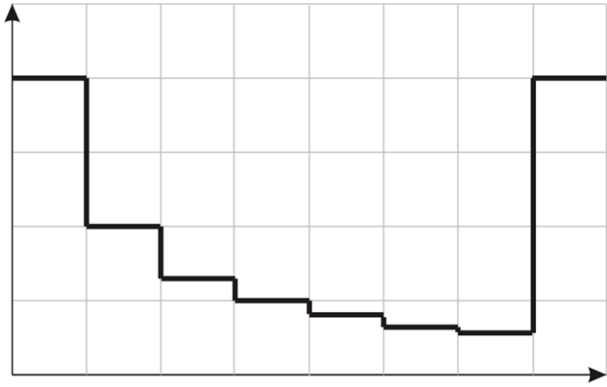
Mon Tue Wed Thu Fri   Sat   Sun Mon
Output signal generated by means of pawn

11.7.3 Calculating the flow rate with the table

This example determines the flow rate of a wastewater system based on the fill level of the wastewater system. The fill level is measured at input 1 and the flow rate should be issued at output 1.

```
static const Table[][Mdn_TablePoint] =         // Static table for converting the height
[                                              // to the flow rate
  [ 0.0,  0.0],
  [ 2.0,  1.0],
  [ 5.0,  3.0],
  [10.0, 10.0],
  [50.0, 15.0]
];


/*
  This code is executed at the end of every
  measurement cycle.
*/
public Mdn_CtrlFinish()
{
  new Float:fHeight;                           // Declaration of the fHeight variable of the
                                               // float type
  new Float:fFlow;                             // Declaration of the fFlow variable of the
                                               // float type
  new Mdn_ValueStatus:iStatus;                 // Declaration of the iStatus variable of the
                                               // Mdn_ValueStatus type
                                               // These variables are reset with every
                                               // measurement cycle

  Mdn_GetCh(MDN_CH_UI1, fHeight, iStatus);     // Read out channel 1 and save the value in
                                               // the fHeight variable and the status in the
                                               // iStatus variable.

  if(iStatus == MDN_STATUS_OK)                 // If the status of channel 1 is OK, execute
  {                                            // the following code in the {} brackets

    new iResult;                               // Declaration of the iResult variable of the
                                               // integer type

    iResult = Mdn_CalcTable( fHeight, fFlow,   // This function calculates the flow rate
                         Table, sizeof Table); // (fFlow) based on the height (fHeight)
                                               // and the table (table).
                                               // The result is linearly interpolated,
                                               // if the height is between 2 entries.
                                               // The return value contains the error value
                                               // of the calculation

    switch(iResult)
    {
      case MDN_TAB_ERR_FLOOR:                  // If the height is lower than the first entry
      {                                        // in the table,
        fFlow = 0.0;                           // set the flow rate to 0.0.
      }
      case MDN_TAB_ERR_CEIL:                   // If the height is greater than the last
      {                                        // entry in the table,
        fFlow = 0.0;                           // set the flow rate to 0.0.
      }
    }
  }

  Mdn_SetCh(MDN_CH_IOUT1, fFlow);              // Write the flow rate on output 1.
}
```

11.7.4 Script parsing

The following example initially issues the device class, modem version and firmware version on the standard output. "*INIT COM1\r\n*" is then issued as the init sequence on Com1. "*POLL COM1\r\n*" is issued in the measurement cycle on Com1. The received answer is issued on the standard output for control purposes. "*OUTPUT COM1\r\n*" is issued at Com1 as soon as the control module and thus the "Mdn_CtrlFinish()" function (see Mdn_CtrlFinish()) is called up during the internal processing of the measurement values.

```
public Mdn_CtrlFinish()
{
  Mdn_SerialTx(1, "OUTPUT COM1\r\n", 13);        // Issue "OUTPUT COM1" string via Com1
}


COM1Event(event)                                // Sub-function to evaluate the events of
Com1
{

  if(event == MDN_SERIAL_EVENT_INIT)            // If the init sequence should be sent
  {
    Mdn_SerialTx(1, "INIT COM1\r\n", 11);       // Send "INIT COM1" string via Com1
  }
  else if(event == MDN_SERIAL_EVENT_MEASURE)    // If the measurement command should be sent
  {
    Mdn_SerialTx(1, "POLL COM1\r\n", 11);       // Send "POLL COM1" string via Com1
  }
  else if(event == MDN_SERIAL_EVENT_TIMEOUT)    // If a timeout has occurred
  {
    Mdn_SerialFinish(1);                        // Terminate data reception via Com1
  }
}

public Mdn_SerialEvent(com, event)              // Access point for evaluating the serial
{                                               // events

  printf("Mdn_SerialEvent( %d, %d)\r\n",        // Issue number of the COM interface and
                       com, event);             // received event via the standard output

  switch(com)                                   // Check which COM interface has triggered
  {                                             // the event

    case 1: COM1Event(event);                   // Call up the sub-function for evaluating
  }                                             // the Com1 events
}

COM1Rx(const data{}, len)                       // Sub-function for evaluating the received
{                                               // data
  /*----------------------------------*/
  /*--- Code der die Daten auswertet --- */
  /*----------------------------------*/

  Mdn_SerialFinish(1);                          // Terminate receipt of data via Com1
}
```

```
public Mdn_SerialRx(com, const data{}, len)        // Access point for evaluating the received
{                                                  // characters

  printf("Mdn_SerialRx( %d, \"%s\", %d)\r\n",      // Issue number of the COM interface,
                    com, data, len);               // received data and number of characters via
                                                   // the standard output

  switch(com)                                      // Check via which COM interface the data was
  {                                                // received

    case 1: COM1Rx(data, len);                     // Call up the sub-function to evaluate the
  }                                                // data received via Com1
}

main()
{
  printf( "%04X DEVICE_CLASS\r\n", DEVICE_CLASS);   // Issue device class via the standard output
  printf( "%04X MODEM_VERSION\r\n", MODEM_VERSION); // Issue modem version via the standard
output
  printf( "%04X CONTROLLER_VERSION\r\n",            // Issue firmware version via the standard
        CONTROLLER_VERSION);                        // output
}
```

11.8 Differences to C

- The pawn is missing the input mechanism of C and is an "integer-only" variant of C. There are no structures or unions. Floating point support must be implemented with user-defined operators and the help of native functions.

