

## KS 98-2 Multi-function unit Engineering manual



A publication of:



**PMA**  
**Prozeß- und Maschinen-Automation GmbH**  
**P.O.Box 310 229 • D-34058 Kassel • Germany**

All rights reserved.

No part of this document may be reproduced or published in any form or by any means without prior written permission from the copyright owner.

## Symbols used on the instrument


 **EU conformity marking**


 **Caution, Follow the operating instructions!**


## Symbols in the text

 Danger of injury

 Danger for the instrument, or of faulty function.

 Danger of destroying electronic components due to electrostatic discharge (ESD).

 Additional information or reference to further sources of information.

 Important hint for avoiding frequent operator faults.

## Liability and warranty

Any information and notes in these operating instructions were composed under consideration of the applicable regulations, the present state of the art and our extensive know-how and experience.

With special versions, additional ordering options or due to the latest technical modifications, the actual scope of delivery may vary from the descriptions and drawings in this manual.

For questions, please, contact the manufacturer.



**Before starting to work with the instrument and before commissioning, in particular, these operating instructions must be read carefully!**

**The manufacturer cannot be held responsible for damage and trouble resulting from failure to comply with the information given in this manual.**

**This product may be subject to change due to improvements of the product features in the course of further development.**

## Copyright

This operating manual should be considered as confidential information, intended only for persons who work with the instrument.

Contraventions are subject to payment of damages. Further claims reserved.

### Content

1.	Operating description .....	10
1.1.	Description .....	10
1.2.	Safety notes .....	11
1.3.	Technical data .....	13
1.4.	Achievements .....	20
1.4.1.	E/A-Module.....	21
1.5.	Mounting .....	22
1.5.1.	Internal switches .....	23
1.5.2.	Retro-fitting and modific. of I/O-ext. (watch connecting diagram).....	23
1.5.3.	I/O extension with CANopen .....	23
1.6.	Electrical connections .....	24
1.6.1.	Safety hints .....	24
1.6.2.	Electromagnetic compatibility.....	24
1.6.3.	Galvanic isolation .....	25
1.6.4.	General connecting diagram .....	26
1.6.5.	Connection Diagram I/O Modules .....	27
1.6.6.	Analog inputs.....	28
1.6.7.	Digital inputs and outputs.....	29
1.7.	Commissioning .....	29
1.8.	Operation .....	30
1.8.1.	Front view .....	30
1.8.2.	Touch features .....	31
1.9.	Menues.....	32
1.9.1.	Short-form dialog.....	32
1.9.2.	Complete dialog.....	33
1.9.3.	Selection of operating pages .....	33
1.9.4.	Language selection.....	34
1.9.5.	Navigation, page selection .....	34
1.9.6.	Adjusting values .....	35
1.10.	Device settings in the main menu .....	36
1.10.1.	Date, Time.....	36
1.10.2.	Device data .....	36
1.10.3.	Online/Offline .....	36
1.10.4.	Calibration.....	36
1.10.5.	Info .....	37
1.10.6.	Status I / O.....	37
1.10.7.	CAN-Status .....	38

## Description

---

1.10.8.	Profibus-Status .....	38
1.10.9.	USB Menü.....	38
1.11.	Operating pages.....	39
1.11.1.	List display .....	39
1.11.2.	Bargraph display.....	39
1.11.3.	Alarm display.....	39
1.11.4.	Graphic trend curve .....	40
1.11.5.	Programmer.....	41
1.11.6.	Controller .....	44
1.11.7.	Cascade controller .....	49
1.12.	Maintenance, test, trouble shooting.....	51
1.12.1.	Cleaning .....	51
1.12.2.	Behaviour in case of trouble .....	51
1.12.3.	Shut-down.....	51
1.12.4.	Default engineering as basic equipment.....	51
2.	Engineering-Tool .....	52
2.1.	Survey .....	52
2.1.1.	Scope of delivery .....	52
2.2.	Installation.....	52
2.2.1.	Hardware and software prerequisites.....	52
2.2.2.	Software installation.....	52
2.2.3.	Licencing .....	52
2.2.4.	Software start.....	53
2.3.	Engineering tool operation.....	54
2.3.1.	Fundamentals of the engineering tool operation .....	54
2.3.2.	Load projects and put them into operation .....	54
2.3.3.	Navigate in the editor.....	54
2.3.4.	Parameterization of function blocks .....	54
2.3.5.	Change / create program logic.....	54
2.3.6.	Function block placement.....	55
2.3.7.	Function block shifting.....	55
2.3.8.	Creating connections.....	55
2.3.9.	Online-operation.....	57
2.3.10.	Trend function.....	57
2.4.	Overview of all menu functions.....	60
2.4.1.	Menu 'File' .....	60
2.4.2.	Menu 'Edit' .....	63
2.4.3.	Menu 'Functions' .....	64
2.4.4.	Menu 'Fixed-funct.' .....	64
2.4.5.	Menu 'Device' .....	65

2.4.6.	Menu 'Online' .....	66
2.4.7.	Menu 'Extra' .....	67
2.4.8.	Menu 'Window' .....	68
2.4.9.	Menu 'Help' .....	69
2.4.10.	Attachment .....	69
3.	Function blocks .....	73
3.1.	Scaling and calculating functions.....	75
3.1.1.	ABSV (absolute value (No. 01)) .....	75
3.1.2.	ADSU ( addition/subtraction (No. 03)) .....	75
3.1.3.	MUDI ( Multiplication / division (No. 05)) .....	76
3.1.4.	SQRT ( square root function (No. 08)) .....	76
3.1.5.	SCAL ( scaling (No. 09)) .....	77
3.1.6.	10EXP (10s exponent (No. 10)) .....	77
3.1.7.	EEXP (e-function (No. 11)) .....	78
3.1.8.	LN (natural logarithm (No. 12)).....	78
3.1.9.	LG10 (10s logarithm (No. 13)).....	79
3.2.	Non-linear functions .....	80
3.2.1.	LINEAR (linearization function (No. 07)) .....	80
3.2.2.	GAP (dead band (No. 20)) .....	82
3.2.3.	CHAR (function generator (No. 21)) .....	83
3.3.	Trigonometric functions .....	84
3.3.1.	SIN (sinus function (No. 80)) .....	84
3.3.2.	COS (cosinus function (No. 81)).....	84
3.3.3.	TAN (tangent function (No. 82)).....	85
3.3.4.	COT (cotangent function (No. 83)).....	86
3.3.5.	ARCSIN (arcus sinus function (No. 84)) .....	87
3.3.6.	ARCCOS (arcus cosinus function (No. 85)) .....	88
3.3.7.	ARCTAN (arcus tangent function (No. 86)).....	89
3.3.8.	ARCCOT (arcus cotangent function (No. 87)).....	89
3.4.	Logic functions .....	90
3.4.1.	AND (UND-gate (Nr. 60)) .....	90
3.4.2.	NOT (inverter (No. 61)).....	90
3.4.3.	OR (OR gate (No. 62)).....	91
3.4.4.	BOUNCE (debouncer (No. 63)).....	92
3.4.5.	EXOR (exclusive OR gate (No. 64)) .....	92
3.4.6.	FLIP (D flipflop (No. 65)).....	93
3.4.7.	MONO (monoflop (No. 66)).....	94
3.4.8.	STEP (step function for sequencing (No. 68)) .....	95
3.4.9.	TONOFF (timer (No. 69)) .....	96
3.5.	Signal converters.....	97

3.5.1.	A2BYTE (data type conversion (No. 02))	97
3.5.2.	ABIN (analog i binary conversion (No. 71))	99
3.5.3.	TRUNC (integer portion (No. 72))	101
3.5.4.	PULS (analog pulse conversion (No. 73))	102
3.5.5.	COUN (up/down counter (No. 74))	104
3.5.6.	MEAN (mean value formation (No. 75))	106
3.6.	Time functions	108
3.6.1.	LEAD ( differentiator (Nr. 50) )	108
3.6.2.	INTE (integrator (No. 51))	110
3.6.3.	LAG 1 (filter (No. 52) )	112
3.6.4.	DELA1 (delay time (No. 53) )	113
3.6.5.	DELA 2 (delay time (No. 54))	114
3.6.6.	FILT (filter with tolerance band (No. 55))	115
3.6.7.	TIMER (timer (No. 67))	116
3.6.8.	TIME 2 (timer (No. 70))	117
3.7.	Selecting and storage	118
3.7.1.	EXTR (extreme value selection (No. 30))	118
3.7.2.	PEAK (peak value memory (No. 31))	119
3.7.3.	TRST (hold amplifier (No. 32) )	120
3.7.4.	SELC (Constant selection (No. 33))	121
3.7.5.	SELD (selection of digital variables - function no. 06))	122
3.7.6.	SELP (parameter selection (No. 34) )	123
3.7.7.	SELV1 (variable selection (No. 35))	124
3.7.8.	SOUT (Selection of output (No. 36))	125
3.7.9.	REZEPT (recipe management (No. 37))	126
3.7.10.	2OF3 ( 2-out-of-3 selection with mean value formation (No. 38))	128
3.7.11.	SELV2 (cascadable selection of variables (No. 39))	130
3.8.	Limit value signalling and limiting	131
3.8.1.	ALLP (alarm and limiting with fixed limits(No. 40))	131
3.8.2.	ALLV (alarm and limiting with variable limits (No. 41))	133
3.8.3.	EQUAL (comparison (No. 42))	135
3.8.4.	VELO (rate-of-change limiting (No. 43))	136
3.8.5.	LIMIT (multiple alarm (No. 44))	137
3.8.6.	ALARM (alarm processing (No. 45))	138
3.9.	Visualization	139
3.9.1.	TEXT (text container with language-dependent selection (No. 79))	139
3.9.2.	VWERT (display / definition of process values (No. 96))	141
3.9.3.	VBAR (bargraph display (No. 97))	146
3.9.4.	VPARA (parameter operation (No. 98))	149
3.9.5.	VTREND (trend display(No. 99))	151

3.10.	Communication.....	154
3.10.1.	L1READ (read level1 data(No. 100)).....	154
3.10.2.	L1WRIT (write level1 data (No. 101)).....	155
3.10.3.	DPREAD (read level1 data via PROFIBUS (No. 102)) .....	156
3.10.4.	DPWRIT (write level1 data via PROFIBUS (No. 103)).....	157
3.10.5.	MBDATA (read and write parameter data via MODBUS - no. 104)) .....	158
3.11.	I/O extensions with CANopen .....	159
3.11.1.	RM 211, RM212 and RM213 basic modules .....	159
3.11.2.	C_RM2x (CANopen fieldbuscoupler RM 201 (No. 14)) .....	160
3.11.3.	RM_DI (RM 200 - (digital input module (No. 15)).....	161
3.11.4.	RM_DO (RM 200 - digital output module (No. 16)).....	161
3.11.5.	RM_AI (RM200 - analog input module (No. 17)).....	162
3.11.6.	RM_AO (RM200 - analog output module (No. 18)).....	164
3.11.7.	RM_DMS (strain gauge module (No. 22)) .....	165
3.12.	KS 98-1- KS 98-1 cross communication (CANopen) .....	167
3.12.1.	CRCV (receive mod. block no's 22,24,26,28 (No.56)).....	167
3.12.2.	CSEND (Send mod. blockno.'s 21, 23, 25, 27 - (No. 57)).....	168
3.13.	Connection of KS 800 and KS 816.....	169
3.13.1.	C_KS8x (KS 800 and KS 816 node function - (No. 58)) .....	170
3.13.2.	KS8x (KS 800/ KS 816 controller function - (No. 59)).....	171
3.14.	Description of KS 98-1 CAN bus extension.....	173
3.14.1.	CPREAD (CAN-PDO read function (No. 88)).....	177
3.14.2.	CPWRIT (CAN-PDO write function (No. 89)).....	178
3.14.3.	CSDO (CAN-SDO function (No. 92)) .....	179
3.15.	Programmer .....	184
3.15.1.	APROG ( analog programmer (No. 24)) / APROGD ( APROG data (No. 25)).....	184
3.15.2.	DPROG ( digital programmer (No. 27)) / DPROGD ( DPROG data(No. 28)).....	202
3.16.	Controller .....	206
3.16.1.	CONTR (Controller with one parameterset (No. 90)).....	206
3.16.2.	CONTR+ (Controller with six parametersets (Nr. 91)) .....	207
3.16.3.	Parameter and configuration for CONTR, CONTR+ .....	209
3.16.4.	Control behaviour .....	211
3.16.5.	Controller characteristics (CONTR und CONTR+) .....	223
3.16.6.	Empirical optimization CONTR / CONTR+ .....	224
3.16.7.	Self-tuning → controller adaptation to the process.....	225
3.16.8.	PIDMA (Control function with particular self-tuning behaviour (Nr. 93)).....	229
3.16.9.	Parameter and configuration for PIDMA .....	232
3.16.10.	Controller characteristics and self-tuning with PIDMA .....	234
3.16.11.	Controller applications: .....	238
3.16.12.	Setpoint functions .....	242

3.16.13.	Process value calculation .....	247
3.16.14.	Small controller ABC .....	252
3.17.	Inputs / outputs .....	255
3.17.1.	UNI_IN (analog universal input-Modul U).....	256
3.17.2.	TC_IN (analog input card TC, mV, mA) .....	262
3.17.3.	R_IN (analog input card) .....	264
3.17.4.	U_IN (analog input card -50...1500mV, 0...10V).....	266
3.17.5.	TPS_IN .....	267
3.17.6.	I_OUT (analog output card 0/4...20mA, +/- 20mA).....	269
3.17.7.	U_OUT (analog output card 0/2...10V, +/- 10V).....	270
3.17.8.	REL_OUT (Relais output) .....	271
3.17.9.	SSR_OUT (Solid-State-Relais Ausgang).....	272
3.17.10.	DIDO (digital input/output card).....	273
3.17.11.	DINPUT (digital inputs (Nr. 121)).....	274
3.17.12.	DIGOUT (digital outputs (No. 122)) .....	275
3.18.	Additional functions .....	276
3.18.1.	LED (LED display (No. 123)) .....	276
3.18.2.	CONST (constant function (No. 126)).....	277
3.18.3.	INFO (information function (No. 124)).....	278
3.18.4.	STATUS (status function (No. 125)) .....	279
3.18.5.	CALLPG (Function for calling up an operating page (no. 127)).....	282
3.18.6.	SAFE (safety function (Nr. 94) ) .....	283
3.18.7.	VALARM (display of all alarms on alarm operating pages (function no. 109)) .....	284
3.18.8.	F_Inp (Frequenz-/ Zählereingang) .....	286
3.19.	Function management.....	287
3.19.1.	Memory requirement and calculation time .....	287
3.19.2.	Sampling intervals .....	288
3.19.3.	EEPROM data.....	288
3.20.	Examples .....	289
3.20.1.	Useful small engineerings.....	289
3.20.2.	Controller applications .....	290
3.20.3.	Programmer fragments.....	290



# Foreword

This manual consists of three descriptive units:


1. *Operating instructions*
2. *Engineeringtool description*
3. *Function block description*


Section 1 holds the required information for identification, mounting, connection and electrical commissioning of the unit under consideration of safety notes of the application and environmental conditions.

The basic principles of operation are explained: Controls and indicators, menu structure and navigation with the cursor, selection of sub-menus and properties as well as adjustment of e.g. s and parameters.

Section II comprises the handling of the engineering-tool, the building of a simple engineering and transmission to the KS 98-2.

Section 3 presents the particular function blocks in detail.

 For functional commissioning, additional descriptions are required; please, order them separately or load them from the PMA homepage: [www.pma-online.de](http://www.pma-online.de).

 As the functions provided in KS 98-2 are composed individually for each application using an Engineering Tool ET/KS 98, entire comprehension of the operating functions requires the relevant Project description with the Engineering

Supplementary documentation:

PROFIBUS-protocoll (GB) 9499-040-82811

ISO 1745-protocoll (GB) 9499-040-82911

# 1. Operating description

## 1.1. Description



The device is a compact automation unit.  
The function can be freely structured via function blocks.

Dependent on version, the basic unit (standard) contains analog and digital outputs as well as relays. Additional inputs and outputs are available as plug-in modules. The basic unit has at least 2 slots. The number of slots can be increased by up to 12 using additional cards (for terminal blocks B and C). An additional optional communication card provides interfaces to communicate with other devices and systems available.

The instrument is a compact automation unit the function of which can be configured and linked together freely by means of function blocks. Each unit contains a comprehensive function library.  
The function blocks can with an engineering tool selected, configured, parameterized and linked together.

I.e. complex mathematical calculations, multi-channel control structures and sequence controllers can be realized in a single unit.

Indication of various operating pages with max. 10 lines is via a full-graphic color display (320 x 240 dots) e.g. numeric input and output of analog and digital signals, values and parameters as well as full-graphic display of barographs, controllers, programmers and trends.

The display color can be switched over dependent on events, or by operation (engineering).

## 1.2. Safety notes

This section provides a survey of all important safety aspects: optimum protection of personnel and safe, trouble-free operation of the instrument.

Additionally, the individual chapters include specific safety notes for prevention of immediate hazards, which are marked with symbols. Moreover, the hints and warnings given on labels and inscriptions on the instruments must be followed and kept in readable condition continuously.

### General

Software and hardware are programmed or developed in compliance with the state of the art applicable at the time of development, and considered as safe.

Before starting to work, any person in charge of work at the product must have read and understood the operating instructions.

The plant owner is recommended to request evidence for knowledge of the operating instructions by the personnel.

### Correct use for intended application

The operating safety is only ensured when using the products correctly for the intended application. The instrument can be used as a multiple function controller for open and closed control loops in industrial areas within the limits of the specified technical data and environmental conditions.

Any application beyond these limits is prohibited and considered as non-compliant.

Claims of any kind against the manufacturer and /or his lawful agents, for damage resulting from non-compliant use of the instrument are precluded, liability is limited to the user.

### User responsibility

The user is responsible:

- for keeping the operating manual in the immediate vicinity of the instrument and always accessible for the operating personnel.

- for using the instrument only in technically perfect and safe condition.

- The operator of the system is recommended to have the operating personnel demonstrably confirm their knowledge of the operating instructions.

Apart from the work safety notes given in these operating instructions, compliance with the generally applicable regulations for safety, accident prevention and environment protection is compulsory.

The user and the personnel authorized by the user are responsible for perfect functioning of the instrument and for clear definition of competences related to instrument operation and maintenance. The information in the operating manual must be followed completely and without restrictions!

The user is responsible that the instrument is operated only by trained and authorized persons. Maintenance and repair may be done only by trained and specialized persons who are familiar with the related hazards.

Operation and maintenance of the instrument are limited to reliable persons. Any acts susceptible to impair the safety of persons or of the environment have to be omitted. Any persons who are under effect of drugs, alcohol or medication affecting reaction are precluded from operation of the instrument.

### Instrument Safety

This instrument was built and tested according to VDE 0411 / EN61010-1 and was shipped in safe condition. The unit was tested before delivery and has passed the tests required in the test plan.

In order to maintain this condition and to ensure safe operation, the user must follow the hints and warnings given in these safety notes and operating instructions.

The unit is intended exclusively for use as a measuring and control instrument in technical installations.

The insulation meets standard EN 61010-1 with the values for overvoltage category, degree of contamination, operating voltage range and protection class specified in the operating instructions / data sheet.

The instrument may be operated within the specified environmental conditions (see data sheet) without impairing its safety.

The instrument is intended for mounting in an enclosure. Its contact safety is ensured by installation in a housing or switch cabinet.

### Unpacking the instrument

Remove instrument and accessories from the packing. Enclosed standard accessories:

- operating notes or operating instructions
- fixing elements.

Check, if the shipment is correct and complete and if the instrument was damaged by improper handling during transport and storage.



#### Warning!

If the instrument is so heavily damaged that safe operation seems impossible, the instrument must not be taken into operation.

We recommend to keep the original packing for shipment in case of maintenance or repair.

### Mounting

Mounting is done in dustfree and dry rooms.

The ambient temperature at the place of installation must not exceed the permissible limits for specified accuracy given in the technical data. When mounting several units with high packing density, sufficient heat dissipation to ensure perfect operation is required.

For installation of the unit, use the fixing clamps delivered with the unit. The sealing devices (e.g. sealing ring) required for the relevant protection type must also be fitted.

### Electrical connections

All electrical wiring must conform to local standards (e.g. VDE 0100 in Germany). The input leads must be kept separate from signal and mains leads.

The protective earth must be connected to the relevant terminal (in the instrument carrier). The cable screening must be connected to the terminal for grounded measurement. In order to prevent stray electric interference, we recommend using twisted and screened input leads.

The electrical connections must be made according to the relevant connecting diagrams.

See electrical safety, page 241.6.4

### Electrical safety

The insulation of the instrument meets standard EN 61 010-1 (VDE0411-1) with contamination degree 2, overvoltage category II, working voltage 300 V r.m.s. and protection class I.

### Commissioning

Before instrument switch- on, ensure that the rules given below were followed:

Ensure that the supply voltage corresponds to the specification on the type label.

All covers required for contact safety must be fitted.

Before instrument switch- on, check if other equipment and / or facilities connected in the same signal loop is / are not affected. If necessary, appropriate measures must be taken.

The instrument must be operated only when mounted in its enclosure.

### Operation

Switch on the supply voltage. The instrument is now ready for operation. If necessary, a warm-up time of approx. 1.5 min. should be taken into account.

### Shut- down

For permanent shut- down, disconnect the instrument from all voltage sources and protect it against accidental operation.



Before instrument switch- off, check that other equipment and / or facilities connected in the same signal loop is / are not affected. If necessary, appropriate measures must be taken.

### Maintenance and modification

The instrument needs no particular maintenance.

Modifications, maintenance and repair may be carried out only by trained, authorized persons. For this, the user is invited to contact the service.

For correct adjustment of wire-hook switch (page 23) and for installation of modular option cards, the unit must be withdrawn from the housing.



**Warning!**

When opening the instruments, or when removing covers or components, live parts or terminals can be exposed.

Before carrying out such work, the instrument must be disconnected from all voltage sources.

After completing such work, re- shut the instrument and re-fit all covers and components. Check, if the specifications on the type label are still correct, and change them, if necessary.

### Explosion protection

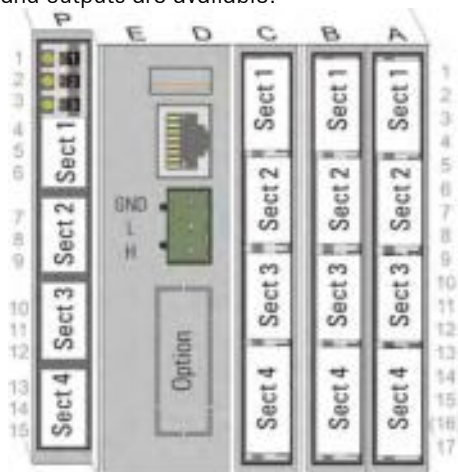
Non-intrinsically safe instruments must not be operated in explosion-hazarded areas. Moreover, the output and input circuits of the instrument / instrument carrier must not lead into explosion-hazarded areas.