- The syntax for floating point values is stricter than that in C. Values such as ".5" and "6." are acceptable in C but must be written as "0.5" and "6.0" in the pawn. The decimal point is optional in C. If an exponent is included, then you can write "2E8" in C, however the pawn will not accept the capital letter "E". Use the lower case letter "e". It requires a comma: e.g. "2.0e8" (see "Numerical constants" on page 182).

- The pawn does not support any "pointer". The pawn includes a "reference" argument to transfer function parameters as a reference (see "Function arguments ("call-by-value" versus "call-by-reference")" on page 196). The "placeholder" argument replaces some applications of the ZERO pointer (see "Standard values of function arguments" on page 198).

- Numbers can be specified in a hexadecimal, decimal or binary format. The octal format is not supported (see "Numerical constants" on page 182). Hexadecimal numbers must start with "0x" ("x" in lower case). The prefix "0X" is invalid.

- "Cases" in a "switch"-statement are not "fall through". At least one statement must follow the "case" label. You must create a composite statement (with {}) to execute several statements (see "switch (expression) {case list}" on page 194). The "switch" statement is a "conditional goto" in C/C++. The "switch" statement is a structured "if" in the pawn.

- A "break" statement only terminates loops. In C/C++, the "break" statement also terminates a "case" in a "switch" statement.

- The pawn supports "array assignments" with the limitation that both of the arrays must be the same length. For example, if "a" and "b" arrays have six lines, the expression "a=b" is valid. In addition to character strings, the pawn also supports literal arrays and thus expressions such as "a = {0,1,2,3,4,5}" where "a" is an array variable with six elements.

- "defined" is an operator and not a preprocessor directive. The "defined" operator in the pawn works with constants (declared with "const"), global variables, local variables and functions.

- The "sizeof" operator returns the size of the variables in "elements" and not in "bytes". An element is an entry or sub-array. Further details are provided in the chapter "Other" on page 189.

- An empty statement is an empty block (with {}) and not a semicolon (see "Composite statements" on page 191). This change prevents frequent errors.

- A division is completed in such a way that the remainder of the division has (or ought to have) the same prefix as the denominator. Divisions (operator "/") are always rounded down to the smaller whole number (whereby -2 is smaller than -1). For example, 5/2 = 2 (2.5 is rounded down to 2), -5/2 = -3 (-2.5 is rounded down to -3). The "%" operator always generates a positive result regardless of the prefix of the numerator (see "Operators and expressions" on page 186).

- There is no unary "+" operator as it is a "no-operation" operator anyway ("a = +1" is not valid; correct: "a = 1").

- Three bit by bit operators have different priorities than in C. The priority level of the "&", "^" and "|" operator is higher than the relational operators. Dennis Ritchie explains that these operators were assigned a low priority level in C as early C compilers did not yet include the logical "&&" and "||" operators so that bit by bit "&" and "|" were used instead.

- The keyword "const" in the pawn implements the "enum" functionality of C.

- In most cases, the forward declarations of functions (i.e. prototypes) are not necessary. A pawn is a two-pass compiler. It detects all of the functions during the first cycle and uses them during the second. User-defined operators must however be declared before use. If available, forward declarations must be exactly the same as the definition of the function. The parameter names in the prototypes and the definitions of the functions must be identical. The pawn attends to the parameter name in the prototype due to the "named parameter" function. The pawn uses prototypes to call up the forward declared functions. To use these with the named parameters during this process, the compiler must already know the names of the parameters (and their position in the parameter list). The parameter names in the prototypes must therefore match those in the definitions.
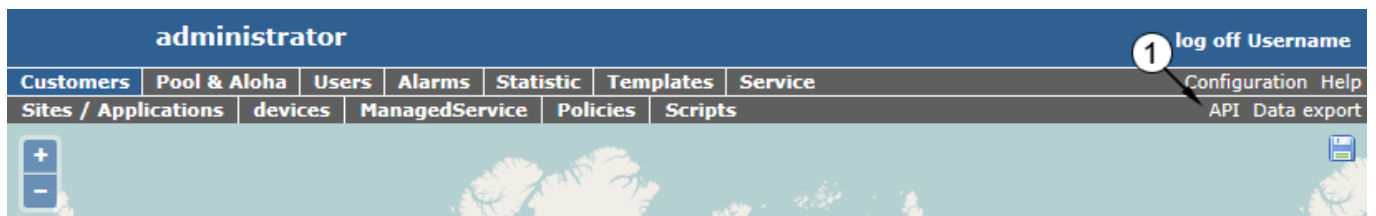
Chapter 12 API

Important note: The relevant licences are required on the myDatanet server to use the API (Application Programming Interface). For future information contact your responsible sales partner.