## 1.3. Technical data

### In-/Outputs Base Unit

#### Arrangement of In- and Outputs

Depending on the version and option, the following inputs and outputs are available:



#### Analog Inputs

##### Universal Input (Terminal A, Sect. 4)

Resolution: 16bit; Measurement cycle: 100ms

##### Thermocouples

According to DIN IEC 584

Typ	Range	Error	Resolution
L <sup>1)</sup>	-200...900°C	≤ 2K	0,05 K
J <sup>1)</sup>	-200...900°C	≤ 2K	0,05 K
K <sup>1)</sup>	-200...1350°C	≤ 2K	0,072 K
N <sup>1)</sup>	-200...1300°C	≤ 2K	0,08 K
S	-50...1760°C	≤ 3K	0,275 K
R	-50...1760°C	≤ 3K	0,244 K
B <sup>2)</sup>	(25)400...1820°C	≤ 3K	0,132 K
T <sup>1)</sup>	-200...400°C	≤ 2K	0,056 K
C <sup>3)</sup>	0...2300°C	≤ 2K	0,18 K
E <sup>1)</sup>	-200...900°C	≤ 2K	0,038 K

<sup>1)</sup> accuracy valid from -100 °C

<sup>1)</sup> accuracy valid from 400 °C

<sup>2)</sup> W5Re/W26Re

Temperature linear in °C or °F)

Input resistance: ≥ 1 MΩ

Cold junction compensation internal/external

##### Sensor monitoring:

Current through sensor ≤ 1 μA

Reverse polarity detection triggered at 30 °C below start of range. Sensor status information is available to be used in the application program.

##### Influence of internal CJC

≤ 0,5 K per 10 K terminal temperature

##### External CJC

selectable: 0...60 °C bzw. 32...140 °F

##### Resistance thermometer

Pt 100 DIN IEC 751 and Temperature difference 2 x Pt 100

Range	Error	Resolution
-200,0...250,0 °C	≤ 0.5K	0,024 K
-200,0...850,0 °C	≤ 1.0K	0,05 K
2x -200,0...250,0 °C	≤ 0.5K	0,024 K
2x -200,0...250,0 °C	≤ 0.1K	0,05 K

Linearisierung in °C oder °F

3-wire connection

Lead resistance ≤ 30 Ω per lead

Sensor current ≤ 1 mA

Input circuit monitoring for sensor/lead break and short circuit.

##### Potentiometers

Range	Error	Resolution
0...500 Ω <sup>1)</sup>	≤ 0.1 %	≤ 0.02Ω

<sup>1)</sup>Rgesamt inkl. 2 x RL

Resistance linear

Sensor current ≤ 1 mA

Adjustment/scaling with sensor connected.

Input circuit monitoring for sensor/lead break and short circuit.

##### Resistance measurement

Range	Error	Resolution
0...250 Ω	≤ 0.25Ω	≤ 0.01Ω
0...500 Ω	≤ 0.5Ω	≤ 0.02Ω

##### Direct current 0/4...20 mA

Range	Error	Resolution
0/4...20 mA	≤ 0.1 %	≤ 0.8μA

Input resistance: 5 Ω

Input circuit monitoring 4...20 mA: I ≤ 2 mA

##### Direct voltage

Range	Error	Resolution
0/2...10 V	≤ 0.1 %	≤ 0.4mV

Input resistance ≥ 50 kΩ

##### Transmitter-Supply

to energize a 2-wire transmitter Galvanically isolated,

Short-circuit proof,

Output: 22 mA / ≥ 17,5 V

## **Digital Inputs**

### **Logic Inputs (Terminal A, Sect. 1)**

#### **Opto-coupler**

Nominal voltage 24 V DC external

Residual ripple:  $\leq 5\%_{pp}$

Current sink (IEC 61131 Typ 1)

Logic „0“ = -3...5 V

Logic „1“ = 15...30 V

Current approx. 6 mA

Galvanic isolation or connections see section 1.6.3 page 25

## **Outputs**

### **Outputs (Terminal P, Sect. 1.4)**

Depending on version 4 Relays or 2 Relays and 2 option module slots

#### **Relay outputs**

Relays have potential free change-over contacts

Contact rating:

Max. 500 VA, 250 V, 2 A at 48...62 Hz,  $\cos\varphi \geq 0$ ,

Min. 12 V, 10 mA AC/DC

Switching cycles:

electrical for  $I = 1A/2A$  (resistive load)

$\geq 800.000 / 500.000$  at  $\sim 250V$ .

If the relays operate external contactors, these must be fitted with RC snubber circuits to prevent excessive switch-off voltage peaks! Varistor protection is not recommended!

## **Modular I/O Extension**

### **Option slots base unit**

#### **Option slots (Terminal A, Sect. 2.3)**

Option slots for sections A2 and A3 are providing 4 terminals each and can support all available option modules. Option modules are galvanically isolated from the base unit. Details of available modules see chapter „In-/Output Extension Modules“.

#### **Option slots (Terminal P, Sect. 3.4)**

##### **(optional instead of relays)**

Option slots for sections P3 and P4 are providing 3 terminals each. They support input modules for standard signals (0/4...20mA, 0/2...10V) and SSR driver modules. Dual channel modules share a common reference terminal. Option modules are galvanically isolated from the base unit. Details of available modules see chapter „In-/Output Extension Modules“.

## **Option module carrier board for terminals B, C**

### **Option slots (Terminal B)**

Behind terminal strip B an optional module carrier board with 4 slots can be installed.

All module types are supported. Each module is galvanically isolated from the rest of the unit. Details of available modules see chapter „In-/Output Extension Modules“

### **Option slots (Terminal C)**

Same specification as terminal B

## **Digital In-/Output board for terminals B, C**

### **Control inputs di1...di10**

Current sink (IEC 61131 Type 1)

Nominal voltage 24 V DC external

Residual ripple:  $\leq 5\%_{ss}$

Logic „0“ = -3...5 V

Logic „1“ = 15...30 V

Current approx. 6 mA

### **Control outputs do1...do4**

Galvanically isolated opto-coupler outputs. Grounded load (common positive control voltage)

Switching capacity: 18...32 VDC;  $\leq 70mA$

Internal voltage drop:  $\leq 0,7V$

Refresh-Rate: 100 ms

Protective circuit: thermal against short circuit; Overload cut-off.

Nominal voltage 24 V DC external

Residual ripple  $\leq 5\%_{ss}$

### **Limitations to take into account**

To avoid inadmissible self-heating the number of output extension modules is limited. The following rule applies:

- A maximum of 6 current outputs or transmitter power supplies are allowed per unit!

Exceedances are checked by the engineering tool.

## In-/Output Extension Modules

### Analog Inputs

#### U-Module

Universal input module

No of channels: 1

The technical design corresponds to the universal input of the basic unit.

#### R-Module

RTD module (resistance thermometer)

No of channels: 2 (with 3- or 4-wire- connection just one).

Type of sensor can be selected separately for each channel!

Sensor current:  $\leq 0,25$  mA

#### Resistance thermometers

Connection: 2-, 3- or 4-wire

Typ	Range	Error	Resolution
Pt100	-200...850°C	$\leq 1$ K	0,071
Pt100	-200...100°C	$\leq 0,5$ K	0,022
Pt1000	-200...850°C	$\leq 1$ K	0,071
Pt1000	-200...100°C	$\leq 0,5$ K	0,022
Ni100	-60...180°C	$\leq 1$ K	0,039
Ni1000	-60...180°C	$\leq 0,5$ K	0,039

Linearization: in °C or °F

Lead resistance

Pt (-200...850°C):  $\leq 30$   $\Omega$  per lead

Pt (-200...100°C), Ni:  $\leq 10$   $\Omega$  per lead

Lead resistance compensation:

not necessary with 3- and 4-wire connection.

For 2-wire connection with short-circuited sensor via the front user interface.

Influence of lead resistance:

negligible with 3 or 4-wire connection

Input circuit monitoring for break of sensor or lead and short circuit.

Short circuit: reacts at 20K below measurement range

#### Resistance measurement / Potentiometers

2-, 3- or 4-wire connection

Potentiometer 2-wire connection

Range	Error	Resolution
0...160 $\Omega$	$\leq 1\%$	0,012
0...450 $\Omega$	$\leq 1\%$	0,025
0...1600 $\Omega$	$\leq 1\%$	0,089
0...4500 $\Omega$	$\leq 1\%$	0,025

Characteristic linear

Cable compensation or Calibration (0%/100%) can be carried out via the user interface with sensor connected.

- 0% calibration for 2-wire resistor measurement
- 0% and 100% calibration for potentiometer

Influence of lead resistance:

negligible with 3 or 4-wire connection.

Input circuit monitoring for break of sensor or lead and short circuit

#### T-Module

Thermo coupler module (TC, mV, mA)

No of channels: 2 (Differential input).

Type of sensor can be selected separately for each channel!

#### Thermocouples

According to DIN IEC 60584

(not Typ L, W(C) und D)

Typ	Range	Error	Resolution
L <sup>1)</sup>	-200...900°C	$\leq 2$ K	0,080
J <sup>1)</sup>	-200...900°C	$\leq 2$ K	0,082
K <sup>1)</sup>	-200...1350°C	$\leq 2$ K	0,114
N <sup>1)</sup>	-200...1300°C	$\leq 2$ K	0,129
S	-50...1760°C	$\leq 3$ K	0,132
R	-50...1760°C	$\leq 3$ K	0,117
B <sup>2)</sup>	(0) 400...1820°C	$\leq 3$ K	0,184
T <sup>1)</sup>	-200...400°C	$\leq 2$ K	0,031
C <sup>3)</sup>	0...2300°C	$\leq 2$ K	0,277
D	0...2300°C	$\leq 2$ K	0,260
E <sup>1)</sup>	-200...900°C	$\leq 2$ K	0,063

<sup>1)</sup> accuracy valid from -100°C

<sup>2)</sup> accuracy valid from 400°C

<sup>3)</sup> C(W) W5RE/W26Re

Linearization in °C or °F

Linearity error: negligible

Input resistance:  $\geq 1$  M $\Omega$

Internal temperature compensation (CJC)

Error:  $\leq 0,5$ K/10K

External JCC possible: 0...60 °C or 32...140 °F

Effect of source resistance: 1mV/k $\Omega$

Sensor monitoring:

Sensor current:  $\leq 1$   $\mu$ A

Reverse polarity detection triggers at 30K below range min.

#### mV- Input

Range	Error	Resolution
0...30 mV	$\leq 45$ $\mu$ V	1,7 $\mu$ V
0...100 mV	$\leq 150$ $\mu$ V	5,6 $\mu$ V
0...300 mV	$\leq 450$ $\mu$ V	17 $\mu$ V

Input resistance:  $\geq 1$  M $\Omega$

Sensor break monitoring: built in Sensor current:  $\leq 1$   $\mu$ A

### ***mA- Input***

Range	Error	Resolution
0/4...20 mA	$\leq 40 \mu\text{A}$	2 $\mu\text{A}$

Input resistance: 5  $\Omega$

Sensor alarm:  $\ll 2 \text{ mA}$  (with 4...20 mA)

Over range alarm:  $\gg 22 \text{ mA}$

### ***V-Module***

High impedance voltage input module

No. of channels: 2

Range can be selected separately for each channel!

Range	Error	Resolution
-50...1500 mV	$\leq 1,5 \text{ mV}$	0,09
0...10 V	$\leq 10 \text{ mV}$	0,56

Voltage linear characteristic

Input resistance:  $\gg 1 \text{ G}\Omega$

Effect of source resistance: 0,25mV/M $\Omega$

Sensor monitoring: not available

### ***P-Modul***

Input module with transmitter supply

No of channels: 1

The technical design of the input corresponds to the T-Module

### ***Transmitter-Supply***

to energize a 2-wire transmitter or up to 4 opto-coupler inputs.

Galvanically isolated, Short-circuit proof,

Output: 22 mA /  $\geq 17,5 \text{ V}$

### ***Analog Outputs***

#### **L-Module**

Linear output module

No of channels: 2

Resolution: 16 Bit

Refresh-Rate: 100ms

Signal ranges: 0/4...20mA, -20...20mA

(configurable by channel)

Resolution: approx. 5  $\mu\text{A}/\text{Digit}$

Error:  $\leq 0,2\%$

Load:  $\leq 500 \Omega$  /  $\leq 150 \Omega$  (selectable)

Influence of load:  $\leq 0,05\%/100 \Omega$

#### ***Used as logic signal***

0 /  $\leq 20 \text{ mA}$

#### ***B-Module***

Bipolar linear output module

No of channels: 2

Resolution: 16 Bit

Refresh-Rate: 100ms

Signal ranges: 0/2...10V, -10...10V

(configurable by channel)

Resolution: approx. 5 mV/Digit

Error:  $\leq 0,2\%$

Load:  $\geq 2 \text{ k}\Omega$

Influence of load:  $\leq 0,05\%/100 \Omega$

#### ***Used as logic signal***

0 /  $\geq 10 \text{ V}$

### ***Digital In-/Outputs***

#### **D-Module**

Digital I/O module

No of channels: 2 (configurable as input or output per channel)

Reverse polarity protection.

#### ***Input***

Current sink IEC 61131 Type 1)

Nominal voltage 24 V DC external

Residual ripple:  $\leq 5\%_{pp}$

Logic „0“: -3...5V

Logic „1“: 15...30V

Cycle time: 100 ms

Galvanically isolated

Input resistance: 5  $\text{k}\Omega$

#### ***Output***

Grounded load (common positive control voltage)

Switching capacity: 18...32 VDC;  $\leq 70 \text{ mA}$

Internal voltage drop:  $\leq 0,7 \text{ V}$

Refresh-Rate: 100 ms

Galvanically isolated

Protective circuit: thermal against short circuit; Overload cut-off.

Nominal voltage 24 V DC external

Residual ripple  $\leq 5\%_{ss}$

#### **A-Module**

SSR driver module

No of channels: 2

Logic „0“: 0V

Logic „1“:  $\geq 10 \text{ V}$

Load:  $\geq 500 \Omega$

## Galvanic Separation

Galvanically isolated areas are visualized in the diagram underneath. In general, each of the I/O modules is galvanically isolated from the rest of the unit. Channels inside a module are not separated.

#### ***Signal- and measurement circuits***

Functional isolation up to a voltage of 33V<sub>AC</sub>/70V<sub>DC</sub> against each other and against ground (according EN 61010-1).



### Mains circuits 90...250 VAC, 24V UC

Safety isolation up to a voltage of 3kV against each other and against ground (according EN 61010-1).

### Remote I/O-Extension

Detailed technical data and functional descriptions of remote I/O systems can be found in the related documents.

### CAN Interface (CANopen)

#### Transmission speeds:

Com. speed	max. cable length
10 kbit/s	1200 m
20 kbit/s	1000 m
50 kbit/s	1000 m
100 kbit/s	500 m
125 kbit/s	250 m
250 kbit/s	250 m
500 kbit/s	100 m
800 kbit/s	50 m
1000 kbit/s	25 m

#### Termination resistor

Internal resistor connectable by switch

#### Transmission mode:

cyclic

#### Error detection:

Automatic node monitoring  
("node guarding").

#### Addressing:

KS 98-X: 1...24 (Default =1)

RM 200: 2...42 (Default =32)

#### Refresh times:

Depending on the selected transmission speed and the number of CAN-nodes connected.

- RM 200: typical 100ms
- Cross communication:  $\geq 200$ ms

#### Maximum Setup RM 200

$\leq 16$  Analog Inputs and  $\leq 16$  Analog Outputs per RM200-Rack!

Digital I/O is only limited by the size of the rack.

Examples: 72 digital In- /Outputs (without analog modules!), or 16 analog Inputs plus 16 analog Outputs plus 8 digital In- /Outputs.

### Engineering Tools

KS98-2 units can be programmed and maintained with the following tools:

ET/KS98: from Version 7.0

SIM/KS98-2: from Version 1.0

OEM/KS98-2: from Version 1.0

### Front Interface (Standard)

The front accessible USB interface uses a standard USB cable.

It provides access for the programming and diagnosis tool ET/KS98-2 even with the unit not connected to power.

### Fieldbus Interfaces (Optional)

#### RS485-Module

Galvanically isolated RS 485

#### Anzahl der Regler pro Bus

max 32 interface modules without repeater.

#### PROFIBUS-DP Module

According to EN 50170 Vol. 2 (DIN 19245 T3)

Read/write access to all process values parameters and configuration data.

#### Configurable process data modules

Max. 4 DPREAD and 4 DPWRIT functions with six analog and sixteen logic variables each can be selected with the Engineering Tool. By suitable internal connections with inputs and outputs of these functions, any internal signal can be routed to the PROFIBUS-DP interface.

The parameter channel provides non-cyclical access to all parameters and configuration data.

Modul	DPREAD	DPWRIT	Parameterchannel
a	1	1	-
b	1	1	x
c	2	2	x
d	3	3	x
e	4	4	x

#### Data format

Values are transmitted using the IEEE-format (REAL) or in a 16-Bit-fixpoint notation (FIX) with one decimal digit (configurable).

#### I/O memory requirements (Byte)

Modul	Read		Write	
	FIX	REAL	FIX	REAL
a	18	26	18	26
b	26	34	26	34
c	44	60	44	60
d	62	86	62	86
e	80	112	80	112

#### Diagnosis/behavior on error

The functions DPREAD and DPWRIT are providing status outputs to indicate error conditions.

### **Transmission speeds and cable lengths**

Automatic speed detection

Speed	max. cable length
9,6 kbit/s	1200 m
187,5 kbit/s	1000 m
500 kbit/s	400 m
1,5 Mbit/s	200 m
12 Mbit/s	100 m

### **Addresses**

0...126 (Factory setting: 126)

Remote addressing supported.

### **Other functions**

Sync and Freeze

### **Connection**

9pin. Sub-D connector

### **Termination resistor**

By selecting a corresponding connector.

### **Cable**

according EN 50170 Vol. 2 (DIN 19 245T3)

### **Accessories**

Engineering Set KS98/PROFIBUS consisting of:

- GSD-file, Type-file
- PROFIBUS-Manual
- Function blocks for S5 / S7

## Display

3,5" color TFT display with

LED backlight

Resolution: 320 x 240 (QVGA)

Capacitive Touch

## Power Supply

depending on order code:

### **AC Supply**

Voltage: 90...250 VAC (48...62 Hz)

Power consumption: approx. 18 VA

(fully equipped)

### **Low voltage supply 24 V UC**

AC voltage: 20.4...26.4 VAC (48...62 Hz)

DC voltage: 18...31 VDC

Power consumption: approx. 18 VA

(fully equipped)

### **Behavior with power failure**

### **User program, configuration, parameter and active setpoints**

Are permanently stored in EEPROM

### **Working data of functions (Programmer, Integrator, Counters, ...**

Stored in a capacitor buffered RAM  
(typically >> 15 Minutes).

### **Real Time Clock**

Backed up with a Lithium battery.

## Environmental Conditions

### **Protection**

Front: IP 65

Housing: IP 20

Terminals: IP 00

according DIN EN 60529 (VDE 0470-1)

### **Ambient Temperature range**

Operation: 0...55 °C

Storage/Transport: -20...60 °C

Humidity: ≤ 75% RH yearly average,  
non-condensing

### **Temperature influence**

Reference temperature 25°C

Temperature influence << 0.05 %/ 10 K

### **Shock and vibration**

Vibration test according to DIN EN 60068-2-6

Frequency: 10...150 Hz

Unit in operation: 1 g / 0,075 mm,

Unit not in operation.: 2 g / 0,15 mm

Shock test according to DIN EN 60068-2-27

Shock: 15 g,

Duration: 11 ms

## Electromagnetic Compatibility

Complies with EN 61326-1

"continuous, non-monitored operation"

### **Safety**

According EN 61010-1 (VDE 0411-1)

- Overvoltage category II
- Contamination class 2
- Working voltage range 300 V
- Protection class II

### General

#### **Housing**

Plug-in unit, inserted from the front

Material: Makrolon 9415, flame retardant, self-extinguishing. Flammability class:: UL 94 V0

Weight:

approx. 750 g (fully equipped)

#### **Mounting**

Panel mounting with 2 fixing elements at top and bottom. Close mounting possible Orientation as desired.

#### **Electrical connections**

depending on order code:

- Screw terminals for conductor cross-section 0,5-2,5mm<sup>2</sup>
- Flat pin connectors 1 x 6,3mm or 2 x 2,8mm according DIN 46244

#### **CE Compliance**

Meets the European Directives regarding „Electromagnetic Compatibility“ and „Low-voltage equipment“

#### **DIN EN 14597 Certificate**

The device is certified to be used as temperature control and limiting equipment according to DIN EN 14597.

#### **UL & cUL Compliance**

(Type 1, indoor use)

File: E 208286

For compliance with UL certificate, the following information must be considered:

- Use only Screw Terminal variant
- Use only 60/75°C copper (Cu) conductors.
- Tighten the terminal-screws with a torque of 0.5 – 0.6 Nm.
- The instrument shall be mounted on a flat surface of a "Type 1 Enclosure" for "Indoor use" only.
- Ambient temperature:  $\leq 50\text{ }^{\circ}\text{C}$
- Power supply:  $\leq 250\text{ VAC}$
- Max. ratings of relay contacts:  
250 VAC, 2 A, 500 W (resistive)  
250 VAC, 2 A, 360 VA (inductive)

#### **In the box**

- Device according to order code
- Concise manual (DE/EN/FR)
- 4 fixing clamps

### Supporting Software

#### **Engineering Tool ET/KS98-2**

Graphical function block editor for programming and maintenance of KS98-2 units

#### **Simulation SIM/KS98-2**

Program to simulate KS 98-2 on a PC screen with full functionality of the device plus:

- Simulation of In- and Outputs
- Trend diagrams
- „Turbo“-Mode (time laps)

#### **USB-Cable**

To connect PC with programming utility to the unit. (KS 98-2 Front USB Interface)

### Delivered Condition

Every unit is coming with a test-engineering that allows to check the in- and outputs of the basic unit.

## 1.4. Achievements

The versions of the device result from the combination of different variants according to the following scheme.

	KS98	-	2	x	x	-	x	x	x	x	x	-	x	0	0
<b>Base Models</b>															
Universal Input, 2 digital Inputs, TPS, RTC				↓											
with Fast-On Terminals				0											
with Screw Terminals				1											
<b>Power Supply &amp; Outputs [P]</b>				↓											
90...250V (2 Relays, 2 option module slots)				0											
24V UC (2 Relays, 2 Option Slots)				1											
90...250V (4 Relays)				2											
24V UC (4 Relays)				3											
<b>Communication Options [D]</b>							↓								
none / for stand-alone applications							0								
Standard Interfaces (Ethernet/USB) and CAN for Remote I/O							1								
Standard Interfaces plus RS485 / Modbus							2								
Standard Interfaces plus Profibus							3								
Standard Interfaces plus Profinet							4	↓							
<b>Options</b>															
none							0								
Datalogger							1								
<b>I/O Extensions [B]</b>								↓							
not fitted								0							
Digital-I/O Extension (10 DI, 4 DO)								1							
Modular Extension (4 option module slots)								2							
<b>I/O Extensions [C]</b>									↓						
not fitted									0						
Digital-I/O Extension (10 DI, 4 DO)									1						
Modular Extension (4 option module slots)									2						
<b>Configuration</b>										↓					
Default settings; No option modules installed										0					
Default settings; Option modules according to additional order line										1					
Preset to specification; No option modules installed										8					
Preset to specification; Option modules according to additional order line										9					
<b>Certification</b>													↓		
Standard (CE certified)													0		
UL / cUL certified													U		
DIN3440 / EN 14597													D		

### 1.4.1. E/A-Module

Extended order code for factory installed option modules

Position of digit within the order code defines the positioning of the module and assignment to output terminals Positions at terminal strips B and C require extension cards

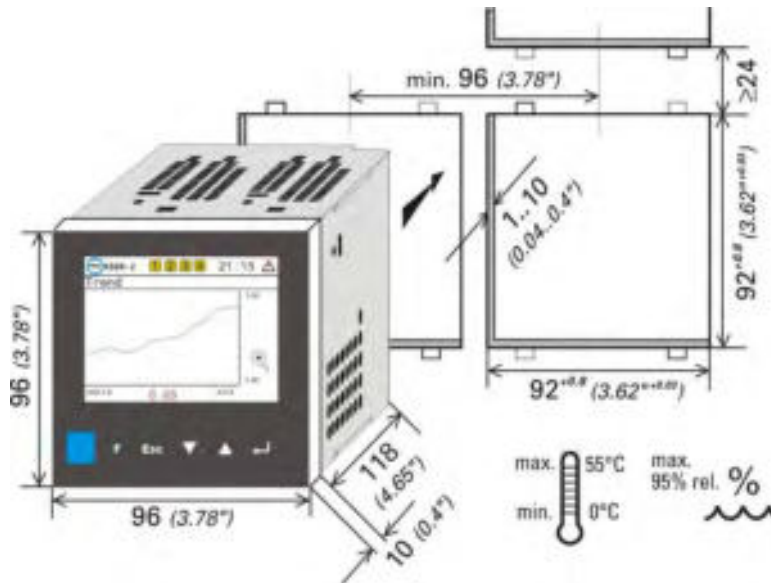
Separate  
module  
orders:

[illegible]

## Modules and possible positions

[illegible]

## 1.5. Mounting



The figure shows the necessary dimensions of the outbreak in the Cabinet wall and the minimum distances to additional devices. For installation, the device is inserted into the outbreak of the Panel or the control panel door from outside. A rubber seal is molded on the front frame of the appliance. This rubber seal must be intact.

The delivery scope includes four fasteners. These be inserted from the inside of Panel on the device, each 2 top and bottom. The threaded rods of fixtures are then screwed from the inside against the Cabinet.

- ⚠ The instrument is mounted by means of four fixing clamps. Insert the module firmly and mount it safely by means of the locking screw.
- ⚠ UL/cUL: Note section "Certificates and approvals"!
- ⚠ Ensure tightness!

**! A rubber seal is fitted on the rear of the instrument front panel (in mounting direction). This rubber seal must be in perfect condition, flush and cover the cut-out edges completely to ensure tightness. Only then is the tightness guaranteed!**

### UL & cUL



**For compliance with cUus certificate, the technical data at the beginning must be taken into account (see technical data, page Fehler! Textmarke nicht definiert.)**

### 1.5.1. Internal switches

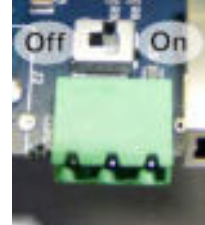


The unit contains electrostatically sensitive components. Comply with rules for protection against ESD during mounting.

Bus termination CAN-Bus:

Both ends of the CAN bus must be terminated. To get a bus termination on the KS98-2, an internal terminating resistor can be connected via a slide switch behind the CAN connector plug.

Position of the switch „ON“



### 1.5.2. Retro-fitting and modif. of I/O-ext. (watch connecting diagram)



The instrument contains electrostatically sensitive components. Original packing protects against electrostatic discharge (ESD), transport only in original packing.

#### **Connection:**

KS 98-1 engineering must be taken into account, because it determines pin allocation and signification of connections! Moreover, the rules for the performance limits must be followed.

#### **Installing Option Modules**

The design of the product enables option modules to be installed to enhance functionality. The instrument is removed from the rear case by pressing the lugs at the top & bottom of the case whilst withdrawing the front bezel forward from the housing

1. Stecken Sie das Modul in die vorgesehene Position

2. Prüfen Sie den korrekten Sitz der Steckverbindung:

3. Drücken Sie den Abstandshalter in die entsprechende Bohrung der Trägerkarte bis er einrastet.








### 1.5.3. I/O extension with CANopen

The unit offers a CANopen-compliant interface port for connection of the RM 200 system, KS 800 or additional KS 98 units with max. five CAN nodes. See installation notes in the CANopen system manual (9499-040-62411).

## 1.6. Electrical connections

### 1.6.1. Safety hints

-  Die dem Gerät beiliegenden Sicherheitshinweise und die Hinweise ab Seite Fehler! Textmarke nicht definiert. sind unbedingt zu beachten! Die Isolierung des Gerätes entspricht der Norm EN 61 010-1 (VDE 0411-1) mit Verschmutzungsgrad 2, Überspannungskategorie II, Arbeitsspannung  $\leq 300$  V effektiv und Schutzklasse II.
-  Die elektrischen Leitungen sind nach den jeweiligen Landesvorschriften zu verlegen (in Deutschland VDE 100).
-  In der Installation ist für das Gerät ein Schalter oder Leistungsschalter vorzusehen und als solcher zu kennzeichnen. Der Schalter muss in der Nähe des Gerätes angeordnet und dem Benutzer leicht zugänglich sein.
-  Bei gezogenem Geräteeinschub muss ein Schutz gegen das Hereinfallen leitender Teile in das offene Gehäuse angebracht werden.
-  Wird das Gerät in den Offline-Zustand geschaltet, so behalten die Ausgänge die Zustände bei, die sie zum Zeitpunkt der Umschaltung hatten!

### 1.6.2. Electromagnetic compatibility

European guideline 89/336/EEC. The following European standards are met: EN 61326-1

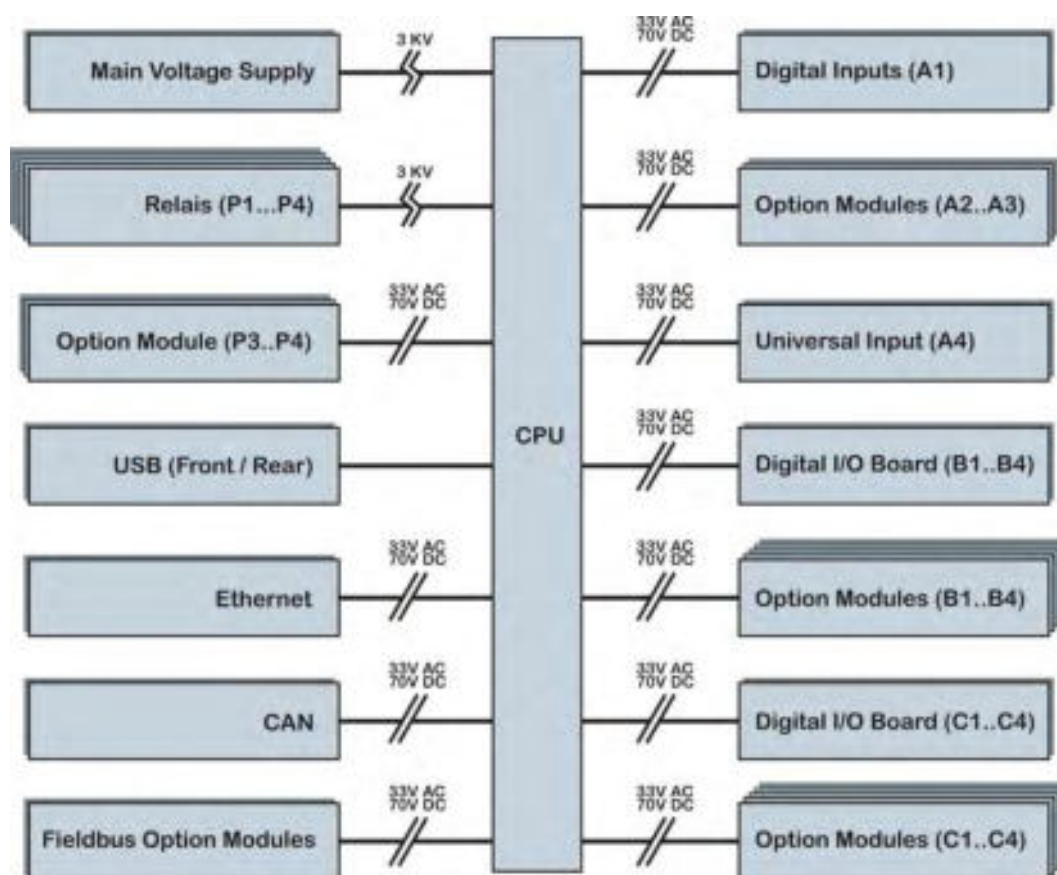
The unit can be installed in industrial areas (risk of radio interference in residential areas).

A considerable increase of the electromagnetic compatibility is possible by the following measures:

- Installation of the unit in an earthed metal control cabinet.
- Keeping power supply cables separate from signal and measurement cables.
- Using twisted and screened measurement and signal cables (connect the screening to measurement earth).
- Providing connected motor actuators with protective circuitry to manufacturer specifications. This measure prevents high voltage peaks which may cause trouble for the instrument.

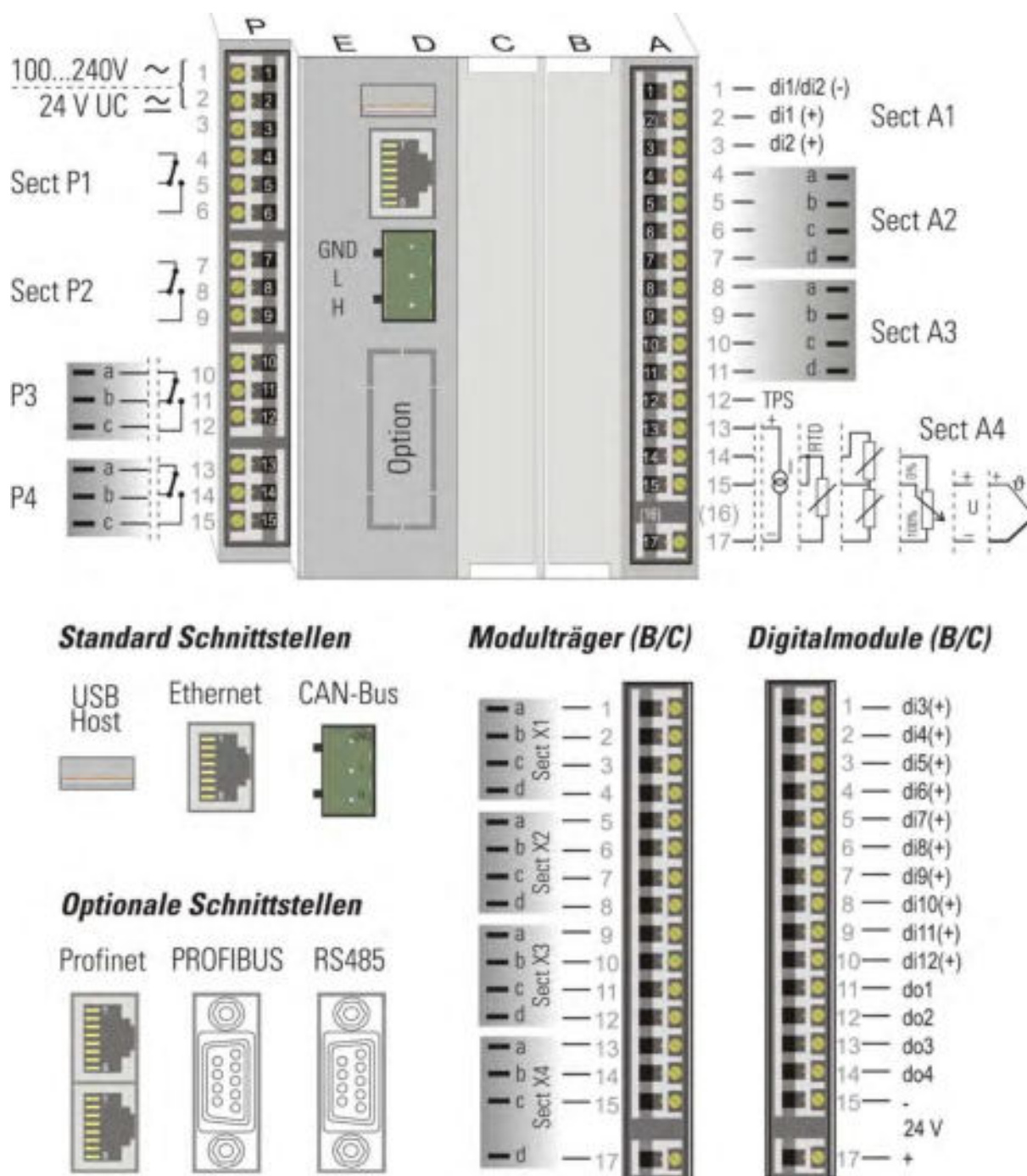


### 1.6.3. Galvanic isolation



- Measuring and signal circuits: functional isolation up to a working voltage  $\leq 33 \text{ VAC} / 70 \text{ VDC}$  against earth (to DIN 61010-1; dashed lines)
- Mains supply circuits 90...250 VAC, 24 VDC: safety isolation between circuits and against earth up to a working voltage  $\leq 300 \text{ V.r.m.s.}$  (to EN 61010-1).
- All I/O extension modules are galvanically isolated from each other and from other signal inputs/outputs (functional isolation). There is no galvanic isolation between the channels of a module.

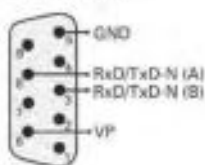
### 1.6.4. General connecting diagram



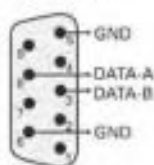
- ⚠ Additionally, the units must be protected by fuses for a max. power consumption of 12,3VA/7,1W per instrument individually or in common (standard fuse ratings, min. 1A)!
- ⚠ The max. permissible working voltage on input and signal circuits is 33 VAC / 70 VDC against earth! Otherwise, the circuits must be isolated and marked with warning label for "contact hazard".
- ⚠ The max. permissible working voltage on mains supply circuits may be 250 VAC against earth and against each other!
- ⚠ On instruments with screw terminals, the insulation must be stripped by min. 12 mm. Choose end crimps accordingly!

### Interface assignment

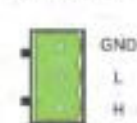
#### PROFIBUS



#### RS485



#### CAN-Bus

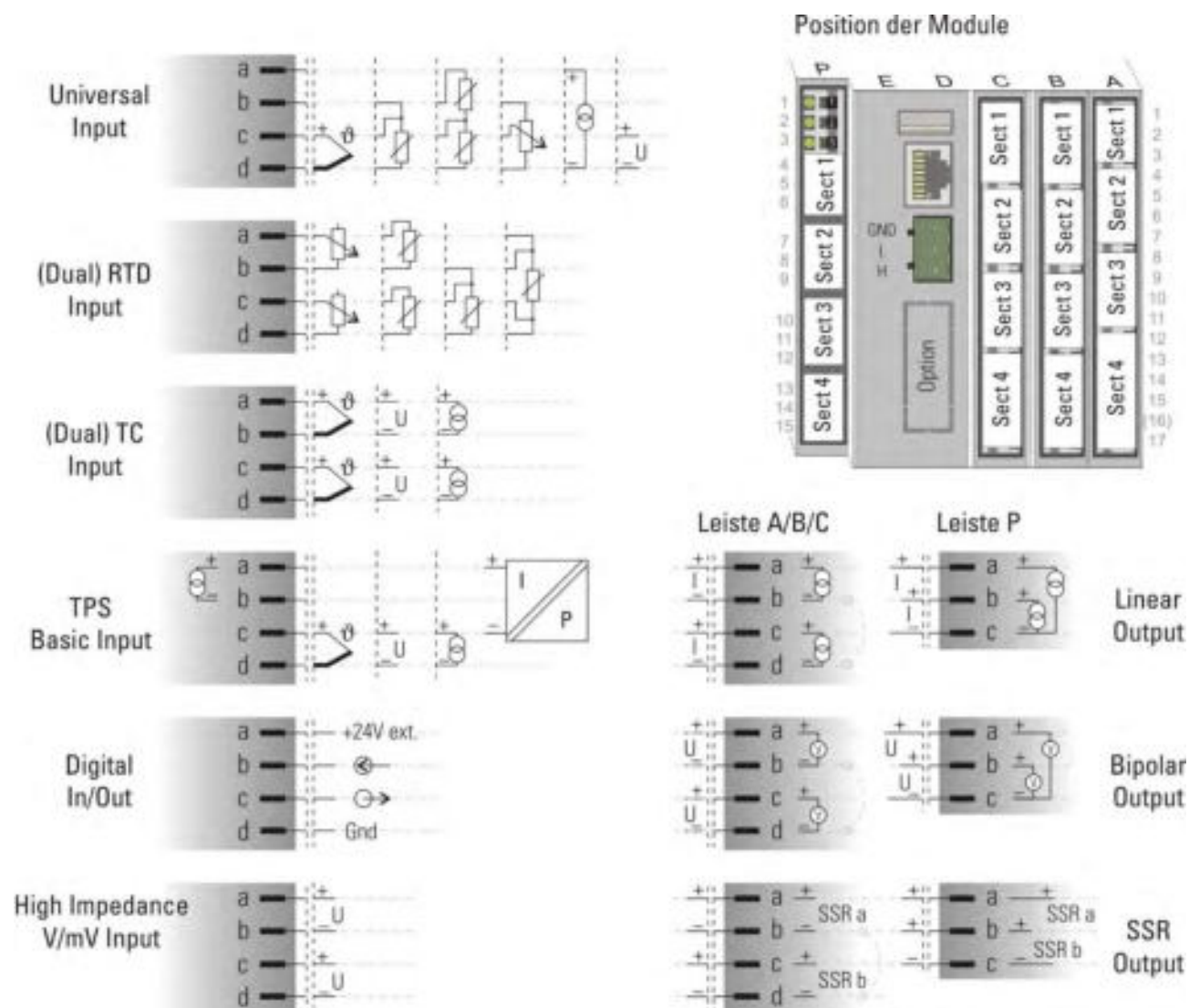


## 1.6.5. Connection Diagram I/O Modules

The inputs and outputs of the multi-functional KS 98-2 instrument can be adapted to the individual needs of the application with plug-in option modules.

The Module Carrier (B/C) cards each offer four positions of various types of I/O modules, which can be combined as needed. There is a limit of max 4 high current outputs (mA out or TPS) per device. The positions of the I/O modules must correspond with the loaded engineering file.

The programmer of the KS98-2 must provide the connection diagram. It can be generated by the engineering tool for the specific device installation.



## 1.6.6. Analog inputs

### Thermocouples

see general connecting diagram on page 26. No lead resistance adjustmentsiehe.

### Internal temperature compensation:

compensating lead up to the instrument terminals.

In the funktionblock of the input  $STK = int.TK$  must be configured.

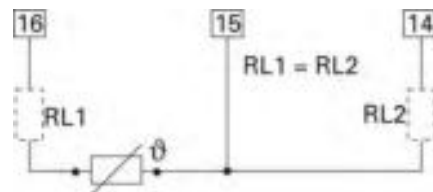
### External temperature compensation:

Use separate cold junction reference with fixed reference temperature.

Compensating lead is used up to the cold junction reference. Copper lead between reference and instrument In the funktionsblock of the input  $STK = ext.TK$  must be configured and at  $TKref =$  reference temperature must be configured.

### Resistance thermometer Pt 100 in 3-wire connection.

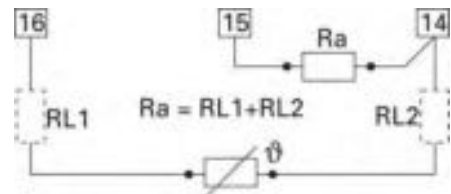
Lead resistance adjustment is not necessary, if  $RL1$  is equal  $RL2$ .



### Resistance thermometer Pt 100 in 2-wire connection.

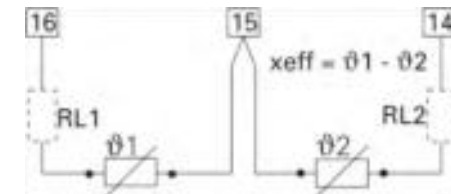
Lead resistance adjustment is necessary:  $R_a$  must be equal to  $RL1 + RL2$

For R\_IN module, lead resistance compensation: → section "Calibrate" page 36



### Two Resistance thermometer Pt100 for difference.

Lead resistance compensation: proceed as described in chapter calibration (see page 36).



### Resistance transducer

Measurement calibration → proceed as described in chapter calibration (see page 36)

### Standard current signals 0/4...20 mA

Input resistance:  $5\Omega$ , configure scaling and digits behind the decimal point.

### Standard voltage signals 0/2...10V

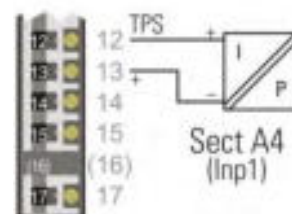
Input resistance:  $\geq 100\text{ k}\Omega$  ( $V\_Modul: \gg 1\text{ G}\Omega$ )

configure scaling and digits behind the decimal point.

### Transmitter supply

All KS98-2 versions contain a potential-free supply voltage for supplying a 2-wire transmitter.

Connection 2-wire measuring transducer (e.g. INP1)



### 1.6.7. Digital inputs and outputs

The digital inputs and outputs must be energized from one or several external 24 V DC sources. Power consumption is 5 mA per input. The max. load is 70 mA per output.

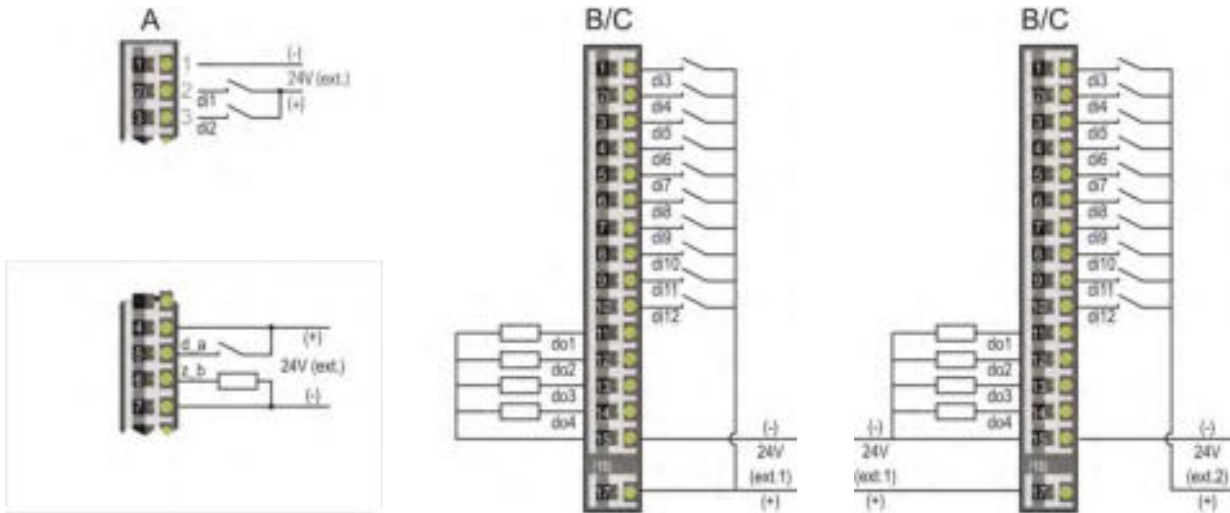
Examples:

Digital inputs (connector A)

Digital I/O module (connector A Sect 2) configured as one di and on do

Digital inputs and outputs at one voltage source (connector B or C)

Digital inputs and outputs at two voltage sources (connector B or C)



## 1.7. Commissioning

Before switching on the instrument, ensure that the following points were considered:

- The supply voltage must correspond to the specification on the type label!
- All covers required for contact protection must be fitted.
- Before operation start, check that other equipment in the same signal loop is not affected. If necessary, appropriate measures must be taken.
- The unit may be operated only in built-in condition.
- The specified temperature limits must be met before and during operation.
- The device is freely programmable. The behavior of the inputs and outputs is therefore determined by the loaded user programming. Before starting, you must ensure that the correct commissioning instructions for the system and the device are available.

**!** The effect of the activation of the individual outputs must be known. Necessary fuses against unintended activation of system components must be carried out in advance. Before switching on, the plant-specific input and output signal types must be set on the device.

This is the only way to avoid damage to the system and to the unit.

If no user programming has yet been loaded in the device, the device is equipped with IO test engineering. With this test program input and output signals can be pre-tested.