12.1 General

The API is provided to export data from the myDatanetserver and import data in to the myDatanet server. However, for "myDatalogMUC " type devices this is limited to the measurement data, the calculated channels, the setpoints of the output channels, the setpoints of the interface output channels, the transmission cycl and the record interval.

12.2 rapidM2M Playground

The rapidM2M Playground enables you to familiarize with the API of the myDatanet server and to test the functions provided. A click on the "API" button takes you to the rapidM2M Playground .

| 1 | opens the rapidM2M Playground |

12.2.1 Übersicht

rapidM2M Playground

1	input field for the user name
2	input field for the password
3	List of all available HTTP-commands. The HTTP commands are grouped according to application fields.
4	Dependent on the HTTP-command selected drop down lists for the selection of the client, the user and the measurement site are displayed, which should substitute the relative wildcard ("$CID"...client, "$UID"...user, "$SID"...site) in the resource path of the HTTP command.
5	button for the execution of the HTTP command
6	opens the website "http://rapidm2m.com/", which provides additional information for developers
7	opens the login dialogue of the myDatanet Server linked to the rapidM2M Playground
8	opens the quick guide for the API
9	button for changing the colour scheme of the rapidM2M Playground
10	window showing the HTTP command selected
11	response code sent by the myDatanet Server as a response to the HTTP command
12	copies the JSON-object, which was created as response to the HTTP command, to the clipboard
13	window showing the documentation of the selected HTTP command. This contains a description of the action to be executed by the selected HTTP command, notes to be considered and a description of the request body as well as the response body.
14	window showing the JSON-object, which is created as response to the HTTP command
15	window showing the last HTTP commands executed

Chapter 13 Maintenance

Important note: To prevent any damage to the device, the work described in this section of the instructions must only be performed by qualified personnel.

The device must be deenergised before any maintenance, cleaning and/or repair work.

13.1 General maintenance

- Regularly check the myDatalogMUC for mechanical damage.
- Check all cables for mechanical damage at regular intervals.
- Clean the myDatalogMUC with a soft, moist cloth. Use a mild cleaning agent, if necessary.

13.2 Fuse replacement

⚠️ *DANGER:*
Risk of fire. An incorrect fuse can cause injuries, damages or emissions. The fuse is located inside the housing. The housing may only be opened by the manufacturer.

If you believe that the fuse of the myDatalogMUC is defective (see "Troubleshooting and repair" on page 213), the device must be sent back in its original packaging to the manufacturer (see "Return" on page 52).

Chapter 14 Removal/disposal

Incorrect disposal can cause environmental hazards.

Dispose of the device components and packaging material in accordance with the locally valid environmental regulations for electronic products.

- Disconnect any charging voltage that has been used.
- Disconnect any connected cables using a suitable tool.

Logo to the EU WEEE Directive

This symbol indicates that the requirements of Directive 2012/19/EU regarding the scrap disposal of waste from electric and electronic equipment must be observed. The device includes a battery or rechargeable battery (lithium) that must be disposed of separately.

Chapter 15 Troubleshooting and repair

15.1 General problems

Problem	Cause/solution
Device does not respond (status LED always off):	• Check the cable connections (see "Connecting the sensors, actuators and power supply" on page 56)
Communication problems	• Evaluate the blink code of the status LED (see "Status LED" on page 76). • Load the device log from the myDatanet server and use DeviceConfig for the evaluation (see "Evaluating the device log" on page 225).
Problems during communication via the LAN interface	• Check the cable connection (see "Overview" on page 22). • Check the configuration of the LAN interface (see "LAN settings" on page 137). • When using a static IP address, check whether there is an IP address conflict in the network. • Check whether an Internet connection is possible via the gateway connected to the myDatalogMUC . • Check whether port 51241 is open in your network.
Failure of the supply voltage is not indicated.	• The alarm for the internal "voltage" measurement value was not configured (see "Technical details about the integrated rechargeable buffer battery" on page 68).
Not all or no data is available on the server.	• The connection was interrupted during transmission, which is indicated by a time-out entry in the connection list (see "myDatanet Server Manual " 805002). Solution: Initiate a ALOHA transmission or wait for the next cyclical transfer. • The assignment of the device and site is not correct (see "Site" on page 79).
The data at the universal input is not plausible.	• Check the cable connections (see "Connecting the sensors, actuators and power supply" on page 56) • Check whether the output signal from the sensor that you are using is compatible with the electrical characteristics of the universal inputs (see "Technical details about the universal inputs" on page 62). • Check whether the universal input configuration matches the sensor output signal (see "Measurement channels" on page 92). • Check the filter settings of the universal input (see "Measurement channels" on page 92). • Check whether the ext. warmup time has been selected for the sensor that you are using (see "Output channels" on page 128).