**!** The effect on connected equipment must be taken into account.

After supply voltage switch-on, start-up logo and Main menu wait! are displayed, followed by display of the main menu during several seconds.

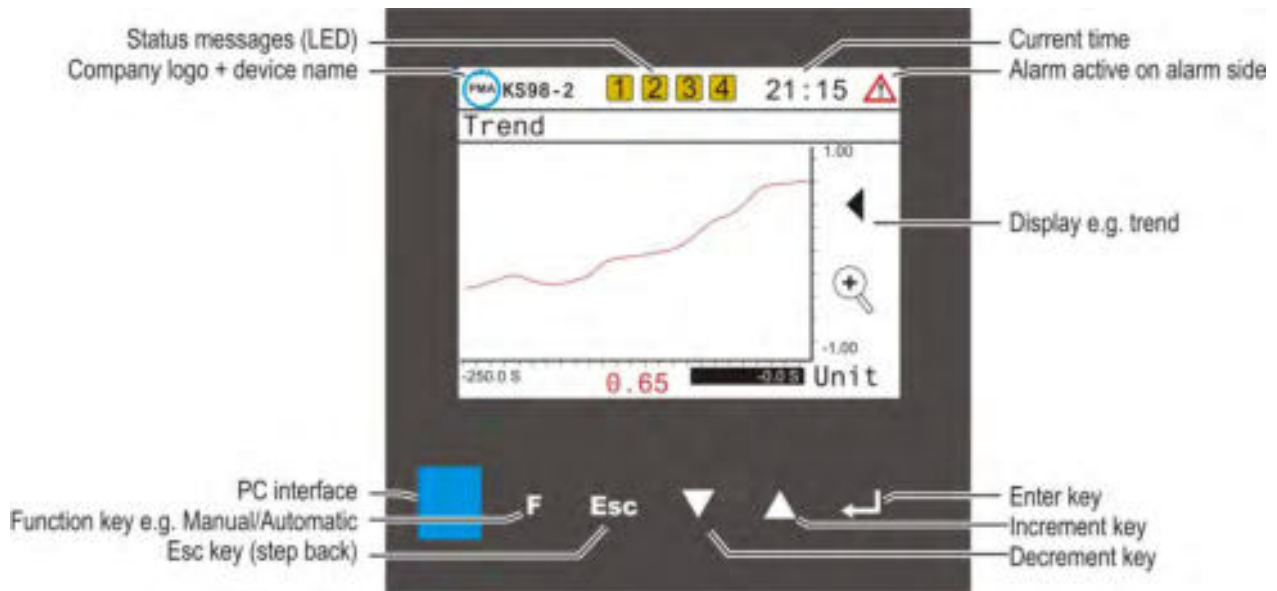
Unless a selection is made during this time, the first operating page (e.g. a controller) is displayed automatically, without marking a line or a field.

## 1.8. Operation

The operation of the device is menu-guided. The menu is divided into several levels which can all be influenced via the engineering, i.e. the final scope of the menu is dependent on the engineering.

This manual describes the operating functions which are independent of the engineering.

### 1.8.1. Front view



Display:

The respective display depends on the functions configured (→ Engineering). TFT touch display (320 x 240 dots).

Status line:

In the status line is shown, the company logo + device name, states provided by the engineering, e.g. Alarms or switching states, the current time and possibly the reference to entries in the alarm page

Keys:

The device is operated by five buttons. With them, the selection of pages, as well as the inputs in the page content will work..

The up / down keys have two functions:

- navigation through menus and on pages
- Changing input values (e.g. )

The two significations of the selector key are dependent on the selected field:

- Pressing the selector key (confirmation / Enter): starts page changing,
- starts value alteration via the up/down keys and confirms the adjustment subsequently (→ page xx).

The escape key is generally used to return to previous pages and to cancel an input action.

The functions of the F-key are dependent on operating page, i.e. this key is sometimes called function key

Function at:	Controller:	auto/manual switchover
	Programmer:	programmer control
	VWERT page	adjustment of digital values (Radio).

PC interface:

PC connection for structuring/wiring/configuring/parameter setting/operating with the engineering tool for KS 98-2.



## 1.8.2. Touch features

The current version includes the following touch functions:

Date / time

Touching the time will jump to the date and time setting page.



Alarm

Touching the alarm message jumps to the display page of the alarms.



## 1.9. Menues

The instrument operation is menu-guided.

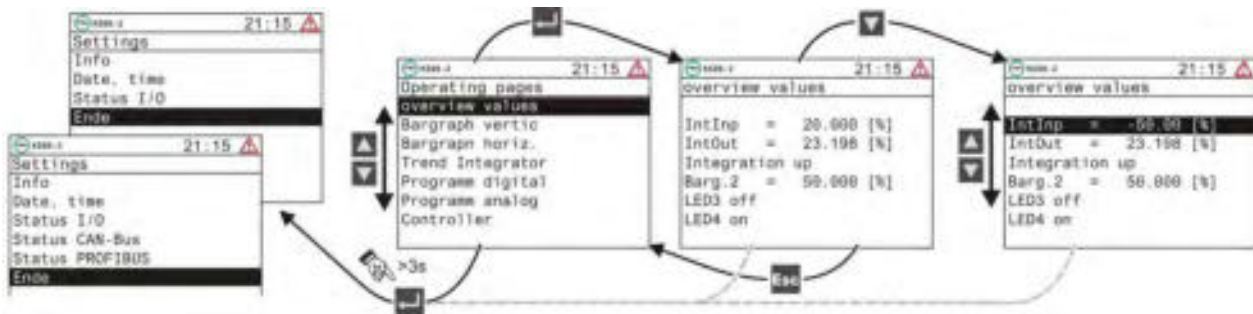
A distinction is made between complete dialogue and short-form dialogue.



In the complete dialog, the main menu with its sub-menus is displayed, i.e. all permitted settings are selectable. During short-form dialog, the main menu is switched off, i.e. unauthorized or accidental access is prevented and only the operating page menu with the permitted operating pages is selectable.

What dialogue is available, is stated in the user programming (function status).

### 1.9.1. Short-form dialog

The operating page menu with the permitted operating pages is selectable. Selecting, marking lines and value adjusting are done as described below.



When pressing key  during > 3 s, a user menu which is different dependent of instrument version (standard / real-time clock and status of I/O CAN-Bus / PROFIBUS) and so on is displayed. Pressing the key  always causes a jump to the previous operating page.

#### Line Info

hardware order no., software order no., software version and operating version of the KS98-2.

#### Line Datum, Uhrzeit

display and adjustment of date and time.

#### Line Status I/O

Placement of slots (terminals A to P), placed module in the slots with appropriate status messages, signals and serial number, hardware and software versions of the modules.

#### Line Status PROFIBUS

status of bus access, parameter setting, configuration and data communication.

#### Line Status CAN-BUS

Address and status of available CAN bus sharing units



## 1.9.2. Complete dialog

A *main menu* for selecting the five *sub-menus*, using which an application-dependent number of *pages* can be selected.


<b>Sub-menu</b>	<i>Contents of pages</i>
<b>Level 1 data</b>	Dependent on engineering, various operating pages are listed and can be selected: Viewing, selecting and adjustment of values, recipes and statuses, etc.
<b>Parameter</b>	A page is provided for each function used with which parameters are adjustable: display and adjust parameters.
<b>I/O-data</b>	A page is provided for each function used: display of input and output data.
<b>Configuration</b>	A page is provided for each function used, which must be configured: Display and adjust configurations. For changing the configuration, the instrument must be set to 'Offline' (r Operating modes).
<b>Miscellaneous</b>	<p>Page <b>Date , Time</b>: display and adjust date and time.</p> <p>Page <b>Device data</b>: display and adjust interface, mains frequency and language.</p> <p>Page <b>Online/Offline</b>: on-line ↔ off-line, cancel configuration.</p> <p>Page <b>Calibration</b>: display and calibrate all signals to be calibrated.</p> <p>Page <b>Info</b>: display hardware / software order no., software version no.</p> <p>Page <b>Status CAN-BUS</b>: status of any connected CAN nodes. ①</p> <p>Page <b>Status PROFIBUS</b>: status of bus access, data communication. ②</p> <p>Page <b>Color setting</b>: Color combination of the display</p>



① Only with option "Standard Interfaces" (KS98-2xx-1xxxx-xxx)

② Only with option "Standard Interfaces" plus PROFIBUS-DP (KS98-2xx-3xxxx-xxx)



## 1.9.3. Selection of operating pages

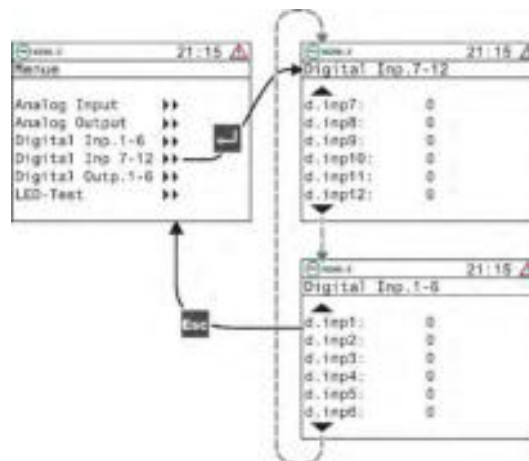
After power switch-on, the instrument starts up with a logo and **Main menu wait!** and then the main menu is displayed during several seconds. Unless a selection is made during this time, the first operating page entered in the sub-menu without marked line is displayed

Pressing the  key always causes a jump to the previous level and abort of value entries.

Use keys  /  - to move the line marking (cursor) upwards up to the start or down to the end of the menu. After pressing the key again, the marking changes from the start to the end and vice versa .

In addition to page selection via the main menu (page list), other pages can be selected from the operating pages, if configured by the engineering:

- Continuation or previous pages are activated via an arrow at the lower (▼) or upper (▲) edge of the page by pressing key  -.
- A subordinated page can be called up using key  in a line marked with ►►.



### 1.9.4. Language selection

German to English: Mark **Allgemeine Daten** → Gerätedaten → Sprach=deutsch. Press **↵**: deutsch blinks. Press **⬆**: english blinks. Press **↵**: Main menu is indicated.

English to German: Mark **Miscellaneous** → Device data → Langu. = english. Press **↵**: english blinks. Press **⬆**: deutsch blinks. Press **↵**: Hauptmenü is indicated.

Fench to English: Mark **Divers** → Données d'appar. → Langu. = francais. Press **↵**: francais blinks. Press **⬆**: english blinks. Press **↵**: Main menu is indicated.

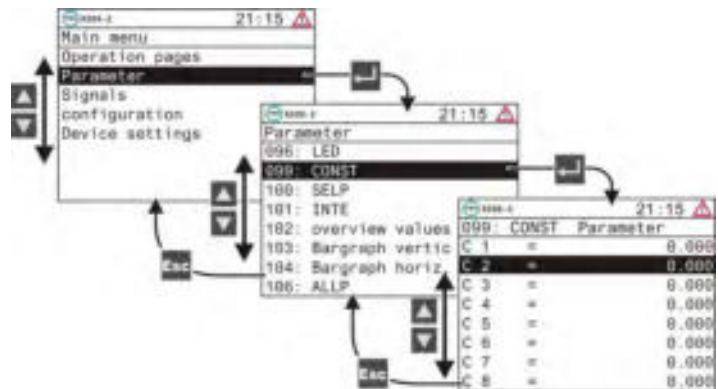
### 1.9.5. Navigation, page selection

The device is operated by **↵**-, **Esc**- and the **⬆** **⬆** buttons. By pressing the **Esc** button for 3 seconds you always go to the main menu.

**👉** When the main menu is locked, the user menu is called up.

Procedure





- ① Press **⬆** **⬆** to select the input field or the line (the selected item is shown inversely),
- ② Confirm the entry with **↵** (for selecting).
- ③ a) If the selected item is a page, the page is opened and navigation can be continued with the **⬆** **⬆** keys.  
b) If an input field was selected, the field starts blinking after pressing key **↵** and the required change can be entered with the **⬆** **⬆** keys. Confirm with key **↵**. The input field stops blinking and the alteration is saved.
- ④ To leave a page and change to the previous bed linen, the **Esc** button is used. Pressing the **Esc** button for 3 seconds repeatedly returns to the main menu.

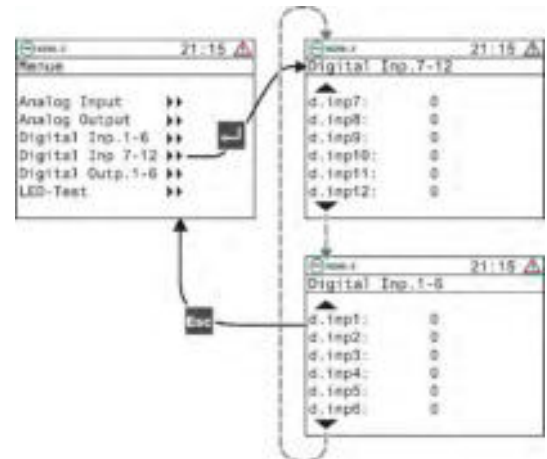


**👉** Unless display of a page is inverse despite actuation of keys **⬆** **⬆**, the items were disabled (e.g. via engineering).

### Operating pages:

These pages offer an additional navigation function:

- Continuation or previous pages (marked by an arrow at the bottom (▼) or top (▲) of the page) can be activated by selecting and pressing key .
- Items marked with ►► open another operating page when selecting () and confirming with key .
- Leaving an operating page is done by using the key .









### 1.9.6. Adjusting values

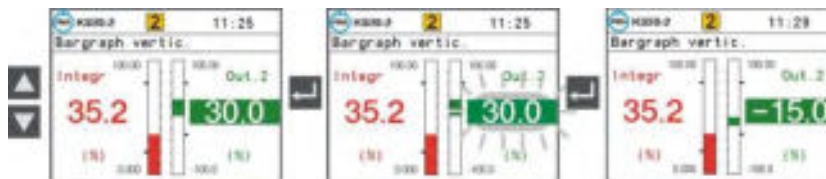
The menu operating pages include various types of fields for adjustment of values:

- analog values, digital values
- selection lists
- times
- on/off switches
- push-buttons
- selector switches (radio button)

#### Adjustment procedure

Select the value to be altered with keys  .

- a) Press key  to start value changing (field blinks). Change the value with keys  .
- Press  to store the change (field stops blinking).
- The longer keys up/down are pressed, the higher is the acceleration. When releasing, the adjustment speed decreases accordingly.



- b) Key . This mode of adjustment is for switches, push-buttons and selector switches.



## 1.10. Device settings in the main menu

### 1.10.1. Date, Time

See and set the date and time.

In the standard interface version, the device contains a backup battery to store this data.

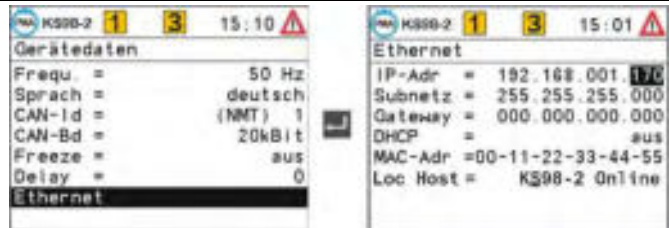
The flashing of the clock in the status line of the KS98-2 indicates that the clock and date must be set.

If the date and time are not updated, the flashing disappears after approx. 3 min., setting of date and time is always possible.

### 1.10.2. Device data

Depending on the hardware version of the device, the settings for the following functions are made on this page:

- Main frequency
- Language
- Interfaces



### 1.10.3. Online/Offline

For configuration changing, switch the unit to 'Offline' and back to 'Online'.

When switching the unit to status off-line, the outputs will remain in the status at switch-over time!

By switching over to on-line, all data is saved.

When terminating the off-line mode by cancellation (Escape config.) the data saved last is loaded back into the working memory.

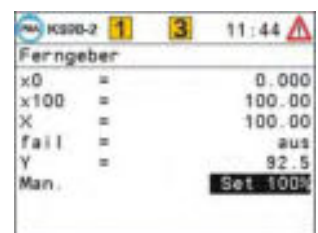
### 1.10.4. Calibration

Press to select the input and to open the calibration page.

*Transducer input:*

Adjusting the transducer start and end:

- ① Select **Quit** and set transducer to start
- ② Press → **Quit** blinks
- ③ Press → **Set 0%** blinks
- ④ Wait until the input has settled (min. 6 s)
- ⑤ Press → **0% done** is displayed
- ⑥ Set transducer to **end**
- ⑦ Press → **0% done** blinks
- ⑧ Press 3x → **Set 100%** blinks
- ⑨ Wait until the input has settled (min. 6 s)
- ⑩ Press → **100% done** is displayed.



Calibration is finished. For exit from calibration press .

Two resistance thermometers are resistance thermometers in 2-wire connection:  
Calibration of lead resistance effect:

- ① select **Quit**. Short-circuit both thermometers or thermometer in the connecting head
- ② Press → **Quit** blinks
- ③ Press → **Set Dif** blinks
- ④ Wait until the input has settled (min. 6 s)
- ⑤ Press → **Cal done** is displayed.



Lead resistance adjustment is finished. Remove both short circuits. For exit from calibration press .

### 1.10.5. Info

This page is used to display all device information

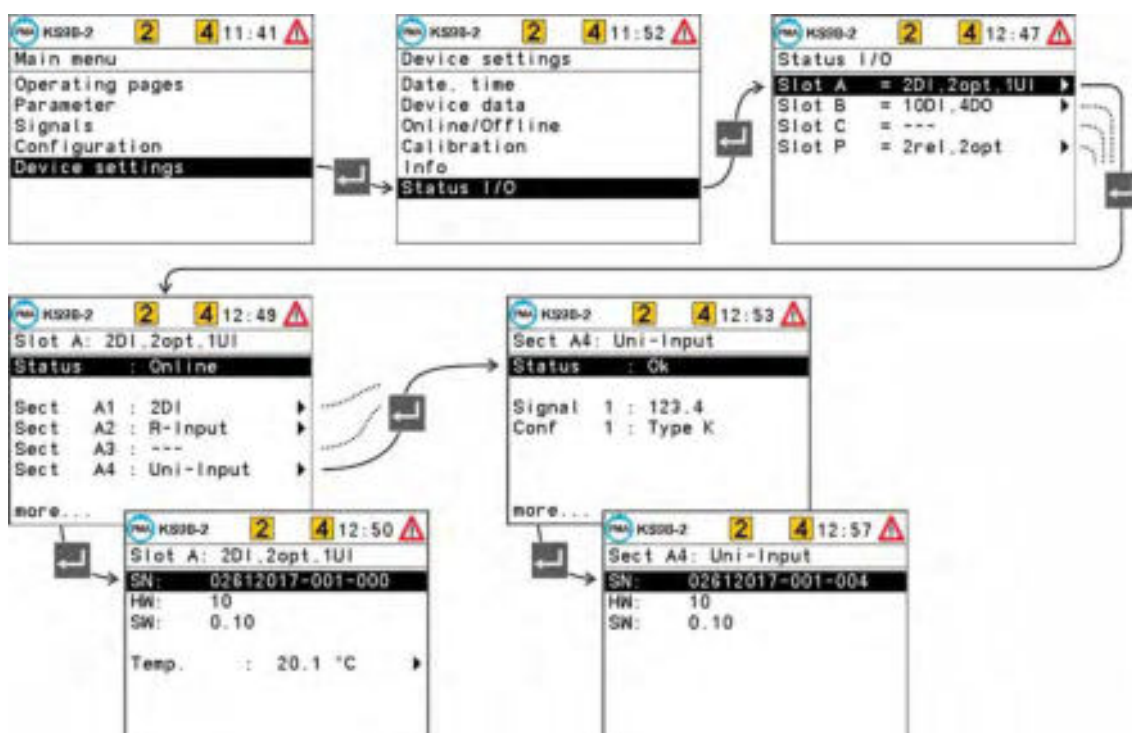
- Hardware
- Software
- Operating version
- Serial no.



### 1.10.6. Status I / O

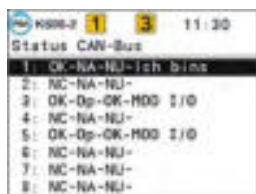
The status page of the I / O modules provides information about the proper installation, possible faulty installations are displayed:

- Difference between configured and plugged module type, unequaled slots or modules are shown with ---.
- **Exceeding the performance limits**
- Under **"more . . ."** you will find the hardware / software version of the slot / module.



### 1.10.7. CAN-Status

The CAN bus status with the connected units is displayed.

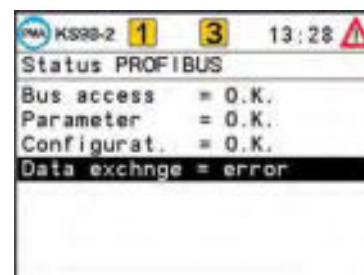


Value	Signification
<b>1...42</b>	Node number
<b>NC</b>	NoCheck: Node existence not checked so far / node not provided
<b>Ck</b>	Check: Node existence just being checked
<b>NR</b>	NoResponse: No reply from this node, but node is required.
<b>OK</b>	Ready: Node has replied and is identified.
<b>ES</b>	EMStart: Node has provided an emergency message.
<b>NA</b>	NotAvailable: Node status is unknown.
<b>PO</b>	PreOperation: Node is in the PreOperational status.
<b>Er</b>	Error: Node is in error status.
<b>Op</b>	Operational: Node is in Operational status.
<b>NU</b>	NotUsed: Node is not required by an own lib function.
<b>Wa</b>	Waiting: Lib function waits for identification of this node.
<b>Pa</b>	Parametrierung: Lib function just setting the nodeparameters.
<b>OK</b>	Ready: Lib function has finished the parameter setting.
<b>String</b>	determined node name

### 1.10.8. Profibus-Status

The Profibus status page provides information on the Profibus connection status. The following error statuses are displayed:

- Bus access not successful
- Faulty parameter setting
- Faulty configuration
- No data communication



### 1.10.9. USB Menü

If a USB stick is plugged into the USB host port (rear USB port), the USB menu can be selected. Here the complete KS98-2 setting can be stored on a USB stick. To transfer an engineering from the USB stick to the KS98-2, the device must be **offline**.

## 1.11. Operating pages

The engineering determines the scope of available operating pages. All available pages are listed in the operating page menu. The various types of pages are explained below.

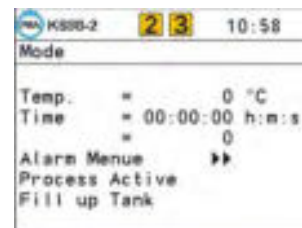
### 1.11.1. List display

The operating page list is intended for display/input of process values and parameters.

Apart from digital, analog and time values, values of type radio button, switch and push-button can be defined in the value listing (→ page 35).

The value signification is determined by the engineering. The displayed values can be input fields.

A complete description of the functionality is available in the function block description (→ page 141)



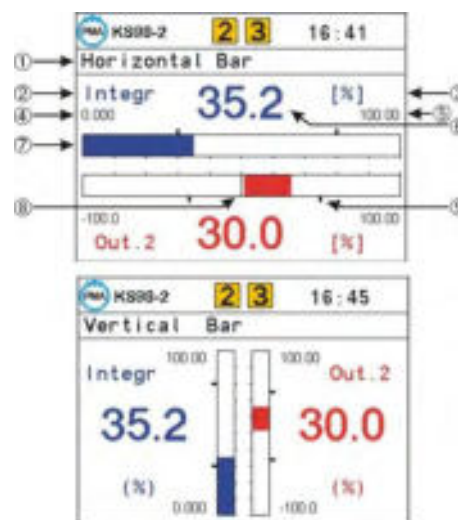
### 1.11.2. Bargraph display

The bargraph page is used for display of two analog variables as a bargraph.

Two further variables can be displayed and changed numerically and need not correspond with the bargraph values.

Four further analog inputs can be used to position two markers at the bargraph side, e.g. for indication of the alarm limit or reference values. When exceeding the limits, an arrow is displayed at the top and bottom end of the bargraph ► (see page 49)

- ① Title
- ② Name for value
- ③ Unit for value
- ④ + ⑤ Scale end values
- ⑥ Display/input field for value
- ⑦ Bargraph
- ⑧ Bargraph origin
- ⑨ Limit value markers for bargraph



### 1.11.3. Alarm display

Alarm display is in the order of occurrence on a list.

One alarm per line is displayed.

- |  |  |
|--|--|
| ① Alarm active                                   | alarm text blinks                                |
| ② Alarm active and ackn.                         | Alarm text not blinking and highlighted in blue  |
| ③ Alarm not active any more and not acknowledged | Alarm text not blinking and highlighted in green |
| ④ Alarm not active any more                      | -----  |

#### Acknowledging an alarm

Select an active alarm for acknowledging with and acknowledge it with .

New alarms are displayed only when rebuilding the page, which is done by pressing key .

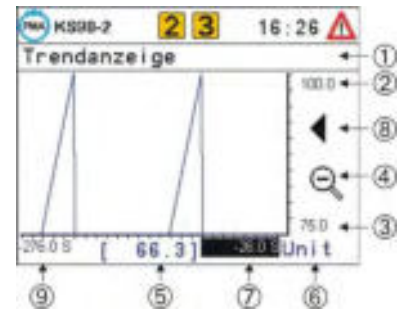




### 1.11.4. Graphic trend curve




The time curve of a process is displayed graphically on the trend page.

- ① Title
- ② + ③ Scale end values
- ④ Zoom switchover
- ⑤ Value at time ⑦/ actual input value
- ⑥ Unit of value
- ⑦ Origin of time axis related to the actual value  
(=0) shift of time axis (scrolling into the past)
- ⑧ Axis shift signalling
- ⑨ End of time axis / earliest value in the displayed trend





#### Zoom value scale

 The value axis can be zoomed by factor 1:4 (cut-out magnification)

 "Select the "zoom" field ④, press , the zoom symbol changes. Now, scaling can be changed by means of keys . The scaling is shifted in steps of 12,5% via field ③

#### Shift of time axis:

Earlier values than those visible in the actual window are also displayed by the trend function (Shift). Values left of the time axis are earlier values. These values can be displayed by changing the origin of the time axis Select field ⑦ with  and shift the scale origin by changing the value

 Symbol  (⑧) indicates the shift. Wird die Zeitskala wieder auf 0 gesetzt, ist die Verschiebung ausgeschaltet.




### 1.11.5. Programmer

- A programmer controls the process sequence of a plant.
- Programmers are configurable freely in structure and scope by means of the engineering.
- A programmer is composed of any number of s (analog values) and control values (digital control bits).
- Any number of programs (recipes) can be stored for a programmer.
- The program is divided into a defined number of segments (program segments).
- The maximum number of segments is determined in the engineering.
- The maximum scope is defined in the engineering.

The actual status of a running program is displayed on the programmer operating page. Dependent on programming, status (run/stop, auto/manual), segment number, net time and the actual (during manual operation) can be changed.

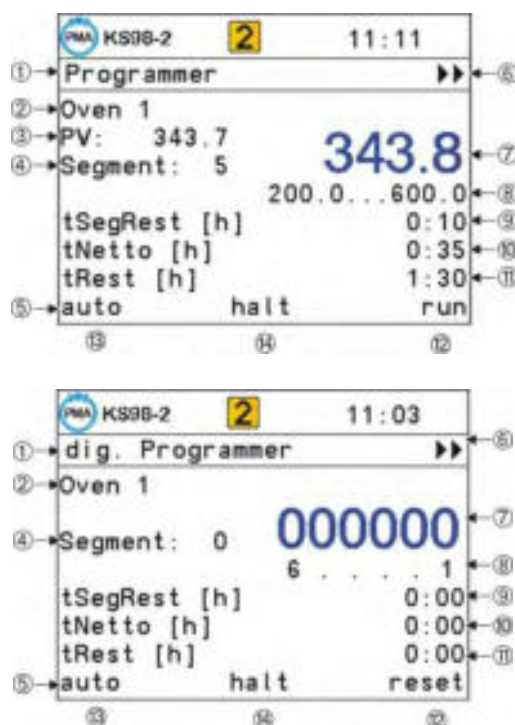
Programmer operation is divided into:

- Program control and monitoring
- Program (recipe) selection
- Adjustment of setpoints/control bits during manual operation
- Parameter setting for program

 Dependent on engineering, parts of this operation can be changed or disabled.


Display of the operating page is always related to one programmer output, whereby analog s and digital control bits are distinguished. Change to the next programmer output is via field ⑥ ►► in the title.

- ① Name of operating page
- ② Program name/no. (recipe)
- ③ [Process value]
- ④ Actual segment no.
- ⑤ Status line
- ⑥ Programmer output switchover
- ⑦ /control value
- ⑧ Setpoint from...to in the current segment
- ⑨ Remaining segment time
- ⑩ Elapsed program time
- ⑪ Remaining program time
- ⑫ Program status  
(stop,run,reset,search,program,quit,error)
- ⑬ auto/manual
- ⑭ halt, end




### Selecting a program


Selection of a program is by alteration of recipe field ②. Dependent on engineering, selection is from a text list or by entry of a number.

 Program selection is possible only in status "reset".

### Controlling a program

Press key  to control the program sequence:

The time curve can be controlled also by changing the elapsed time ⑩ or segment number ④ (preset)

 Dependent on engineering, parts of this operation may be changed or disabled.



### Program parameter setting

Select the program for editing via field "Rec" ②. Call up the relevant s/control values, segment times and types with menu item "program" in the status line (field ⑫)

A page on which the selected program is displayed as "RecEdt" is opened. The parameters are listed in the order of segments

The data blocks are displayed dependent on engineering. The type of individual segments can be changed dependent on data block type.

Selection of all programs including the inactive ones is possible on line **RecEdt** in any programmer status.

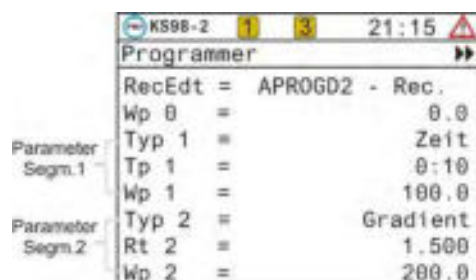
When using recipe names, these names are displayed on the editing page. Switching over to the parameters of a different recipe can be done by altering the recipe name. This is possible at any time and does not cause switchover of the active recipe.

A segment list is completed with end identification -- : -- in parameter  $T_{pn}$  of the last segment. When setting the last segment time  $T_n$  to a valid value (higher or equal to 0), the next parameter is displayed automatically  $T_{n+1}$  = -- : -- etc

This procedure permits shortening of a current program by adjusting a value  $< 0$  in the required position for  $T_n$  = -- : -- with key 

The following segments will be suppressed in the program. The relevant segment parameters remain unchanged and can be re-activated by input of a valid value for  $T_n$ .





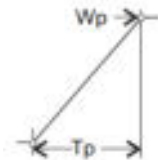
## Segment types

Dependent on segment type, the following parameters can be altered:

Wp i	Target setpoint
D i	Control value in segment i
Tp i	Segment duration
Rt i	Segment gradient
Typ i	Segment type

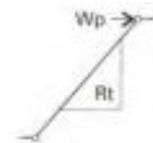
### Ramp segment (time)

With a ramp segment (time), the runs linearly from the start value (end of previous segment) towards the target (Wp) of the relevant segment during time Tp (segment duration)



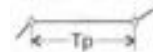
### Ramp segment (gradient)

With a ramp segment (gradient), the runs linearly from the start value (end value of previous segment) towards the target value (Wp) of the relevant segment. The gradient is determined by parameter Rt.



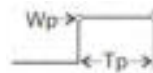
### Hold segment

With a hold segment, the end of the previous segment is output constantly during a defined time which is determined by parameter Tp.



### Step segment

With a step segment, the program goes directly to the value specified in parameter Wp. The reached due to the step change is kept constant during the time determined in parameter Tp.



### Waiting and operator call

All segment types can be combined with "Wait at the end and operator call".

If a segment with combination "wait" was configured, the programmer goes to stop mode at the segment end. Now, the programmer can be restarted by pressing the **PRESET**-key

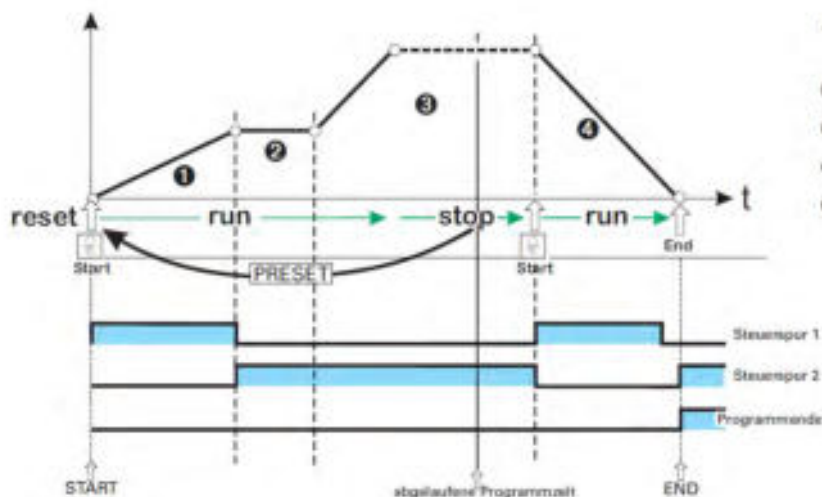




Fig.25


- ① Segmenttyp = Zeit
- ② Segmenttyp = Halten
- ③ Segmenttyp = Zeit und warten
- ④ Segmenttyp = Gradient

### Manual mode

The programmer output can be overwritten for each page. For this, the relevant page must be switched off to "manual"⑬. In this mode, the or control value can be overwritten ⑦. The control value is changed separately for each control bit. Press  to continue. Field ⑬ permits returning to the automatic mode (→ page 41)

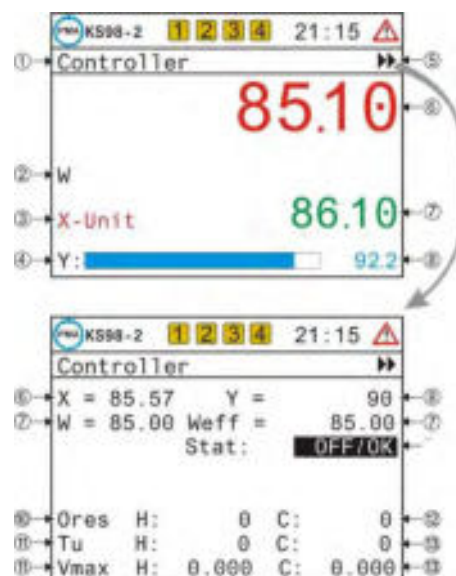
 The program run is not interrupted by the manual mode.


### 1.11.6. Controller

The controller page permits intervention into process control loops. Input fields (, source, correcting variable during manual mode, parameter set switchover) are selected via the  key, pure display fields are skipped.

 Dependent on engineering, the input fields can be disabled.

- ① Page title
- ② source (Wint, Wext, W2)
- ③ Physical unit
- ④ Bargraph of correcting variable Y or XW or Xeff
- ⑤ Entrance into the self-tuning page
- ⑥ Effective process value
- ⑦ Controller setpoint
- ⑧ Value of correcting variable Y or XW or Xeff
- ⑨ Self-tuning/command input status
- ⑩ Self-tuning result heating
- ⑪ Process characteristics heating
- ⑫ Self-tuning result cooling
- ⑬ Process characteristics cooling



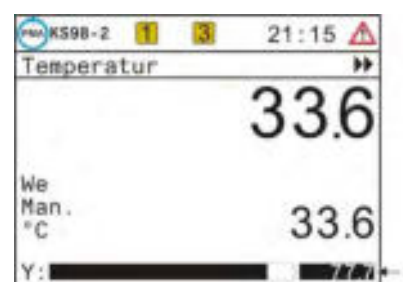
Apart from entries and switchover operations, further actions can be started: Switching over to manual operation is done via key  and field ⑤ provides access to the controller self-tuning page.

### Input fields on the operating page

#### Handverstellung

This field can be used to adjust the correcting variable during manual operation. Adjustment is enabled only during manual operation. Unless manual operation is active, the field cannot be selected.

When changing over to manual operation, the bargraph display is always switched over to Y display (correcting variable), also with X1 or XW defined in the configuration. The actual correcting variable is displayed right beside the bargraph.



#### Manual correcting variable

Alteration of manual correcting variable ⑧ by means of keys  is at three speeds. Press the key to start the adjustment at a speed of 1% / sec. After 3 sec., switchover to 2.5% / sec occurs and switchover to 10%/sec. is after another 3 sec

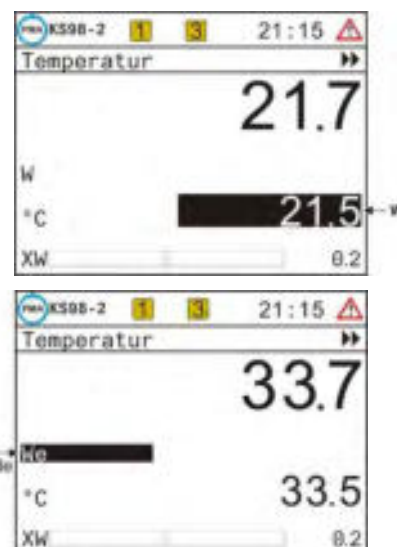
### Setpoint

The internal setpoint can be altered at any time, also when another is active.

### Setpoint source

source switchover is possible via a selection field on the controller page.

Dependent on controller configuration, selection of Wint, Wext and W2 is possible. The field can be left with Quit, if no switchover is needed.




### Self tuning

Determination of the optimum process parameters is possible by self-tuning. Self-tuning is available for processes with compensation and no delay time.

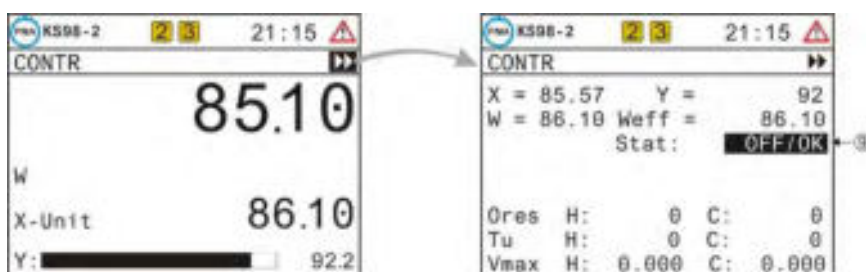
Dependent on controller type, parameters  $Xp1$ ,  $Xp2$ ,  $Tn$ ,  $Tv$ ,  $Tp1$ ,  $Tp2$  are determined.

### Preparation

- Adjust the desired controller behaviour.
- The parameters  $Tn$  or  $Tv$  can be switched off by the value = 0.0.
- P- controller:  $Tn = 0.0$   $Tv = 0.0$
- PD- controller:  $Tn = 0.0$   $Tv > 0.0$
- PI- controller:  $Tn > 0.0$   $Tv = 0.0$
- PID- controller:  $Tn > 0.0$   $Tv > 0.0$
- If a controller has several parameter sets, selection of the parameter sets to be optimized is required (( POpt=1...6). If necessary, these settings must be made available when creating the engineering ).
- Switch the controller to manual mode (key ). Alter the correcting variable to reach the working point.

The process must be in a stable condition. Self-tuning starts only, when process value oscillations are smaller than 0.5% of the control range during one minute (controller display: process at rest' (PiR))

 If necessary, other control loops in the plant must be set to manual mode as well.




### Setpoint reserve:


To permit self-tuning, the distance between setpoint and process value must be higher than 10% of the setpoint range before self-tuning start.

With inverse controllers, the must be higher than the process value. With direct controllers, it must be smaller. The determines a limit which is not exceeded during self-tuning

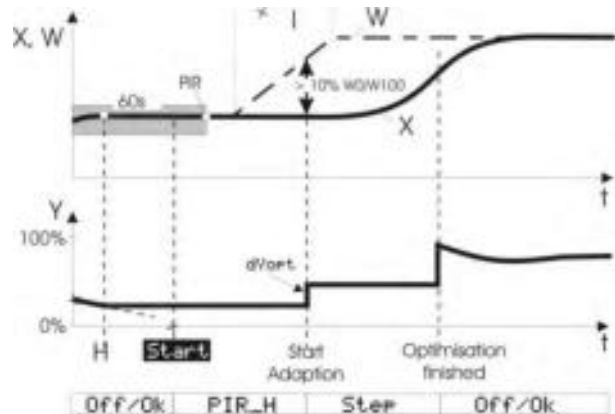
### Self-tuning start

Select function **Stat**: OFF/OK and confirm it with .


**Stat**: OFF/OK blinks and is switched over to **Stat**: Start by pressing .

Pressing key  starts the self-tuning attempt. The can be changed also subsequently. After successful self-tuning, the controller changes to automatic operation and controls the with the new parameters.

When 'Process at rest' (PiR) is detected and a sufficient reserve is provided, the correcting variable is changed by output step (increased with indirect controller, decreased with direct controller).



**!** The size of the output step change is set to 100% as standard. In critical processes, this value (parameter dYopt) may have to be reduced to prevent damage to the process. The parameter is adjustable in the engineering, or via the parameter dialogue of the main menu, if the engineering is known. In case of doubt, contact the programming engineer.

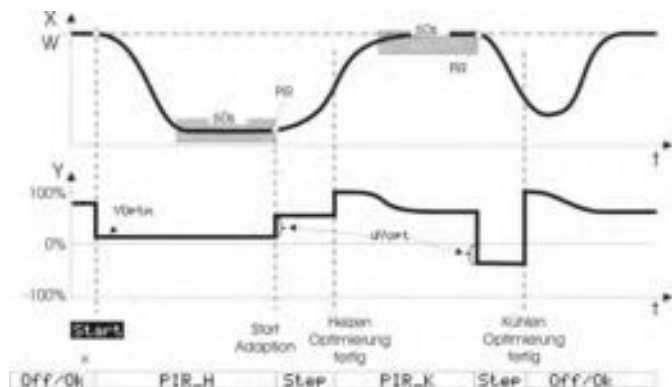
When self-tuning is finished with an error (Ada\_Err or 0err on the controller page), the initial correcting variable is output until self-tuning is finished by pressing key .

### Self-tuning procedure with heating and cooling processes:

(3-point / split-range controller)

Self-tuning starts as with a "heating" process. After self-tuning end, the controller settings based on the calculated parameters are made. This is followed by line-out at the pre-defined, until PiR is reached again.

Subsequently, a step to cooling is made to determine the "cooling" parameters. When cancelling the cooling attempt, the parameters for "heating" are also taken over for cooling. No error message (Ada\_Err) is output.



**!** Whilst self-tuning is active, the control function is switched off!

The self-tuning statuses are indicated with priority in the display field for manual operation.

- Self-tuning running, display: **ORun**
- Self-tuning faulty, display: **OErr**

Self-tuning completed with an error is finished by pressing key **F** twice.



### Self-tuning cancelation

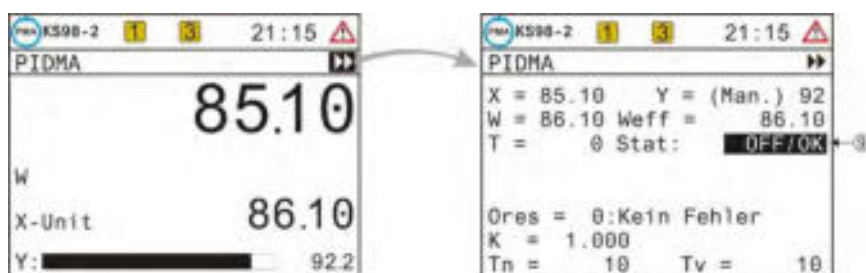
Self-tuning can be stopped at any time by pressing key **F**, or by selecting Stop in the Stat field (status).

### Signification of self-tuning messages ORes1/ORes2 for controller type CONTR and CONTR+

ORes1/2	Signification or trouble cause	Possible solution
0	No attempt was made or attempt cancelled by Stat: Stop or switchover to manual mode ( <b>F</b> key).	
1	Cancellation: Faulty correcting variable output action, X does not change in the direction of.	Change controller output action.
2	Finished: self-tuning was completed successfully (reversal point found, safe estimation)	
3	Cancellation: The process does not respond or responds too slowly (change of $\Delta X$ smaller than 1% in 1 hour)	Close control loop.
4	Completed, without <b>AdaErr</b> : Successful attempt, process has a low reversal point Cancellation, with <b>AdaErr</b> : Attempt failed, process stimulation low (Reversal point found, but estimation is unsafe)	Optimum result with low reversal  Increase output step change <b>dYoptm</b> .
5	Cancellation: Self-tuning cancelled because of exceeded hazard.	Increase separation of process value (X) and (W) when starting, or decrease <b>YOptm</b>
6	Completed: attempt successful, but self-tuning cancelled due to exceeded hazard. (Reversal point not reached so far; safe estimation).	
7	Cancellation: Output step change too small, $\Delta Y < 5\%$	Increase <b>Ymax</b> or set <b>Yoptm</b> to a smaller value..
8	Cancellation: reserve too small, or exceeded whilst PiR monitoring is busy.	Vary stable correcting variable <b>YOptm</b>



The controller type PIDMA offers the following self-tuning page.



For self-tuning preparation, parameters must be adjusted dependent on process and engineering. For this purpose, special knowledge of the applicable function block is required, i.e. it should be done by the programming engineer. Self-tuning start is as described above.

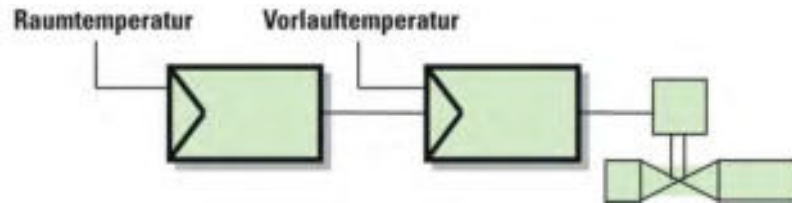
### Signification of self-tuning messages ORes for PIDMA

ORes	Signification or error cause	Possible solution
0	No attempt was made	
1	Xlimit too small	Step change threshold too small: compared to the process noise, the step change threshold is too small. Start a new attempt with a higher positioning pulse.
2	dYopt large	Positioning pulse too high: the correcting variable would exceed the positioning limits when the selected pulse height is output. Start a new attempt with smaller positioning pulse or reduce the correcting variable in manual mode previously.
3	Start again	No rest. The autotuner has detected that the process is probably not at rest. Please wait, until reaching the rest condition. Another possibility is to activate the drift compensation or to increase the positioning pulse. Note: With pulse width modulated (PWM) control outputs (2 and 3-point controller), oscillations of process value PV are susceptible to occur even during manual mode, if the corresponding cycle time t1 (t2) is too long. In this case, the controller cycle times should be as low as possible.
4	dYopt small	Positioning pulse too small: the step response is hidden by process noise. Start a new attempt with a higher positioning pulse, or take measures to reduce the noise (e.g. filter).
5	No peak	Max. detection failed: after output of the positioning pulse, no maximum / minimum in the process value curve was detected. The settings for the process type (with / without compensation) should be checked.
6	Output sat	Positioning limits during self-tuning were exceeded. During the attempt, correcting variable MV has exceeded the positioning limits. Repeat the attempt using a smaller positioning pulse or a reduced correcting variable during manual mode.
7	Controller type	No self-tuning result for the specified combination P/I/D can be found
8	Monotony	Process not monotonous: the process has a strong all-pass behaviour (temporarily, the process value runs in opposite direction) or serious trouble during the attempt.
9	Extrapolation	Extrapolation failed: after the positioning pulse end, no process value decrease was detected because of excessive noise. Increase the positioning pulse or attenuate the noise.
10	Bad result	Result useless: excessive noise, or the determined process parameters do not correspond to the description of a process with dead band. Start a new attempt with a higher positioning pulse or attenuate the noise.
11	Man. break	The self-tuning attempt was canceled manually by the operator with „STOP“.
12	Direction	Faulty output action: the expected output action of the step response is opposed to the correcting variable. Cause can be faulty setting of the output action, or e.g. inverting actuators. Change the controller output action.