Problem	Cause/solution
The interface channel data is not plausible (modbus, RS485).	• Check the cable connections (see "Connecting the sensors, actuators and power supply" on page 56) • Check that the selection of the modbus address and slave address (only in "modus master" mode) is set correctly in the interface channel configuration (see "Interface channels 1-32" on page 110). • Check the configuration of the modbus interface (see "Interfaces" on page 80). • Check the settings of the load or clamp resistances (see "Technical details about the modbus interfaces (Com1, Com3)" on page 63).
The interface channel data is not plausible (serial, RS232).	• Check the cable connections (see "Connecting the sensors, actuators and power supply" on page 56) • Check that the column selection is set correctly in the interface channel configuration (see "Interface channels 1-32" on page 110). • Check the configuration of the serial interface (see "Interfaces" on page 80). The number format or the number delimiter(s) may be configured incorrectly. • If there are less columns in the data message than specified during the configuration of the interface channels (see "Interface channels 1-32" on page 110), this is recognised as an invalid frame, no data is recorded and a corresponding entry is generated in the device log (see "Error codes of the RS232 interface " on page 66).
The alarm state of a measurement channel was not identified.	• Increase the measurement cycle.
The alarm state was not transmitted although the data is present.	• Check the alarm settings of the measurement channel • The connection was interrupted during transmission, which is indicated by a time-out entry in the connection list (see "myDatanet Server Manual " 805002). Solution: Initiate a ALOHA transmission or wait for the next cyclical transfer.
The alarm message was not sent although the alarm was signalled.	• Check the settings of the alarm schedule (see "myDatanet Server Manual " 805002). • Check the address data of the alarm schedule (see "myDatanet Server Manual " 805002).
The relays do not work.	• Disruption to the voltage that is conducted via the relays • The site settings were overwritten by a server-PLC.
The value set via the input screen on the myDatanet server is not issued at the output channel.	• The site settings were overwritten by a server-PLC. • Check whether the output channel is being written to via the script. The value specified via the input screen on the myDatanet server is overwritten with a value calculated by the script.
The instruction list is not being executed correctly.	• Check that the correct script type was selected during the configuration of the control (see "Control" on page 79). • Load the device log from the myDatanet server and use DeviceConfig for the evaluation (see "Evaluating the device log" on page 225).

Problem	Cause/solution
The pawn script is not being executed correctly.	• Check that the correct script type was selected during the configuration of the control (see "Control" on page 79). • Load the device log from the myDatanet server and use DeviceConfig for the evaluation (see "Evaluating the device log" on page 225). A list of all the possible pawn script error codes is included in the chapter "Pawn script error codes" (see "Pawn script error codes" on page 180).

15.2 Log entries and error codes

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1000	POWER ON	0	---	System start completed
		> 0	---	Internal system error Restart due to internal system error. There may be a hardware problem if the "POWER ON" log entry with a parameter code of >0 is contained in the device log several times. Contact the manufacturer in this case (see "Contact information" on page 237).
1020	ERROR SOD	##	---	Internal system error There may be a hardware problem if this error is contained in the device log several times with the same parameter code. Contact the manufacturer in this case (see "Contact information" on page 237).
1030	UV LOCKOUT	---	---	The device switches to energy saving mode and terminates all of the operations as the rechargeable battery or battery voltage is too low. Only the charge controller, if present, remains active.
1031	UV RECOVER	---	---	The rechargeable battery or battery voltage once again suffices to guarantee reliable operation. The device resumes normal operation in accordance with the configuration.
1033	MODEM_UPDATE	##	---	Update of the firmware for the modem controller was completed successfully. This entry is always duplicated in the device log. In the first entry, the parameter specifies the major version number (e.g. 3 for 03v011), while in the second entry it specifies the minor version number (e.g. 11 for 03v011).
1034	CONTROLLER_ UPDATE	##	---	Update of the firmware for the measurement controller was completed successfully This entry is always duplicated in the device log. In the first entry, the parameter specifies the major version number (e.g. 3 for 03v011), while in the second entry it specifies the minor version number (e.g. 11 for 03v011).

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1111	ALOHA STOP	0	---	End of ALOHA transmission mode
		##	---	ALOHA transmission mode was stopped unexpectedly. The parameter specifies how many more seconds ALOHA transmission mode should have lasted.
1114	BACKUP SUPPLY	1	---	The power supply was switched to the rechargeable buffer battery.
		0	---	The power supply was switched back to V IN.
1161	LOG REFORMATFILE	---	---	The file system was re-formatted. This means that all of the recorded data and log entries were deleted.
1200	MODEM ERROR			Modem error (see "Modem error" on page 222)
1202	MODEM CMME ERROR	##	---	The GPRS modem indicates a +CME error. The parameter specifies the type of error.
1203	SELECTED NETWORK	##	---	A new GSM network was selected. This entry is always duplicated in the device log. In the first entry, the parameter specifies the MCC (Mobile Country Code), while in the second entry it specifies the MNC (Mobile Network Code) of the selected GSM network.
1204	BANNED NETWORK	##		A GSM network was added to the ban list. This entry is always duplicated in the device log. In the first entry, the parameter specifies the MCC (Mobile Country Code), while in the second entry it specifies the MNC (Mobile Network Code) of the selected GSM network.
1205	BAN LIST CLEARED	---	---	The ban list was deleted as there were no GSM networks in the receiving range that have not been placed on the ban list.
1212	ERROR MODEM IRREGULAR OFF	##	---	Indicates a faulty connection. The parameter includes a counter that indicates how many consecutive connections have not worked.
1224	MODEM SIM RESET	---	---	The GPRS modem has been restarted to reinitialise the SIM chip again.
1252	MODEM TO CON	##	---	Timeout while a connection is being established. The parameter contains a counter that indicates how many consecutive connection attempts have not worked.

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1281	ZLIB STREAMPROCESS ERR	##	---	Internal error Contact the manufacturer if the device log includes this error several times (see "Contact information" on page 237).
1282	ZLIB STREAMFINISH ERR	##	---	Internal error Contact the manufacturer if the device log includes this error several times (see "Contact information" on page 237).
1320	LAN	0	LAN LINKUP	LAN cable was connected.
		1	LAN LINKDOWN	LAN cable was disconnected.
		2	LAN TIMEOUT	Timeout when establishing a connection to the server or when transferring data
		3	LAN OPEN SOCKET FAILED	No socket could be opened. TCP/IP connection could not be established.
		4	LAN DISCONNECT	LAN connection was disconnected.
		5	LAN NO IP	The LAN interface has not been assigned an IP address or an invalid one. No LAN cable is connected.
		6	LAN DHCP DISCOVER	"DHCP discover" was sent to determine whether a DHCP server is available.
		7	LAN DHCP REQ	An IP address was requested from the DHCP server.
		8	LAN DHCP REREQ	Request to continue to use the IP address received during the "DHCP request".
		9	LAN DHCP NEW IP	A new IP address was received from the DHCP server.
		10	LAN DHCP ABORT	Timeout during communication with the DHCP server.
		11	LAN MAC READ ERROR	The MAC address of the LAN interface could not be determined. This means the LAN interface cannot be used. Contact the manufacturer (see "Contact information" on page 237).

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1321	LAN_SOCK_STATE	23	LAN SOCK ESTABLISHED	Socket connection disconnected.
		24	LAN SOCK FIN WAIT	
		26	LAN SOCK CLOSING	
		27	LAN SOCK TIME WAIT	
		28	LAN SOCK CLOSE WAIT	
		29	LAN SOCK LAST ACT	
1440	DECAY MEM ERR	---	---	Too many measurement values are being taken into consideration during the decay (see "Measurement channels" on page 92). Either reduce the decay period or the measurement cycle.

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
1600	STATE	1	OK	Last connection OK
		2	TRANSMISSION ERROR	Last transmission faulty Try again later
		3	MEASUREMENT ERROR	Last measurement faulty • Check the configuration. • Check the connection between the device and sensor.
		4	HOLD	Standby (GPRS ON, measurement OFF) If ALOHA transmission mode is initiated on the device, the myDatalogMUC switches back into "RUN" mode (GPRS ON, measurement ON).
		5	TRANSPORT	Transport lock (GPRS OFF, measurement OFF) see "Standby"
		6	OFFLINE	Offline (GPRS OFF, measurement ON) see "Standby"
		7	NO NET (NET LOCK)	Netlock/no matching provider • Try to improve the position of the antenna. • Check whether the device is in the coverage area (www.microtronics.com/footprint).
		8	NO NET	No GSM network • Improve the position of the antenna • Try again later
		10	GPRS ERROR	No GPRS connection Try to improve the position of the antenna.
		11	IP ERROR	No myDatanet server available • Check whether port 51241 is enabled on the myDatanet server. • Try again later
		12	SIM CARD ERROR	Faulty SIM chip Contact support.

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
2000 - 2999	MODULE ERR			Module-specific error (see "Error codes of the RS232 interface " on page 66) or area for customer-specific error codes, that can be written in the device log by means of the "Mdn_WriteLog()" pawn script function (see "Various" on page 174).
3000 - 3099				Error codes of the script execution. The following error descriptions apply depending on the selected script type: • "Pawn script error codes" on page 180

15.2.1 Modem error

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
GPRS error				
1200	BEARER LINE BUSY	-993	---	Line busy Try again later
	BEARER NO ANSWER	-992	---	No answer • Try again later • Check whether the device is in the coverage area (www.microtronics.com/footprint).
	BEARER NO CARRIER	-991	---	No carrier • Try again later • Check whether the device is in the coverage area (www.microtronics.com/footprint).
	BEARER GPRS FAILED	-988	---	GPRS setup failure • Try to improve the position of the antenna • Check whether the device is in the coverage area (www.microtronics.com/footprint).
	BEARER PPP LCP FAILED	-987	---	Error when negotiating the LCP Internal error
	BEARER PPP AUTH FAILED	-986	---	Error occurred during PPP authentication Internal error
	BEARER PPP IPCP FAILED	-985	---	Error when negotiating the IPCP Internal error
	BEARER PPP LINK FAILED	-984	---	PPP receiver does not react to echo request Internal error
	BEARER PPP TERM REQ	-983	---	PPP session terminated by the receiver. Internal error
	BEARER CALL REFUSED	-982	---	Incoming call refused Try again later
	BEARER UNKNOWN	-981	---	Try again later

| Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
Error when establishing the GPRS connection				
1200	BEARER BAD HDL	-980	---	Invalid handling Internal error
	BEARER BAD STATE	-979	---	The carrier was not stopped. Internal error
	BEARER DEV	-978	---	Error occurred during link layer initialisation Internal error
	BEARER GPRS START UNKNOWN	-977	---	Internal error
Error when disconnecting the GPRS connection				
1200	BEARER GPRS STOP BAD HDL	-976	---	Invalid handling Internal error
	BEARER GPRS STOP UNKNOWN	-975	---	Internal error
1200	NETLOCK FAILURE	-970	---	Error when selecting the network • Check whether the device is in the coverage area (www.microtronics.com/footprint).
1200	BAND SEL FAILED	-969	---	A network could not be found on the GSM900/1800 or on the GSM850/1900 band. • Try to improve the position of the antenna. • Check whether the device is in the coverage area (www.microtronics.com/footprint).
TCP channel error (1/2)				
1200	CHANNEL ABORTED	-965	---	An attempt is being made to write to/read a TCP client that is no longer available. Try again later
	CHANNEL CSTATE	-964	---	The channel is not in the "WIP_CSTATE_READY" state. Try again later
	CHANNEL NOT SUPPORTED	-963	---	The option is not supported by the channel. Try again later