### 1.11.7. Cascade controller

With cascade control, two coupled controllers act on a common actuator. A process value for the master and a process value for the slave controller are required.




The slave is determined via the external by the master. Cascade operation is possible in the following statuses.

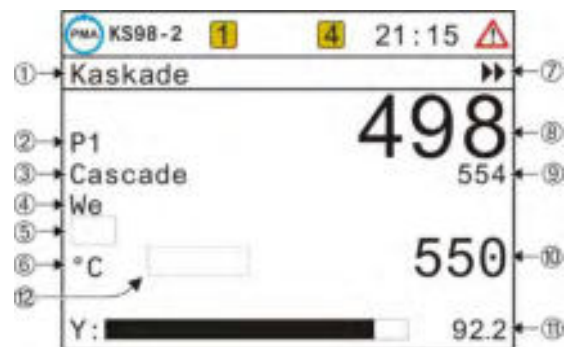
Die Kaskade kann in folgenden Betriebszuständen bedient werden:

#### Automatic mode

In a cascade, master and slave operate automatically during automatic mode. The master and process value are the relevant variables for process control. The master is adjustable. The process value of the slave ⑨ is displayed additionally.

 "Cascade" is displayed.

- ① Operating page title
- ② Parameter set selection, if available
- ③ Switchover field cascade mode (open/closed)
- ④ Setpoint-source of master ( $W_{int}$ ,  $W_{ext}$ ,  $W_2$ )
- ⑤ Display field for manual mode (otherwise empty)
- ⑥ Physical unit (master block parameter)
- ⑦ Entry into self-tuning
- ⑧ Master process value
- ⑨ Slave process value
- ⑩ Setpoint (from master in automatic mode, from slave with open cascade)
- ⑪ Bargraph and display (Y from slave or X/XW from master)
- ⑫ Display of slave selection with open cascade (otherwise empty)



### Cascade opened

For opening the cascade and control by means of the slave controller (see note text "Slave" on the operating page), switchover field ③ is switched to "Casc- Open".

 "Casc-open" is displayed

The slave setpoint is displayed now.


Now, the slave controller setpoint becomes the variable used for process control and can be adjusted.


The process value of the master control loop is set by the cascade loop rather than being controlled. Setpoint-switchover between setpoint operation by master or slave is always possible

In cascade mode, the master information is displayed in the fields for setpoint, setpoint-source, physical unit and X/XW bargraph. With open cascade (display "Slave"), the slave information is displayed



### Manual mode


Switchover to manual is via key  (display in field ⑤). The cascade status (open/closed) is not affected. In manual mode the process is controlled directly with the slave correcting variable. The slave correcting variable can be adjusted during manual operation

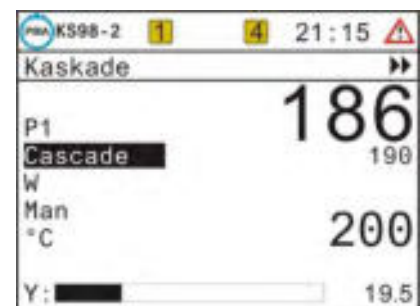
 "Man" is displayed.

#### *Cascade optimization*

In a cascade, the slave controller and then the master must be optimized.

The self-tuning entry of the cascade operating page  
►► relates always to the slave!

 For optimizing the master controller, the master must be selected purposefully via the operating menu. For this, the project description must be used.



## 1.12. Maintenance, test, trouble shooting

### 1.12.1. Cleaning

Housing and front panel can be cleaned using a dry, lint-free cloth. No use of solvents or cleansing agents.

 Avoid using solvents or cleansing agents!


### 1.12.2. Behaviour in case of trouble

The unit needs no maintenance. In case of trouble, check.

- Is the unit in on-line mode?
- Is the supply voltage connected correctly?
- Are voltage and frequency within the tolerances?
- Were all connections made correctly?
- Do the sensors and actuators work properly?
- Is the engineering OK?
- Is the unit configured for the required operating principle?
- Do the adjusted parameters have the required effect?
- Are the I/O extension modules plugged in and clicked in position correctly?
- Is a terminating resistor activated (can be required dependent on the instrument position in the bus topology with CANopen and PROFIBUS DP)?
- Were the required EMC measures carried out (screened cables, earthings, protective circuits, etc.)?
- Does the diagnostic page of the test engineering indicate an error?


If the unit does not function correctly after these checks, it must be shut down and replaced. A defective unit can be returned to the supplier for repair.

### 1.12.3. Shut-down


 Disconnect the supply voltage completely and protect the unit against accidental operation. As the instrument is mostly connected with other facilities in the control loop, consider the effects before switching off and take measures to prevent the occurrence of undesired operating conditions!

### 1.12.4. Default engineering as basic equipment

Without setting (engineering), the KS98-2 contains a default engineering, which ensures the checking of the always existing inputs and outputs of the basic device. The engineering includes a simple application with a controller as well as alarm processing, trend display and color changeover, furthermore the display of a running text is shown.

 If KS98-1 is provided with customer-specific engineering, the engineering description is applicable.

 This engineering is not suitable for controlling a plant. This requires customized engineering.

 Faulty settings can cause damage to instrument and plant!

## 2. Engineering-Tool

### 2.1. Survey

The engineering tool for KS 98-1 enables the user to make an engineering which is specially tailored for his application. The engineering tool mainly comprises a function block editor supported by the IEC 1131-3 standards.

The engineering tool offers the following functions:

- By selection from a menu, functions can be selected and placed in the working window.
- Outputs and inputs can be connected graphically.
- When shifting functions, the connections are dragged.
- Function configuration and parameter setting.
- Downloading of the engineering into KS 98-1
- Adjustment management.
- Long-term storage of various engineering on hard disk.

The connection from the programming PC to the multifunction unit KS 98 is made via USB or Ethernet with commercially available cables.

#### 2.1.1. Scope of delivery

The latest version of the engineering tool is available for download at [www.west-cs.de](http://www.west-cs.de).

For professional use, a license must be obtained, which will be delivered in the form of a license number.

### 2.2. Installation

#### 2.2.1. Hardware and software prerequisites

To use the engineering tool, the following system requirements are required:

- A PC in usual equipment
- MS-Windows (95, 98, VISTA, XP, 7, 8, 10)


#### 2.2.2. Software installation

Download the installation from our website and run Setup.exe.

You should make a backup copy of the installation.

#### 2.2.3. Licencing

During initial installation of the engineering tool, an input mask for entry of the licence number is displayed. Unless a licence number is entered, the engineering tool starts only as a demonstration version with limited functions (saving and downloading of an engineering into KS 98 is not possible in the demonstration version). The licence number is given on the enclosed registration form. Please, keep the registration form carefully. You need the licence number in case of re-installation and when you have to make use of the technical support. Please, complete the registration sheet immediately and send it to the specified address via fax or as a copy via mail. In return, you will get technical support and regular information on product updates.

 Please, note the PMA licence conditions for software products.

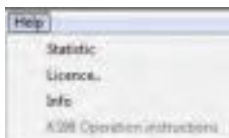


### Updates

In the download section of our website [www.west-cs.de](http://www.west-cs.de), the latest version of the software is available for download. New versions can be easily installed. The license number is stored within the system and does not need to be reentered during an update.

### Changing the licence number

A change of the licence number or licensing of a demonstration version (conversion into a full version) can be done via the menu bar



„Help“ → “Licence”.

In the window displayed after clicking on Licence, input mask ‘PMA licensing’ can be called up via Edit. Now, the new licence number can be entered in this input mask.



### 2.2.4. Software start



Starting software “engineering tool KS 98” is by double-clicking on the icon created by the installation program in program group “PMA Tools”.



## 2.3. Engineering tool operation

### 2.3.1. Fundamentals of the engineering tool operation

The Engineering Tool provides a graphical editor that allows you to view, modify and create user programs. The engineering tool differentiates between two use cases:

- Parameterize = load project, adapt parameters, transfer project to device
- Programmieren = Funktionsblöcke platzieren, verdrahten und voreinstellen

The parameterization functions are used to commission existing user programs without changing the flow logic.


For programming, the editing functions must also be enabled (lock button in the button bar)  

With the programming functions enabled, you can place, wire, and configure function blocks from the library of a selected device variant to modify or create a user program.

The following section describes the individual functions of the engineering tool.

### 2.3.2. Load projects and put them into operation

Existing user programs can be loaded into the work area via the menu item "File" or the speed-dial keys. Editing enable is initially disabled when loading projects to prevent accidental changes. Data of the individual function blocks can be viewed and adjusted if necessary. This is done by double-clicking or right-clicking the corresponding function blocks in the graphical editor. Via the menu item "Online" or the download button, the user programs can be transferred to a device.

 User programs can also be read back from a connected device.

### 2.3.3. Navigate in the editor

The editor starts in the standard view where a section of the workspace is displayed in a manageable size. The image section can be moved in three ways:

- Via the scrollbars at the edge of the editing window
- Use the mouse wheel up / down or mouse wheel + shift right / left
- With pressed Ctrl-key directly with the mouse

With the key combination "Ctrl+a" you get into the overview (overall view). A double-click at any position returns to the default view and positions the frame at this point.

### 2.3.4. Parameterization of function blocks

Via the context menu or by double-click on a function block, a parameter dialog can be opened. Parameter and configuration data of the function block are displayed in list form with description and value limits and can be edited.

Keys in the parameter dialog

- OK: Close the dialog and save the settings
- Cancel: Closes the dialog without accepting the settings
- Default: Restores the settings to the default values

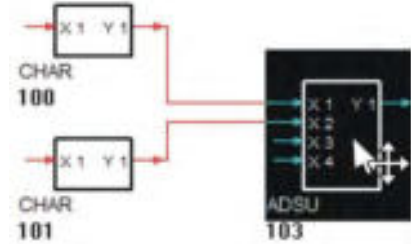
### 2.3.5. Change / create program logic

To create or change the logic of user programs, the editing functions must first be enabled. This is done via the button with the lock symbol or the key combination Ctrl + E. The following points describe the basic functions of the graphical editor.

### 2.3.6. Function block placement

A function can be selected either via menu bar 'Functions' or 'Fixed functions', or by entry of the function name in capital letters. With the engineering tool in the placement mode (mouse pointer is a hand symbol), the name of the instantaneously selected function is displayed in the bottommost status line.

By clicking with the right mouse key, the selected function is placed in the actual mouse pointer position



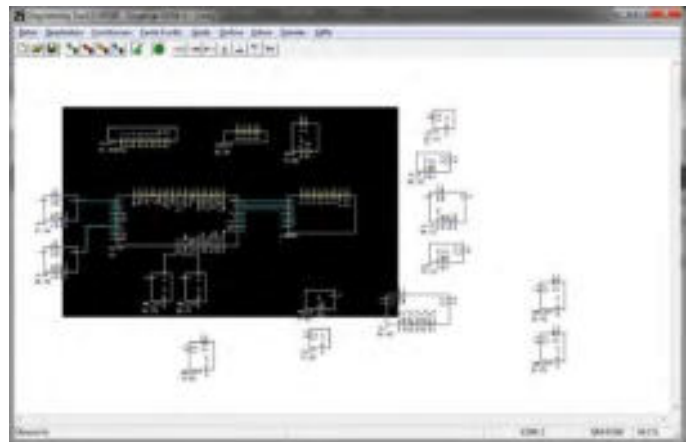
### 2.3.7. Function block shifting

After clicking with the mouse pointer on a function block, the mouse pointer is displayed inversely and can now be shifted either by means of the cursor keys or using the mouse (keep left mouse key pressed). The linked connection lines are dragged. Shifting several function blocks simultaneously is possible only in the survey display ('Edit' → 'Survey' or by pressing key 'a')

Procedure:

- Calling up the survey (Strg+a)
- While holding down the left mouse button, mark the area to be moved.
- Move the marked area while holding down the left mouse button.
- Double-click or (Ctrl + a) returns to the normal view

**i** Only the function blocks that are completely within the marking are moved!



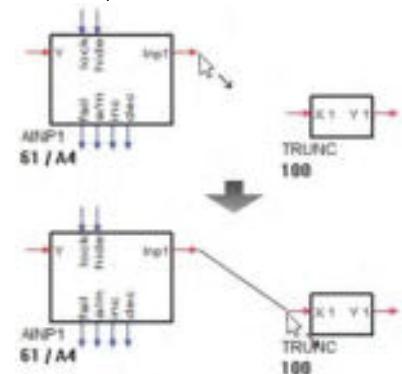
### 2.3.8. Creating connections

Connections can be made between inputs and outputs of the same signal type. The engineering tool distinguishes between analog signals (measured values, calculated values, ...) and digital signals (switch positions, simple control outputs, ...)

Procedure:

- Click on the tip of an output arrow with the left mouse button.
- With the mouse key kept pressed, draw the connection to the arrow start of the required input
- Release the mouse key

**i** The connection is displaced according to a standard algorithm.



### Connecting the signal to additional inputs

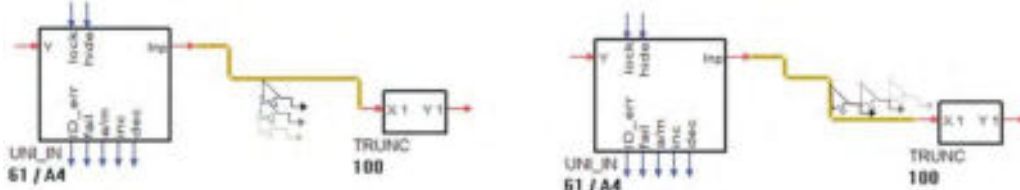
The status of an output signal may be of interest for several functions. If a connection is selected, a branch can be created directly by clicking on an input with the left mouse button while holding down the shift key. By selecting a specific segment of a connection line, it can be determined at which point the branch is connected.



## Handling connections

For increasing the graphic display clarity, subsequent handling of the connection lines is possible. The left mouse key can be used to select a line in the wiring mode (arrow symbol). The selected line is now displayed in a different colour and thickness. If this line belongs to a network (an output is connected with several inputs), the relevant lines are displayed in the colour of the selected line, but in normal thickness. Now, the individual segments of the selected line can be shifted by positioning the mouse pointer on a segment and with the left mouse key kept pressed. The segments can also be displaced using the cursor keys

If further line segments are required, the last segment can be prolonged and displaced as a new segment. In this way, max. seven variable segments can be created.



Function 'Standard connection' can be used for returning the selected line into its standard form (Function key F11).

## Changing signal source connections

In order to prevent the necessity to delete all connections and to remake them manually to the new source when changing the network wiring for another signal source, connecting a complete network to another source is possible. This is done automatically by clicking on the signal source (output) and simply clicking on the new source with key Ctrl (Strg) pressed. Thereby, all inputs are connected to the new source automatically.

## Overlapping lines

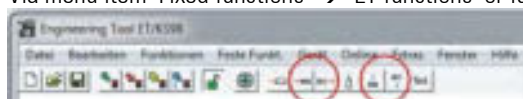
Extensive engineering frequently feature overlapping lines which do not belong to a network.

Function key F5 can be pressed for searching the engineering for overlapping lines. The first line which is found will be marked. Pressing function key F6 will search in the engineering and count how many overlappings are found. The line found last will be marked. To ensure the clarity of the engineering, these lines shall be shifted apart, until line marking after pressing keys F5 or F6 stops.

With networks, concentration of lines running in parallel is desirable. For this, seize a line at the segment which is passed by all lines to be concentrated, and move the marked segment across all network lines with the shift-key pressed. Segment shifting can be done also using the cursor keys. Concentration of the segments belonging to a network is also caused by pressing key F7. Note, however, that the catching area is limited with this function!

## Variable editor and virtual links

Via menu item 'Fixed functions' → 'ET functions' or fast exess button, data sources and destinations (analog and digital) can be selected and inserted into the engineering as special block. These sources can be assigned to variable names as with all other functions in the parameter dialogue. In the parameter dialogue of data destinations, the already defined variables are displayed in a listbox from which the required variable can be selected and assigned. These 'virtual' links are interpreted as 'full' lines in KS 98-1. Thus e.g. auxiliary functions can be placed at the periphery of the engineering without the need to draw confusing lines through the whole engineering, whereby clarity and readability are improved considerably



- ❗ However, these special blocks and their variable names are not stored in KS 98-1 and cannot be reconstructed when reading out directly from a KS 98-1. They are rather shown as full lines
- ❗ Even if you have no virtual connections that make sense for documentation purposes, to pay attention to signals via connection variables. The assigned names of inputs and outputs are also taken into account in the connection diagram.



## 2.3.9. Online-operation

### Debug



The debug mode can be enabled/disabled either via the online button (yellow double arrow), menu item "Online → Debug" or directly with F4. The corresponding communication parameters must first be set correctly via the menu (Online / Communication).

Operating data are then exchanged cyclically (approx. 0,5s) with the KS 98 or SIM/KS 98. The generated display blocks will show the relevant values.

Parameters can be changed online using the parameter dialog. After confirmation with "OK", they are transmitted to the KS 98-1. The results are displayed immediately.



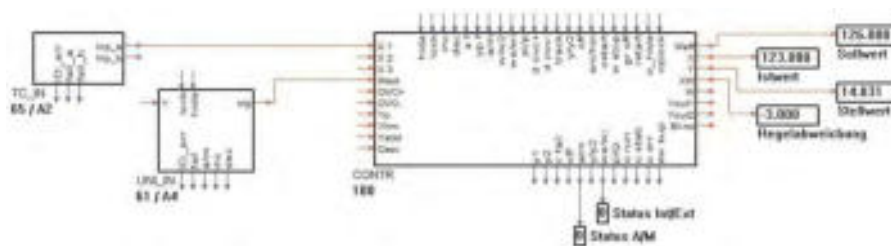
Parameter changes via the KS 98-1 front panel are not transmitted to the PC during the debug mode. For synchronize it is necessary to read back from the device.

### Display blocks (analog and digital)



In 'Fixed functions' → 'ET functions' or via fast selection button debug blocks (X-Disp and d-Disp) can be selected and inserted into the engineering as live data blocks.

The display elements can be assigned names for clarity. If an online connection to the device is active, the display elements show the current operating data that is exchanged via the connections.



Via 'Online' → 'Delete debug functions', all debug blocks can be deleted simultaneously e.g. after finishing the engineering test.

## 2.3.10. Trend function

### Survey of the characteristics

Apart from the trend function specially designed for "Controllers" (CONTR, CONTR+) of the simulation package SIM/KS98-1, additional trend windows can be implemented. Each trend is able to display 7 analog values and 12 logic states from the Engineering against a time axis. Several independent trend displays can be shown simultaneously.

As soon as the trend function(s) have been called (L1READ), and the Engineering has been transmitted to the KS 98-1 or SIM/KS 98-1, the trend dialog window is opened via the menu item (Online → TREND).

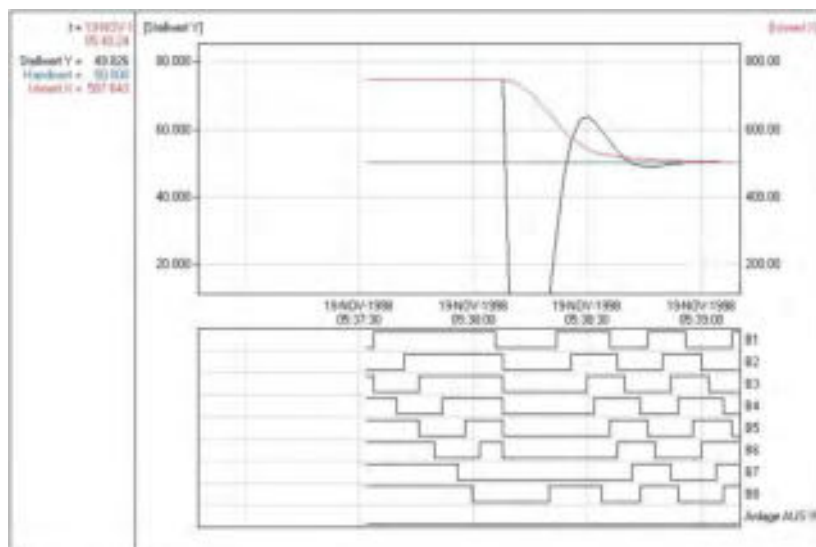
#### Trend dialog window

This window is used to select the required trend function, its duration, and the time interval. Clicking the START button initiates the trend recording function, and a display window is opened for the selected trend curve. In this way, it is possible to open several display windows



### Display window

Two different scales can be assigned to any pair of measurement values, which simplifies interpretation of the trend curves. The length of the time axis is adjusted by means of "Sampling cycle time" and "Number of measurement values" (samples) for each trend display. The time axis is either "absolute" with date and time (hh:mm:ss) or "relative" (changeable during the recording). The exact analog values are shown numerically at top left in the display window. If the cursor line is active, the analog values can be read precisely at any point of time. Trend display can be stopped (frozen) and resumed; the measurement continues in the background! By holding down the left mouse key, the cursor can be used to mark an area for enlargement. This zoom function is canceled via <View><Complete display>



### Preparations in the ET/KS 98

The trend function is an application that runs independently of the Engineering Tool. The values to be displayed are received directly from the KS 98-1 or from the simulation software SIM/KS 98-1. Data transmission is executed via the communication modules L1READ (blocks 1...20), which must first be configured. For each L1READ, 7 analog values and 12 logic statuses from the Engineering can be "connected". Usually, a single L1READ is enough to display the characteristic values of an application in relation to each other. However, up to 20 display windows can be provided. Please note that the communication load depends on the transmission cycle adjusted in each display window, and can cause problems. Make sure that the following limits are not exceeded:

<i>transmission cycle</i>	<i>Number of trend windows</i>
1s	≤ 2
2s	≤ 4
4s	≤ 8
8s	≤ 16

### Implementing the trend function

When the preparations in the Engineering have been completed, and it has been transmitted to the KS 98-1 or SIM/KS 98-1, the trend dialog box is opened directly from the Engineering Tool ET/KS 98 by means of (Online → Trend)

All the L1READ functions implemented in the Engineering, are displayed in a list box with their assigned "names", from where they can be selected for display



The lower part of the trend dialog box shows all the signals connected to the selected L1READ function, complete with block number, block title, and connection description or variable name

The lower part of the trend dialog box shows all the signals connected to the selected L1READ function, complete with block number, block title, and connection description or variable name.

Trend recording of the selected L1READ function can now be started directly with the "Start" button. The trend parameters can be changed previously by means of the "Change" button. The duration of the trend curve (visible time axis) is determined by the product of "Sampling time x sampling steps". Furthermore, you can define what happens after the selected recording duration has expired: Either the recording is terminated ("Stop at end") or continued ("Ring memory/moving"). With the second option, previous values are deleted!