Log entry		Parameter		Description
Code	Plain text	Code	Plain text	
TCP channel error (2/2)				
1200	CHANNEL OUT OF RANGE	-962	---	The optional value is outside the permissible range. Try again later
	CHANNEL MEMORY	-961	---	Internal error
	CHANNEL INTERNAL	-960	---	Internal error
	CHANNEL INVALID	-959	---	Invalid option or parameter Internal error
	TCP DNS FAILURE	-958	---	The name could not be resolved in an IP address. Internal error
	CHANNEL RESOURCES	-957	---	There are no further TCP buffers available. Try again later
	CHANNEL PORT IN USE	-956	---	TCP server port is already being used. Try again later
	CHANNEL REFUSED	-955	---	The TCP connection has been refused by the server. Try again later
	CHANNEL HOST UNREACHABLE	-954	---	No route to the host. Try again later
	CHANNEL NETWORK UNREACHABLE	-953	---	No network available Try again later
	CHANNEL PIPE BROKEN	-952	---	TCP connection interrupted Try again later
	CHANNEL TIMEOUT	-951	---	Timeout (DNS request, TCP connection, ping response, etc.) Try again later
	CHANNEL UNKNOWN	-950	---	Internal error

15.3 Evaluating the device log

15.3.1 Evaluating the device log on the myDatanet server

The last 300 log entries on the myDatanet server can be called up via the button shown below that is located in the measurement device list. As the log entries are sent to the server in the transmission cycle in the same way as the measurement data, only the log entries up to the last server connection are available.

```
FW Ctrl.:   02.004
FW          02.005
Mdm.:       06.063g
FW OS.:
```
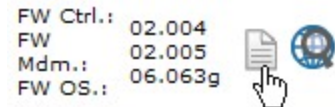
The manual for the server ("myDatanet Server Manual " 805002) includes a detailed description of the evaluation of the device log on the myDatanet server.

Chapter 16 Spare parts and accessories

16.1 Antennas

Description	Quantity	Order number
Antenna 900 SMA-M angled	1	300026
Antenna 900 SMA-M	1	206.806
Extension cable for antenna SMA-M/SMA-F 2,5m	1	206.807
Flat antenna Smart Disc SMA-M 2,5m	1	206.816
Flat Antenna Smart Disc US SMA-M 2,5m	1	206.818
Angle adapter SMA-M/SMA-F	1	300318

16.2 Power supply

Description	Quantity	Order number
Power supply 24V 2,5A for top-hat rail mounting	1	206.667

16.3 Adapter

Description	Quantity	Order number
Gender changer 9-pin D-Sub male/male	1	206.684
Null modem adapter 9-pin D-Sub female/male [1]	1	206.686

[1] Please ensure that pin 9 on the adapter is not looped through

16.4 Other accessories

Description	Quantity	Order number
myDatanet Tool Pen	1	206.646

Chapter 17 Document history

Rev.	Date	Changes
02	16.10.2012	**Chapter "Technical data" on page 15** *Number of available "Interface channels" and "Interface output channels" added* **Chapter "Block diagram" on page 22** *Chapter added* **Chapter "General product information" on page 24** *Explanation of the "Interface output channels" added* **Chapter "Error codes of the RS232 interface " on page 66** *Breakdown of the error codes added* **Chapter "Control" on page 79** *Selection of the pawn script* *Explanation adjusted to the modified input screen* **Chapter "Config" on page 90** *Incorrect explanation of the "Character timeout" corrected* **Chapter "Pawn script" on page 149** *Chapters describing the functionality of the pawn script added.*
03	18.12.2012	**Chapter "Automatic selection of the GSM network" on page 45** *Chapters describing the functionality of the GSM network selection added* **Chapter "Script eradication" on page 46** *Chapter added* **Chapter "API" on page 150** *Explanation of the core functions and mathematical functions added.* **Chapter "Log entries and error codes " on page 216** *Breakdown of the error codes added* **Chapter "Power supply" on page 227** *Power supply 12V 1,25A (206.623) added*
04 (1/3)	14.06.2013 (1/3)	**Chapter "Declaration of conformity" on page 13** *Declaration of Conformity updated* **Chapter "Technical data" on page 15** U_{max} *for the PWM, frequency, digital, day counter and interval counter modes added.* *Information regarding the country list profile of the SIM chip has been added.* **Chapter "Functional principle" on page 29** *Graphic added to the "Functional principle" chapter.* **Chapter "Internal processing of the measurement values" on page 31** *Chapter added* **Chapter "ALOHA transmission mode" on page 44** *Information added indicating that the GSM network is selected again when ALOHA transmission mode is initiated.* **Chapter "Assembly in a control cabinet" on page 54** *Chapter added*