## Displaying the trend curves

Clicking the "Start" button initiates the trend recording function and opens a display window. Values are displayed from left to right. Via the buttons in the dialog box, the trend display can be stopped (STOP) or moved into the background (Invisible), whereby the display window is closed. The lower part of the dialog box now shows a numeric display of the actual values of the connected variables

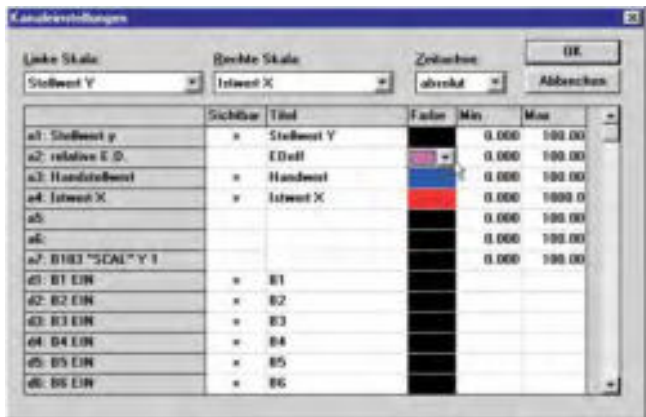
-  When using the simulation software SIM/KS 98-1, the "Turbo" mode should always be switched off
-  Before starting the universal trend, the controller trend of the SIM/KS 98-1 simulation should be terminated!

### Buttons in the display window

#### Symbol Description



- Open file
- Save file
- Transfer selected item to the clipboard memory
- Print file



#### Symbol Description



- Stop/continue trend recording
- Enable/disable cursor line
- Parameter dialog for trend adjustments
- Program information

## Adapting the trend curves

The menu items (Extras → Options) in the display window enable you to adapt the trend curves ("Channel settings"). While the trend recording is running, the "Channel settings" window can also be opened from the trend dialog box via (Options → Dialog). The "Channel settings" are stored together with the Engineering in the KS 98-1 or the SIM/KS 98-1.

The following settings can be adapted:

- Selection of the graphic curves (x)
- Description (title)
- Curve colour
- Value ranges (min / max)
- Assignment of the left / right scales to variables
- Time axis (absolute / relative)

## Calling the trend function without ET-KS 98

The settings in the trend dialog box are saved via the menu items (File → Save as ...> in the file name.dat). This enables the trend dialog box to be opened by calling trend\_di.exe, and the required trend recording to be started without the Engineering Tool.

However, a KS 98 or the Simulation SIM/KS 98 with the corresponding Engineering must be connected

It is also possible to open a display window directly by opening the corresponding file name.dat, if a link with trend\_di.exe is provided under Windows.

## Subsequent trend analysis

The contents of a display window can also be saved as a name.trd file, that can be opened later for analysis and evaluation. The cursor line, display options, and the zoom function are active.

## 2.4. Overview of all menu functions

### 2.4.1. Menu 'File'

This menu item permits standard data handling functions which are also known from other Windows. Via this menu, e.g. finishing the program is possible.



#### New

Select command "New..." in the file menu, if you want to open an empty engineering without title. Working width / height and scroll bars are set to standard values. The existing engineering is removed from the working memory.

#### Open

This function can be used for reading in already created engineering. After selection of this command, a standard dialogue box in which the relevant drive as well as the required path and file name are selected is displayed.

#### Save

This function can be used for saving an engineering created by you as a file. Storage is with the file name used when reading in.

#### Save as

This function permits saving of an already loaded project under a different name. For this, enter a new name into the field provided for this purpose. If the file extension is omitted, the file is saved automatically with extension .EDG.

#### Project info

Execution of this command is followed by display of an input mask for specification of general information on the project. Date of modification and operating version are entered automatically.

The following parts of the project info are stored in Unit:

- the first line 'Project name' (max. 45 characters can be edited freely)
- the modification date
- the operating version



After clicking on button **Frame header**, a window for text entry for the drawing header is opened.



For print-out with drawing header, tick box 'Use frame' (→ see ①).

#### Frame header

a	Text 11	Text 14	Text 17	Datum	Text 20	<div><div>PMA</div><div>Projekt- und Maschinen- Automatisierung GmbH</div><div>Information 07</div><div>34723-42000</div></div> <div><div>①</div><div>Zeichnung</div><div>Text 1</div><div>Text 2</div></div>	Projekt	Text 3	+	Text 7
b	Text 12	Text 15	Text 18	Text 19	Text 21		+	Text 8		
c	Text 13	Text 16	Text 19	OWG	Text 22		24h	Text 9		
	Änderung	Datum	Name	Notizen	Text 23		24h			

### Import

This command can be used to add stored part engineering, including all set parameters to the currently loaded. If no block number ranges are available for certain functions in the current engineering, an error message appears. By hedging individual recurring project parts in sections, they can be quickly combined to form new engineering sequences (for example programmer, parameterized controllers, etc.).

### Export

The export function can be used to save the graphical wiring or engineering data in various formats for use outside the engineering tool.

#### *Export of graphic wiring*

Creates a graphic of the engineering as a wmf-file



#### *Export of parameters and configuration data*

The parameters and configuration data of the function blocks used are stored as a text or XML file.

#### *Export a variable list*

The assigned variable names can be exported as a text file and then be edited with Word or Excel. The list contains separated by analog and digital variables:

- the name of the variable
- the block number of the source
- the title of the source
- the current port number of the source
- The meaning of the connection (e.g., Weff)

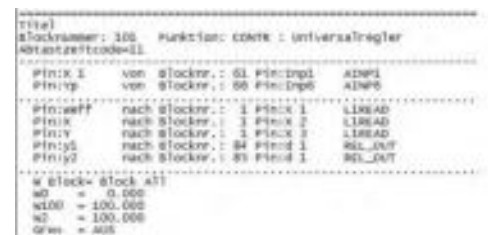
### Print

After calling up this menu item, an additional selection is displayed:



- Graphics  
Engineering print-out
- Text  
Print-out of parameter and configuration data of individual function blocks
- Connecting diagram  
Print-out of the connecting diagram (see page for connecting diagram)

Subsequently, the standard mask for adjusting printer functions under Windows is displayed. The data of the actual project are output in a standard form on the connected printer.



#### *Print-out of a section*

For printing parts of an engineering, the part to be printed must be marked in the survey mode. In the standard printer mask, click on "Mark before starting the print-out. This is only possible with graphic print-out. Printing out a section with drawing header is not possible.



## Overview of all menu functions

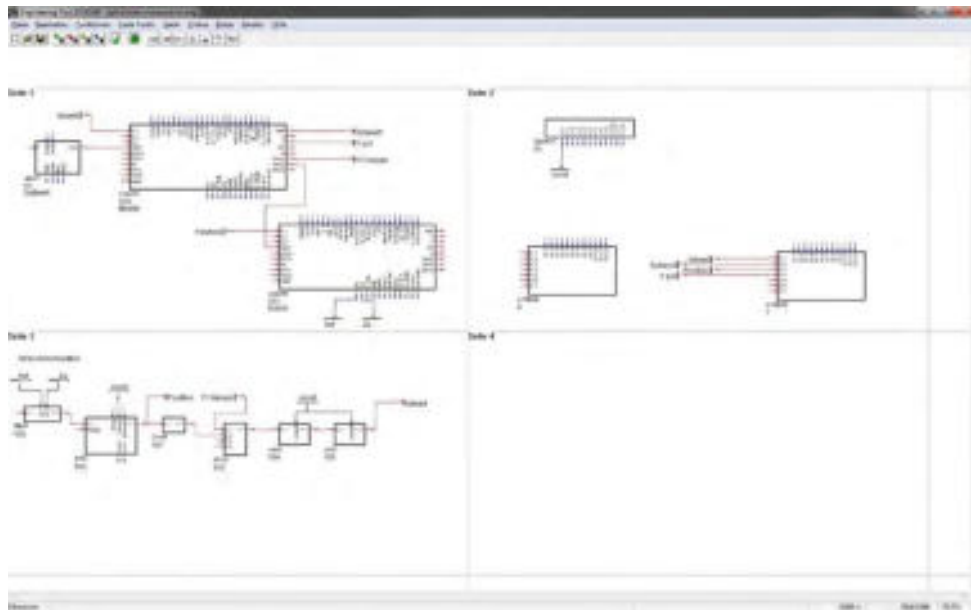
### *Graphic print-out with frame header*

Graphic print-out is possible with or without frame header. As standard, printing is without frame header. Selection is in input mask 'Drawing header'

In the drawing header, additional information such as operator, date, version no. etc. can be entered. On the left beside the PMA logo, there is free space for a customer-specific logo. The PMA logo can be removed or replaced by a customer-specific logo. The drawing header exists as a bitmap (..\Framexd (e / f) in the installation directory and can be edited with standard drawing programs.

### *Page grid in the engineering survey*

An engineering can be printed out either completely on only one page, or as a marked section. By mouse-clicking (left key) in any position of the engineering with key **Ctrl** pressed, a page grid on which the engineering can be printed out is included into the survey. The pages are numbered linewise from left to right and from top to bottom in the print-out and can be printed out with or without frame header. The pagewist printout can be controlled via the standard printer dialog. Selecting "All" will print an overview on one page. When selecting "Page 1..n" the page division is taken into account.



## Display last project files

The last 4 handled or stored projects are displayed. The project is loaded after clicking on it.

## Exit

This command finishes working with the engineering tool. Moreover, finishing the program via the system menu field is possible, as in every Windows-supported program. For this, select option "Exit" accordingly. Before finishing, you may be asked, if you want to save the changes in the project handled last. If you deny, the changes are canceled, otherwise, they are stored. Select "Cancel", or press key "ESC" for exit from the dialogue box and returning to the current project .

### 2.4.2. Menu 'Edit'

#### Undo (Ctrl + Z)

This command can be used for canceling the last action.

#### Cut (Ctrl+ X)

Removes the selected function block, places it in the buffer and activates the insertion cursor. The block can now be placed elsewhere. Wirings are deleted.

#### Copy (Ctrl+ C)

Copies the selected function block to the buffer and activates the insertion cursor. In the overview, a selected area is copied to the clipboard.

#### Paste (Ctrl+ V)

Depending on the mouse position, operating mode and display mode, the procedure is as follows:

- In the normal representation:
  - If the mouse pointer is on a free area and editing is enabled, a previously copied function block including a copy of the data of the original block is inserted.
  - If a function block is selected and the function block type matches the data in the buffer, the parameters and configurations are adopted.
- In the overview:
  - If editing is enabled, a previously copied area including all connections and all parameter settings is inserted at the bottom of the workspace. The inserted area is selected and can be moved with the mouse to the desired location.

#### Delete (Del)

Deletes the marked function or connection (Del key)

#### Parameter dialog

With this command, the parameter dialog of a selected function can be called. Double-clicking on the function block leads to the same dialog. In the parameter dialog, the parameter and configuration data of the individual function blocks are set.

#### Rename

This command can be used to change the name (title) of a function block. For function blocks with visualization, this title appears in the menu lists and in the page header. The default title of a function block is the type of block.


#### Block number

With this command the block number dialog of a selected function can be called. The block number defines the order of processing within a time slice. The block number can be changed to any valid value for this function. Normal arithmetic functions can be set to block numbers 100 to 1950. If the selected block number is already used, the numbers of all functions are moved from this block number to the next free block number. If no free block number can be found, the block number specification is rejected. For I / O function blocks, the block number defines the hardware assignment. In the block number dialog, the assignment can be selected via a list of permitted slots.



#### Timing dialog

This command can be used to call up the dialog for directly assigning the time slice of a selected function. (Function see timing).

-  The time slice assignment can be displayed below the function block. (see default setting)



#### Standard connection

A connection which was changed manually can be re-calculated automatically via this command.



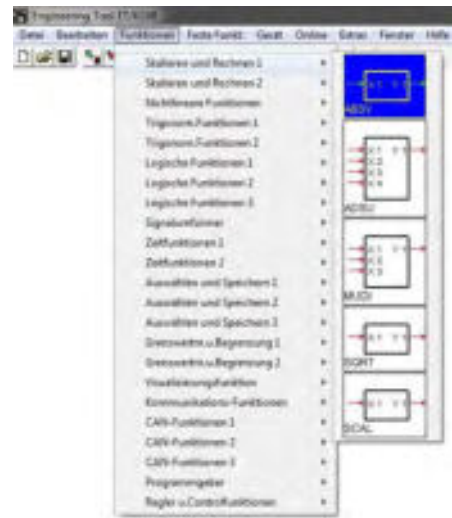
### 2.4.3. Menu 'Functions'

The menu / functions takes you to the function block library of the selected device.

#### Function Block Library

The library appears as a list of function blocks grouped together in groups. Below the groups (e.g., scaling and arithmetic), the function blocks belonging to this group appear.

By clicking on a function block this is selected. The selected function block is displayed in the status bar and can be placed in the editor with the mouse. The block numbers of these function blocks can be freely selected in the range from 100 to 1950.

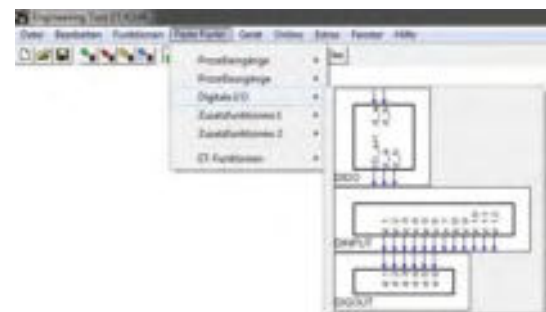


### 2.4.4. Menu 'Fixed-funct.'

The 'Fixed functions' menu can be used to select hardware-related function blocks and auxiliary functions of the engineering tool. These include, for example, function blocks that represent input modules.

#### Device functions

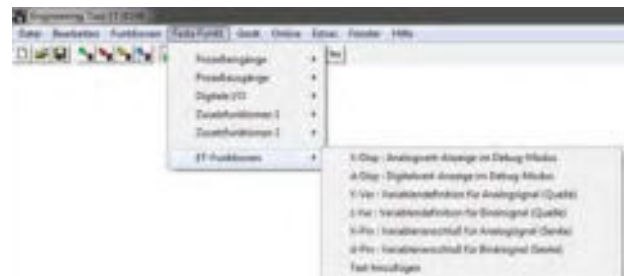
The selection and placement of the function blocks corresponds to that of the other function blocks. The block numbers are fixed or have a reference to the addressing of I / O modules which must be observed.



#### ET-Functions

To increase the clarity of an engineering, descriptive texts can be placed anywhere in the editor. A text block can consist of up to 78 characters. The text block can be moved or deleted like any other object.

Furthermore, there are elements in this area that allow indirect wiring via variables. Thus, widely used signals such as e.g. "EMERGENCY" can be easily connected to inputs of function blocks without having to pull long connections.



#### Indication elements

The indication elements (X-Disp and d-Disp) can be inserted into the wiring as a live data display. When an online connection to the device is active, the indication elements display the current data.



## 2.4.5. Menu 'Device'

### Device selection

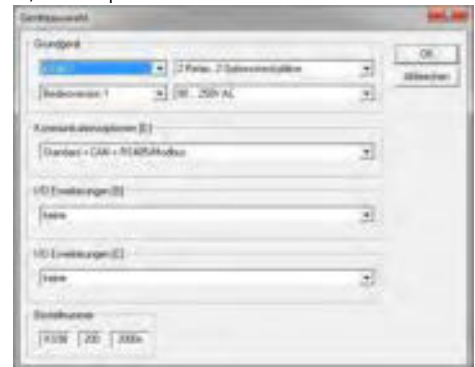


The command is used to select the device to be programmed and its variant.

Use the dropdown items to select the device options.

The order number resulting from the respective selection appears at the bottom left.

The reverse way (entering the order number of an existing device) is also possible



### Operating version

The operating version of a device is the version of the data set. It is related to the software version. If new functions / data are added with a new software version, the operating version is adapted.

### The operating version can be displayed via the device operating menu (/ main menu / general data / info /).

### Device parameters

The basic settings for the device are made in the input mask.

### These data are transmitted to the device, provided that the instrument version is correct.

### Checkbox: Freezing the outputs for download. With this field enabled, the engineering is prepared in such a way that during the next download to the multi-function unit, the outputs are frozen in their present status. With the field disabled, the outputs are switched off during this time. This means that the selected function only becomes effective after the next download



### CANparameter

The menu item "CANparameter" can only be selected, if "KS 98-1, CAN I/O extension" has been enabled during device selection

In the window, in addition to the address and the communication speed, it is also set whether the device is the network manager "CAN-NMT master", or it is only a reactive communication user "CAN slave".



### It is important to ensure that the CAN baud rate is set to the same speed throughout the network. Speeds between 10 KB and 1 MB are available (default is 20 KB).

### If communication is to take place between a KS98 and external I / O extension modules, KS98 is configured as NMT master. The master automatically receives the address 1.

### For the cross-communication between several KS 98 with each other, it is necessary that one KS98 be configured as a master and the others as a slave.

### Password (F2)

To protect a configured device against unauthorized reading and changing of the user program, a password can be defined. The password can be set and changed via this menu item. Entering the password during transmission is also possible.

## 2.4.6. Menu 'Online'

### Communication

This menu item can be used to select the interface via which communication with the device is to take place. For serial connections, baud rate and device address can be set, for Ethernet IP address and subnet. USB connections are made via virtual COM ports. The assignment of the COM port number is created during the driver installation and can be found in the Windows Device Manager. The checkbox "Transfer address and baud rate" refers to the communication parameters set in the "Device / Device Parameters" dialog.

### Project PC ← KS98

After calling up this menu item, an additional selection is displayed:

#### Engineering

Read back the complete engineering from the device.

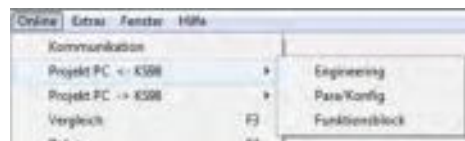
#### Para/Config


Reading back the configuration and parameter data.

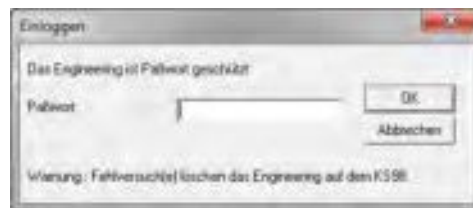
To ensure error-free reading of the data, the engineering in the device must match that in the engineering tool.

#### Function block

Configuration and parameter data read-in of a function block marked in the engineering.

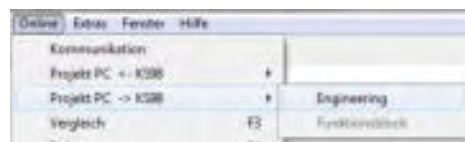


 With password protection of an engineering, dialog box "Log in" (see Fig.: 45) is displayed. This dialog box asks you to enter the password for the existing engineering. Exceeding the " number of permitted faulty attempts " during the password query deletes the engineering in the multifunction unit and renders the device unusable for the user.



### Projekt PC → KS98

After calling up this menu item, an additional selection is displayed:



#### Engineering

After selecting this menu item, a dialogue box is displayed.

When clicking on the OK button, the actual engineering is transmitted into KS 98 without password protection. The project stored in the instrument so far is overwritten.



When clicking on button "new password" in the dialogue box, the password dialogue is opened. Enter password, password mode and number of permitted faulty attempts

When clicking on button OK, the current engineering is transmitted into KS 98 with password protection. Thereby, the project stored in the unit so far is overwritten.



## Overview of all menu functions

The password mode determines the access privilege to the KS 98 data via the interface

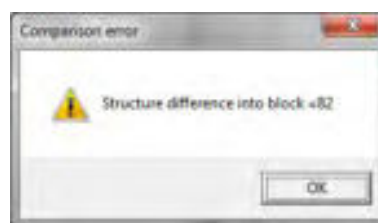
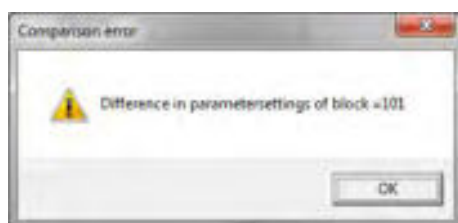
	Structure	Configuration	Parameter	Operating data
Reading and writing of operating data	✓	✓	✓	✗
Reading and writing of I / O and trend data	✓	✓	✓	✗
Reading and writing parameters and display texts	✓	✓	✗	✗
Reading and writing configurations	✓	✗	✗	✗
Reading and writing the structure data	✗	✗	✗	✗

### Function block

Transmission of configuration and parameter data of a function block marked in the engineering.

## Comparison (F3)

Permits comparison of the active engineering with the contents of the connected KS 98-1 or of the simulation. The operation is started immediately after clicking on this item. The comparison result is displayed in a message window. Possible messages are shown in an own window.



## Debug (F4)

Establishes an online connection to the device. This activates the cyclical display of operating values in the configured display blocks.

## Deleting indication elements

If desired, this function can be used to remove all indication elements from the engineering after a test session has been completed.

## Trend

See description of 'Universal trend function' (chapter 2.3.10 from page 57 )

## Set status "pwrchk"

The STATUS function provides a digital output "pwrchk" (power fail check), which goes to "0" when the power returns after a failure. By means of the function "Set status pwrchk" this output can be reset to "1", thus enabling a functional test of the engineering to be triggered after a power failure.

## 2.4.7. Menu 'Extra'

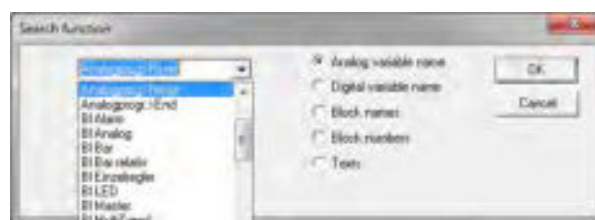
### Language

The engineering tool language can be selected.



### Search

Selecting this menu item opens the window "Search function". It is possible to search for existing elements in the different groups. All possible items are displayed in a list box, where they can be selected. If the search is successful, the corresponding part of the engineering is displayed, whereby the identified element appears inverse.



### Reorg, block no

Subsequent deletion of functions causes gaps in the list of occupied block numbers. With the menu item (/ Edit / Reorg Block Nr) a coherent renumbering can take place.

With the parameter "Free block numbers" you can leave a gap for new function blocks at the beginning.

### Enlarge sheet

With a very extensive engineering, increasing the window greater may be necessary.

### Basic setting


Two display modes are possible. The hatched mode is recommended for print-out on black-and-white printers or displays (e.g. laptop). In all other cases, the colour setting should be used.



## 2.4.8. Menu 'Window'

### Overview (button a)

The overview shows a representation adapted to the screen size the entire engineering. Press the key again to return to the working view (default 100% view).

 By double-clicking on a section in the overview, this section is displayed in normal view.

In the button bar, the view is represented by icons, pressing the button switches to the corresponding view.

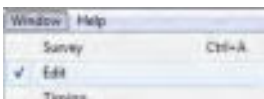
Working view



Overview



### Edit



If selected (checked), the graphical editor is enabled

Editing released



Editing disabled




### Timing

This command calls up the timing dialogue in the normal view. This dialogue can be used for determination of the time slot assignment for each function block. Time slot assignment can be done also from the parameter dialogue. In the survey, a handling simulation which indicates the order in which the function blocks are calculated is called up.

The timing dialogue shall indicate the time consumed by the function blocks in the time slots, as well as the handling sequence. Therefore, two different time slots selection functions are available. The timing sequence is displayed in the bottommost line, whilst the individual time slots are displayed at the top. If a scanning time of a time slot is selected, all function blocks which are calculated in this time slot part are displayed in the left box. Thereby, the order corresponds to the timing sequence. The right box contains all function blocks which are not allocated to a time slot so far.

If a function block is selected, it can be shifted into this time slot by mouse clicking on the relevant field. Clicking outside a time slot will remove it from a time group without re-allocating it. If several function blocks are to be moved simultaneously, they can be selected by means of or with the left mouse key.