Rev.	Date	Changes
04 (2/3)	14.06.2013 (2/3)	**Chapter "Connection examples" on page 61** *Chapter added.* **Chapter "Technical details about the modbus interfaces (Com1, Com3)" on page 63** *Detailed explanation regarding the permissible signal levels added.* **Chapter "Technical details about the RS232 interface (Com2)" on page 65** *Detailed explanation regarding the signal levels has been added.* **Chapter "Interfaces" on page 80** *Information added indicating which parameters can no longer be employed when script parsing is used.* **Chapter "Measurement channels" on page 92** *Additional explanation regarding the difference between "Cnt.Day" and "Cnt.Intervl." added* *Additional explanation regarding the "Out of range" and "Namur borders" modes added* **Chapter "Interface channels 1-32" on page 110** *List of the possible access functions for the modbus slave mode added.* *Information added indicating which parameters can no longer be employed when script parsing is used.* **Chapter "Interface output channels 1-32" on page 118** *List of the possible access functions for the modbus slave mode added.* *Information added indicating which parameters can no longer be employed when script parsing is used.* **Chapter "Basic setting" on page 134** *A more accurate explanation of the "Default view" parameter has been added* *Examples of the effects of the record intervals and measurements cycles on the required data volume have been added.* **Chapter "Technical details about the outputs" on page 67** *Information regarding the maximum current of the relays has been added.* **Chapter "Interface channels 1-32" on page 110** *Titles of the parameters for the limits of the measurement range during active scaling have been changed from "Min", "Max" to "0%", "100%" and "Min Modbus", "Max Modbus" to "0% Modbus", "100% Modbus".* **Chapter "Output channels" on page 128** *Examples to explain the burst interval in conjunction with the ext. warmup time have been added* **Chapter "Functions" on page 193** *Information about the required include files has been added.* **Chapter "Mdn functions" on page 197** *Explanation for the Mdn_GetTrigger(), Mdn_WriteLog(), Mdn_CtrlFinish(), Mdn_SetPacked(), Mdn_SetPackedB(), Mdn_GetPacked(), Mdn_GetPackedB(), Mdn_SerialEvent(), Mdn_SerialRx(), Mdn_SerialTx() and Mdn_SerialFinish() functions added* *Sub-chapters "String functions" and "Help functions" added*

Rev.	Date	Changes
04 (3/3)	14.06.2013 (3/3)	**Chapter "Script parsing" on page 204** *Chapter added* **Chapter "XML interface" on page 227** *Chapter added* **Chapter "Log entries and error codes " on page 216** *Explanation regarding the range for customer-specific error codes added* **Chapter "Power supply" on page 227** *Technical data regarding the power supply units added*
05 (1/2)	18.02.2014 (1/2)	**Chapter "Declaration of conformity" on page 13** *Declaration of Conformity updated* **Chapter "Overview" on page 22** *LAN interface added* **Chapter "Functional principle" on page 29** *LAN interface is available as a second communication channel* **Chapter "Functionality of the internal data memory" on page 40** *Chapter added* **Chapter "Behaviour in connection modes when using both communication channels" on page 42** *Chapter added* **Chapter "Procedure in case of connection aborts" on page 43** *Chapter added* **Chapter "ALOHA transmission mode" on page 44** *Description of the network selection and connection process updated.* **Chapter "Ban list" on page 45** *Description of the network selection and connection process updated.* **Chapter "Script parsing" on page 46** *Explanation of the communication process when using the retry function added.* **Chapter "Technical details regarding the energy supply" on page 70** *Chapter added* **Chapter "Interfaces" on page 80** *Description about configuring the script parsing has been adjusted to server version 39.11.* *Example to explain the link between the "Timeout", "Retry" and "Measurement cycle" added* **Chapter "Measurement channels" on page 92** *Explanation of the digital modes revised* *Explanation of the out of range modes for 0-2V and 0-10V mode revised* **Chapter "Calculated channels" on page 124** *Explanation of "delta" channel mode added* **Chapter "Output channels" on page 128** *Example to explain the filter time in conjunction with the ext. warmup time added.* **Chapter "Alarm settings" on page 133** *Explanation of the "Fault alarm" and "Fault warning" added*

Rev.	Date	Changes
05 (2/2)	18.02.2014 (2/2)	**Chapter "Basic setting" on page 134** *Explanation for configuring the communication channel added* **Chapter "General" on page 149** *Source for additional information on the PAWN script language added* **Chapter "Functions" on page 193** *Specification of the include file is now only required for string functions (PAWN API level 2)* **Chapter "Example" on page 200** *Examples adjusted to API level 2* **Chapter "Data record compression" on page 277** *More detailed description of the summation method added* **Chapter "Troubleshooting and repair" on page 213** *Proposed solutions for problems communicating via the LAN interface have been added* **Chapter "Spare parts and accessories" on page 227** *Accessories list revised* **Chapter "Glossary" on page 235** *Explanation of the NaN values added*
06 (1/3)	23.01.2018 (1/3)	**Chapter "Declaration of conformity" on page 13** *Declaration of Conformity updated* **Chapter "Technical data" on page 15** *Specified load for 0...2V mode amended from 10k96 to 10k086* **Chapter "Safety instructions" on page 19** *Safety instructions revised* **Chapter "General safety instructions" on page 20** *Safety instructions revised* **Chapter "Intended use" on page 24** *Information on the manufacturer's liability disclaimer regarding the loss of data has been updated.* **Chapter "Device labelling" on page 25** *Explanation of the symbol, that specifies the country list profile of the SIM chip, added* **Chapter "Obligation of the operator" on page 26** *Reference to the guideline about the minimum health and safety requirements for use of work equipment by workers at work, has been updated* **Chapter "Personnel requirements" on page 26** *Chapter added* **Chapter "Internal processing of the measurement values" on page 31** *Additional explanation on universal inputs that are operated in digital mode, has been added* *Explanation of the control module and record module revised* *Explanation of the setpoint module and scale module (outputs) added*

Rev.	Date	Changes
06 (2/3)	23.01.2018 (2/3)	**Chapter "Installing the myDatalogMUC " on page 53** *Further information on the correct installation added* **Chapter "Technical details about the modbus interfaces (Com1, Com3)" on page 63** *List of the possible access functions for the modbus slave mode added.* **Chapter "Site" on page 79** *Explanation of the "Application template" parameter added* *Explanation for assigning tags added* **Chapter "Control" on page 79** *Description on uploading binary files to the server has been added* *Script templates are no longer supported by server version V39.2.* **Chapter "Measurement channels" on page 92** *Explanation of the difference between the event trigger and level trigger for modes other than "Digital" mode, added* **Chapter "Interface channels 1-32" on page 110** *Explanation of the "Script" basic settings added* *List of the possible access functions for the modbus slave mode added.* *The "0% modbus" and "100% modbus" parameters have been moved from the "Basis" to the "Scaling" tab.* *Explanation of the "Modus" parameter added* *Modbus commands are now decimally encoded.* **Chapter "Trigger" on page 117** *Information added indicating that if "Script" is set as the basic setting for the measurement channel, the triggers cannot be initiated automatically by the system.* *Explanation of the difference between the event trigger and level trigger added* **Chapter "Alarms" on page 116** *Information added indicating that if "Script" is set as the basic setting for the measurement channel, the alarms cannot be triggered automatically by the system.* **Chapter "Interface output channels 1-32" on page 118** *Explanation of the "Script" basic settings added* *The "0% modbus" and "100% modbus" parameters have been moved from the "Basis" to the "Scaling" tab.* *List of the possible access functions for the modbus slave mode added.* *Explanation of the "Modus" parameter added* *List of possible modbus commands revised depending on the data type* *Modbus commands are now decimally encoded.* **Chapter "Calculated channels" on page 124** *Explanation of "Clone" channel mode added* **Chapter "Internal channels" on page 131** *Explanation of the difference between the event trigger and level trigger added* **Chapter "Alarm settings" on page 133** *Explanation of the "Offline alarm" parameter has been supplemented.*

Rev.	Date	Changes
06 (3/3)	23.01.2018 (3/3)	**Chapter "Basic setting" on page 134** *Explanation of the "Report template" parameter has been added* **Chapter "FTP export settings" on page 138** *Explanation of the parameter for entering the time stamp of the last FTP export added* **Chapter "Measurement instrument" on page 138** *Explanation for assigning tags added* *Explanation added indicating that the instrument class can no longer be changed once the instrument is connected to the server* **Chapter ""Customer" area" on page 142** *Explanation about the area, where an image file can be displayed as a "Map" and/or the OpenStreetMaps map can be displayed, has been revised* *Explanation of the "List of tags" added* *Explanation of the search field to filter the customer list added* **Chapter ""Site" area at customer level" on page 144** *Explanation about the area, where an image file can be displayed as a "Map" and/or the OpenStreetMaps map can be displayed, has been revised* *Explanation of the "List of groups" added* **Chapter "Instruction list"** *Chapter removed* **Chapter "XML interface"** *Chapter removed* **Chapter "Pawn script" on page 149** *Chapter revised* *Explanation of the arrays including symbolic indices added* *Explanation of the constants added* *Explanation of the "Mdn_GetChScale", "Mdn_GetChMode", "Mdn_ GetTimezoneOffset", "Mdn_GetAlarmCfg" and "Mdn_ GetTriggerCfg" functions added* *Explanation of the "Mdn_GetTime" and "Mdn_GetDate" functions revised* **Chapter "Messkanäle" on page 154** *Numbers of the measurement channels for the extension module added* **Chapter "Pawn script error codes" on page 180** *Explanation of "SCRIPT_ERR" error code added* *Possible causes for the "AMX_ERR_EXIT" and "AMX_ERR_ MEMACCESS" error codes added* **Chapter "API" on page 207** *Chapter added* **Chapter "Removal/disposal" on page 211** *Chapter revised* **Chapter "Troubleshooting and repair" on page 213** *Possible causes for implausible data at the universal input added* **Chapter "Log entries and error codes " on page 216** *Explanation of the "LAN MAC READ ERROR" log entry added*

Chapter 18 Glossary

Aloha

Connection mode that is specially designed for commissioning. The device maintains a connection to the server for a configurable amount of time and takes a measurement every 3 seconds. During this process, only the internal measurement values (GSM level, voltages, etc.) and the values from the universal inputs (if available) are generated. The determined measurement values are not saved but are just sent to the server for display purposes.

Footprint

The manufacturer's devices are equipped with subscriber identity modules (SIM) ex-works for the purpose of mobile data transmission. The footprint describes the countries and regions, in which a mobile connection is available (see www.microtronics.com/footprint).

NaN value

The myDatanet uses special encoding to display different error statuses in the measurement values, for example. By setting a measurement value to "NaN", it is clearly marked as invalid and is thus not used for any further calculations. In the measurement value graphs, a measurement value that has been set to "NaN" is indicated by an interruption in the graph. When downloading the data, a measurement value set to "NaN" is indicated by an empty data field.

Chapter 19 Contact information

Support & Service:
Microtronics Engineering GmbH
Hauptstrasse 7
3244 Ruprechtshofen
Austria, Europe
Tel. +43 (0)2756 7718023
support@microtronics.com
www.microtronics.com

**Microtronics Engineering GmbH
(Headquarters)**
Hauptstrasse 7
3244 Ruprechtshofen
Austria, Europe
Tel. +43 (0)2756 77180
Fax. +43 (0)2756 7718033
office@microtronics.com
www.microtronics.com

WE LIVE M2M

TÜV-ZERTIFIZIERTE
EN ISO 80079-34
ATEX-Richtlinie
ATEX PRODUKTION

TÜV-ZERTIFIZIERTES
EN ISO 9001
Qualitätsmanagementsystem
MANAGEMENTSYSTEM

TÜV-ZERTIFIZIERTES
EN ISO 14001
Umweltmanagementsystem
MANAGEMENTSYSTEM

TÜV-ZERTIFIZIERTES
EN ISO 50001
Energiemanagementsystem
myDatanet

Microtronics Engineering GmbH | www.microtronics.com