-  As default, the engineering tool assigns the newly placed function blocks to the time slot with 100 ms scanning time.
-  The total of calculation times of all function blocks per time slot must not exceed 100%. A calculation time of a time slot exceeding 100% is displayed by a colour change (red) in the timing dialogue. In this case, the function block repartition must be changed.
-  The survey display shows a simulation that indicates in which sequence the function blocks are calculated. The sequence is either shown automatically, or is defined individually by the user via the keys v (= forwards) and r (= backwards).

### Error

The menu item allows a status window to be displayed or put back in the background if errors have occurred during loading or when switching over the operating version.

### Connecting diagram

Depending on the selected hardware version and the inputs and outputs used, a connection diagram suitable for the user program is dynamically created and displayed. The inputs and outputs are marked with the symbols of the set sensors and the names of the signals in the engineering.

-  A signal that is linked to a variable gets the name of the variable
-  A signal without tag connection receives the pin name of the IO function block

## 2.4.9. Menu 'Help'

### Manual

Opens the manual in PDF format.

### Statistic

After executing this command, a status window opens in which general engineering information is displayed.

### Licence

Information on your entered licence number required with questions you may have is displayed in this information window. A new licence number can be entered via button Change.

### Info

If the Info command is selected in the Help menu, an info window appears showing the version number of the engineering tool. If you have any questions about the engineering tool, please contact our technical support for this version number.

## 2.4.10. Attachment

### Function keys



Calling up the help ...

- General descriptions of the ET/KS 98 operating principle.
- Survey and description of the library functions (with the function block selected or the parameter dialog box opened). Prerequisite: The checkbox for help must have been clicked during installation!



Calling the password dialog.



Comparison of the engineerings in KS 98 and ET is started.



Debug mode is activated  
KS 98 or SIM/KS 98 must be connected!



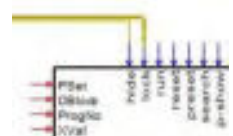
Search line overlapping.  
During the wiring mode, line overlappings are searched. The first overlapping is displayed (shown marked). In the upper left corner of the screen, either Account=0" (result negative) or Account=1" is displayed.



Search all line overlappings  
The overall engineering is searched for overlappings. Overlappings found are displayed shortly on the screen, but only the last overlapping found is displayed continuously.



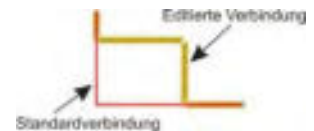
Unify adjacent lines  
Lines which belong to one connection, with only a few parallel pixels, can be unified with F7. Thereby, a line segment must be selected. (Shifting using the mouse may be not pixel-exact, however, exact superposition is also possible by means of the arrow keys.).



Line colour / -type of logic connections  
On the screen, distinction of analog and logic connections is easier in colour. In a print-out (black and white), distinction by a dashed line is better for clarity. Switch-over is always possible by pressing F9.



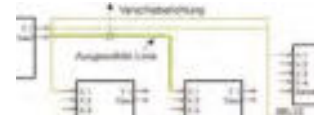
Create standard connection  
Connections of two dots are drawn automatically as short as possible at right angles (standard connection) and can be edited manually. F11 transforms a selected edited connection into a standard connection.



Language selection  
Language selection (German/English) for the engineering tool operator interface (menus, dialog boxes, etc.) is possible during working in the main menu. However, the language of help texts for KS 98-1 functions can be selected only during installation!



... Unify line segments  
When shifting a line over other lines which belong to the same signal source with the Shift key kept down, these lines are dragged and superposed automatically after releasing the shifted line (start segments cannot be shifted!). Closely adjacent lines are superposed automatically by clicking on a connection with the shift key kept down.


















Multiple connections  
A signal source can be connected with several inputs by marking an already existing connection and clicking on further inputs with the Strg key kept down. Thereby, the screen section can be shifted to the relevant position using the scroll keys, if the input to be connected is out of the visible area.



Seitenraster  
A mouse click at any place on the screen with key pressed, will draw a grid that represents the print pages. In the print-out, the pages are numbered linewise from left to right and from top to bottom and can be printed out with or without frame header.

## Mouse key functions

	Left mouse key 	Right mouse key 
<i>Standard view</i> without editing permission 	Double-click on a function block; → Opens the parameter dialog  Click on a function block; → Marks the function block Holding on vacancy plus Ctrl; → move the workspace	Click on a function block; → opens a context menu with reference to the function block  
<i>Standard view</i> with editing permission 	Double-click on a function block; → Opens the parameter dialog  Click on a function block; → Mark the block  Sticking to a functional block; → move the function block Click on a line; → Line is marked Hold tight the end of an exit; → Connection line to an input can be pulled Holding on to a line; → move the line segment Holding on vacancy plus Ctrl; → move the workspace	→ Click on a free space; → Opens a context menu for placing new items  Click on a function block; → opens a context menu with reference to the function block.        
<i>Overview display</i>	Hold on and pull; → Marks an area for joint editing Double click; → Change to normal view (parameterization or programming mode)   <b>The position of the mouse pointer determines which position of the engineering is to be displayed in the work area.</b>	 


## Function of the scroll wheel

The working area can be moved with the scroll wheel.

- Scroll wheel alone moves up and down
- Scroll wheel + shift key moves right and left

## Tips und tricks

- Block selection  
Insiders enter the short-form name of the required function (e.g. ADSU) followed by acknowledgement to avoid the detour via the menu bar. The status line shows that the block is in the cache. If the desired function already lies on the work surface, it is sufficient to selecting and de-selecting it again in order to place the block type in the buffer. Then you can place the function with Ctrl + V.
- Search  
Entry of a block number (displayed in the upper left of the screen) and acknowledgement with Enter displaces the screen and displays the searched function block with marking (functions also in the survey display)

- **Parameter setting**  
Double-clicking on a function block opens the parameter dialog. In the dialog there are buttons for copying the parameters and to return to the factory settings.
- **Line segments**  
6 other line segments can be inserted into the last line segment (before the destination input), if the connection is selected. For this purpose, seize the last line segment before the input and draw it into the required direction.
- **Calculation sequence**  
The survey display can be used to show the calculation sequence. The sequence is indicated by consecutive marking of the function blocks. Start/stop of the marking procedure by means of the key "t". While the procedure is running, pressing key "r" stops the marking and moves back to the previous function block. Pressing key "v" moves to the next function block.
- **COM test**  
Quick communication testing is possible by transmission of an "Empty" engineering to KS 98-1.
- **Copying parameters**  
If a function block is selected during the edit mode, its parameters can be stored in the clipboard by pressing "C". If another function block of the same type is then selected, the stored parameters can be copied into the block by pressing "V". This is a particularly useful feature for all blocks with many parameters (e.g. CONTR; APROGD; ...)  
If an area with the snap frame is selected in the overview, the entire contents of the field can be copied to the clipboard with Ctrl + C. Ctrl + V copies the content to the engineering and can be placed with the mouse pointer. Parameters and inner connecting lines are adopted. External connections are cut off.  
The function can also be used to transfer data into other engineerings, provided the same operating version is involved.
- **Moving areas of the engineering**  
If an area has been selected in the survey display by means of the capture frame, its entire contents can be moved with the mouse, if the left mouse key  is held down. Parameters and internal connecting lines are redrawn automatically.
- **Termination of long-lasting functions, such as Comparison (F3) with the ESC key**
- **Connections via variables**  
To track connections via variables there is a search function. If a connection variable is selected, the data source is searched for using the "d" key. The key "r" is used to search successively all entries that refer to this variable.



### 3. Function blocks

The KS98-2 function library contains all functions which are normally used for plant operation. These include:

- Functions for calculation of mathematic formulas from simple addition to exponent function.
- Logic functions and functions for realization of control sequences.
- Numerous selection and storage functions are helpful for signal processing.
- Alarm and limit value functions are indispensable for plant safety.
- Interface functions facilitate communication with adjacent and supervisory systems.
- Functions for implementation of complex and flexible control and program sequences as well as profile control tasks meet the most exacting requirements.

The wiring principle of composed functions such as programmer, controller cascades and stepping control are explained in the relevant basic function descriptions in this manual.

Examples for the basic engineering as mentioned in this manual, and further application examples for various requirements are included on a CD as a collection of examples with detailed description, or available on request.

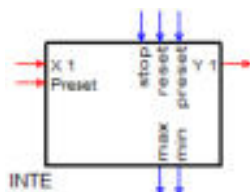
#### General KS 98-2 function block features

The features of multi-function unit KS 98-2 are determined by purposeful connection of standard function blocks with adjustable parameters.

In the KS98-2 engineering, a function block represents a black box with analog inputs (from left), analog outputs (to right), digital control inputs (from top) and control or status outputs (to bottom), as shown in the integrator diagram.

The following abbreviations are used for general inputs which mean process values and outputs which mean function results:

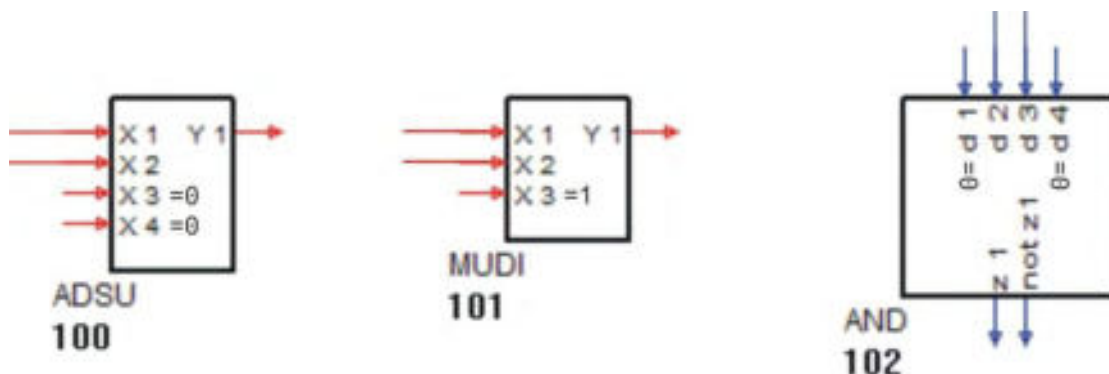
- analog inputs: X1, X2, ...
- analog outputs: Y1, Y2, ...
- digital inputs: d1, d2, ...
- digital outputs: z1, z2, ...



**i** The abbreviations for inputs and outputs with special signification are derived from their function.

## Overview of all menu functions

Not all inputs and outputs of a function block need to be connected. The following rule is applicable: open inputs are without effect. Examples: totalizer, multiplier, AND gate. In some cases, the connection of an input has an additional effect, if e.g. priority handling is concerned (programmer control inputs).



Function blocks are numbered by the engineering tool in the order of occurrence from 100 to max. 2000 as standard. Calculation of function blocks in the unit is dependent on this order. By changing the block number, the handling order is adapted. Function blocks which can be used only once or with reference to the hardware (inputs/outputs) are always within a numeric range of 0-100.

Function blocks are preset to a scanning rate (computing cycle) of 100 ms. The computing cycle can be increased to 800 in steps of 200, 400, whereby the processor load is reduced. [Detailed information is given in the ET98 operating manual.](#)

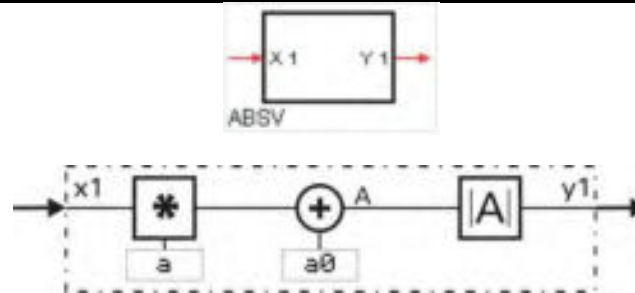
The parameters of each function block are adjustable. Apart from an individual description for documentation purposes, the majority of blocks is provided with function-specific parameters. In addition to very special ones, some parameters are quite frequent. For these general values, the same identifiers are always used:

a, b, c, d	factors without special signification
a0, b0, ... x0, y0	appended 0 as a symbol for an offset
	x0 = offset of an input, y0 = offset of an output
T, Ti	times in seconds (delays, pulse or pause duration)
Mode	These parameters are used to select function parameter setting by means of the described parameter or by means of an analog input (dynamic parameter setting)

Digital control inputs for binary selection (e.g. SELV1 for selection of 4 analog values) are normally numbered from left to right d1, d2. Note that d2 is the less significant bit despite numbers running in opposed direction. In all cases where the bit order also expresses an order of values, please, refer to the documentation of the special block in the following chapters.

### 3.1. Scaling and calculating functions

#### 3.1.1. ABSV (absolute value (No. 01))



$$y = |a \cdot x_1 + a_0|$$

The absolute value of a number is its number without polarity sign. This is the best solution for scaling a value that can't become negative, in reference to calculating time. This function block should be used, when scaling must not use a lot of calculating time.

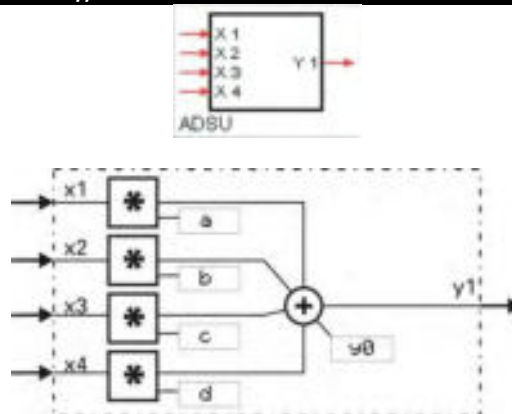
Input variable **x1** is multiplied by factor **a** (parameter). Now, constant **a0** is added. The absolute value of the resulting value is formed and output at **y1**.

Beispiel:

$y_1 = \text{ABS}(a \cdot x_1 + a_0)$       $a=5$     $x_1=2$     $a_0 = +5$    results in    $y_1 = 15$   
 $y_1 = \text{ABS}(a \cdot x_1 + a_0)$       $a=5$     $x_1=2$     $a_0 = -20$    results in    $y_1 = 10$

Parameter	Description	Range	Default
a	Multiplication factor	-29 999...999 999	1
a0	Offset	-29 999...999 999	0

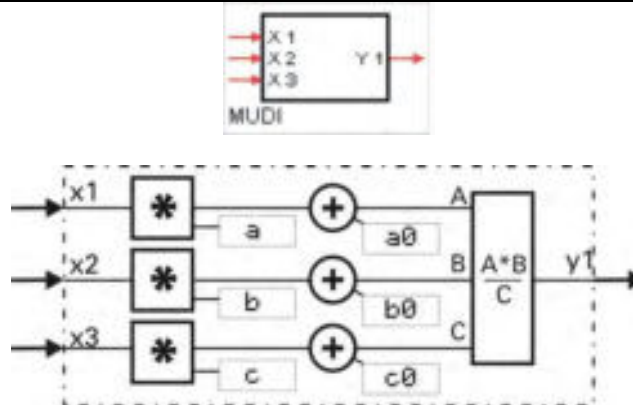
#### 3.1.2. ADSU ( addition/subtraction (No. 03))



$$y_1 = a \cdot x_1 + b \cdot x_2 + c \cdot x_3 + d \cdot x_4 + y_0$$

Input variables **x1...x4** are multiplied by factors **a...d**. Constant **y0** is added to the sum of evaluated inputs. Value "0" is assigned automatically to unused inputs.

Parameter	Description	Range	Default
a...d	Multiplication factor	-29 999...999 999	1
y0	Offset	-29 999...999 999	0

**3.1.3. MUDI ( Multiplication / division (No. 05))**

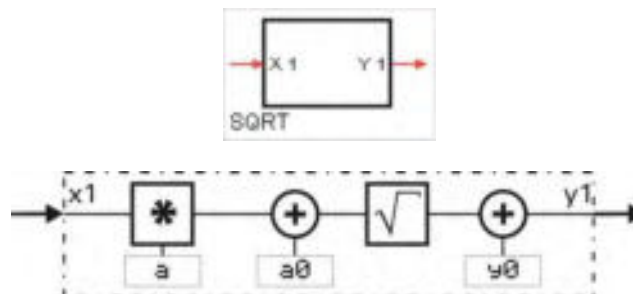
$$Y_1 = \frac{A \cdot B}{C} = \frac{(a \cdot x_1 + a_0) \cdot (b \cdot x_2 + b_0)}{c \cdot x_3 + c_0}$$

Input variables  $x_1 \dots x_3$  are multiplied by factors  $a, b, c$ . The relevant constants  $a_0, b_0, c_0$  are added. The output variable corresponds to the product.

Value "1" is assigned automatically to unused inputs.

With divisions by "0" ( $C = c \cdot x_3 + c_0 = 0$ ) output  $y_1$  is set to  $1.5 \cdot 10^{37}$ .

Parameter	Description	Range	Default
$a \dots c$	Multiplication factor	-29 999...999 999	1
$a_0 \dots c_0$	Offset	-29 999...999 999	0

**3.1.4. SQRT ( square root function (No. 08))**

$$y_1 = \sqrt{a \cdot x_1 + a_0} + y_0$$

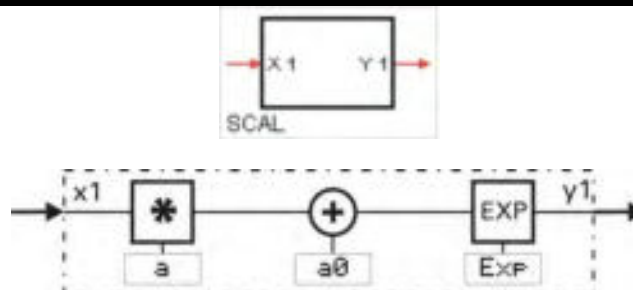
Constant  $a_0$  is added to input variable  $x_1$  multiplied by  $a$ . The result is subjected to square root extraction.

Constant  $y_0$  is added to the result of square root extraction.

If the expression under the root is negative, the square root expression is set to 0. As a result:  $y_1 = 0$ .

If the input is not connected, this is interpreted as  $x_1 = 0$

Parameter	Description	Range	Default
$a$	Multiplication factor	-29 999...999 999	1
$a_0$	Input offset	-29 999...999 999	0
$y_0$	Output offset	-29 999...999 999	0

**3.1.5. SCAL ( scaling (No. 09))**

$$y_1 = (a \cdot x_1 + a_0)^{Exp}$$

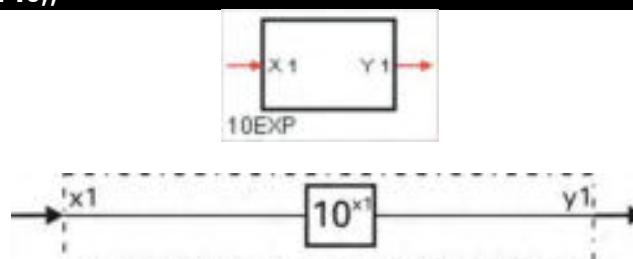
Input variable  $x_1$  is multiplied by factor  $a$  and added to constant  $a_0$ .  
The result  $(a \cdot x_1 + a_0)$  is set to the power  $Exp$ .

If  $x_1$  is not used, this is interpreted as  $x_1=0$ . With  $Exp = 0$  SCAL outputs 1.

Parameter	Description	Range	Default
$a$	Multiplication factor	-29 999...999 999	1
$a_0$	Offset	-29 999...999 999	0
$Exp$	Exponent	-7...7	1

Example:  $y_1 = \sqrt[3]{x_1^2} = x_1^{\frac{2}{3}} = x_1^{0,\bar{6}}$

This function block should be used only, if the exponential function is needed. Factor  $a$  and the offset  $a_0$  are also available with functions that need less calculating time (e.g. ADSU, MUDI, ABSV).

**3.1.6. 10EXP (10s exponent (No. 10))**

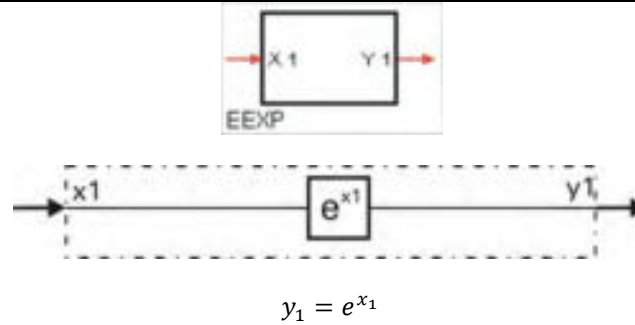
$$y_1 = 10^{x_1}$$

Input value  $x_1$  is calculated according to formula  $y_1 = 10^{x_1}$  and output at the  $y_1$  output.  
An unwired  $x_1$  is interpreted as  $x_1 = 0$  (in this case  $y_1$  is 1).

If the value at input  $x_1$  is higher than 36,7, an overflow may occur. In this case, output  $y_1$  is set to  $1.5 \cdot 10^{37}$  rather than forming the power.

**Note:**  
10EXP is the reversal function of function LG10.

### 3.1.7. EEXP (e-function (No. 11))



The e-function is calculated.

If input signal  $x_1$  is higher than 85, there may be an overflow. In this case,  $y_1 = 1,5 \cdot 10^{37}$  is output rather than forming the power.

If  $x_1$  is not wired, this is interpreted as  $x_1 = 0$  and thus as  $y_1 = 1$ .

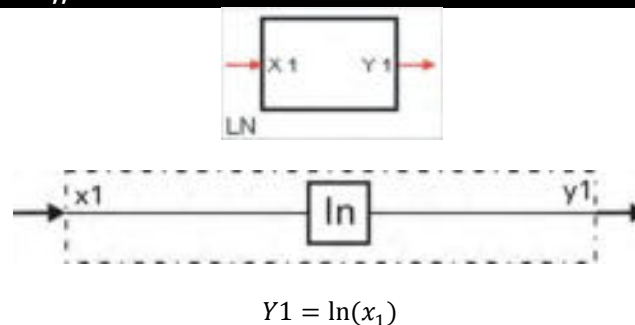
**Note:**  
EEXP is the reversal function of function LN.

Examples:

With an input value of  $x_1 = 5$ , output value  $y_1 = 148,413159$ .

With an input value of  $x_1 = 0,69314718$ , output value  $y_1 = 2$ .

### 3.1.8. LN (natural logarithm (No. 12))



The natural logarithm of input variable  $x_1$  is formed.

The basis of natural logarithms is constant  $e$  (2,71828182845904).

If  $x_1$  is not wired, this is interpreted as  $x_1 = 1$ . In this case  $y_1$  is 0.

With a negative input variable  $x_1$ ,  $y_1 = -1,5 \cdot 10^{37}$  is set

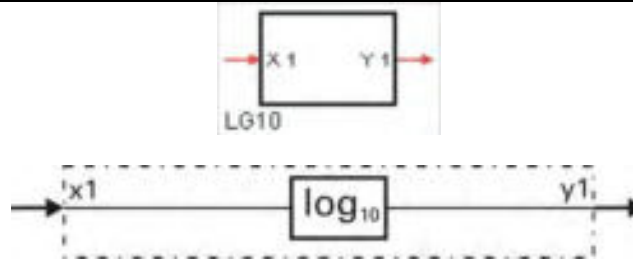
**Note:**  
LN is the reversal function of function EEXP.

Examples:

The result of input value  $x_1 = 63$  is an output value of  $y_1 = 4,143134726$ .

The result of input value  $x_1 = 2,71828182845904$  is an output value of  $y_1 = 1$ .

### 3.1.9. LG10 (10s logarithm (No. 13))



$$Y1 = \log(x_1)$$

The common logarithm of input variable **x1** is formed. LG10 provided the logarithm of a number to base 10. If **x1** is not wired, this is interpreted as **x1** = 1. In this case, **y1** is 0.

With a negative input variable **x1**, **y1** = -1,5 • 10<sup>37</sup> is set

**i** Note:  
LG10 is the reversal function of function 10EXP.

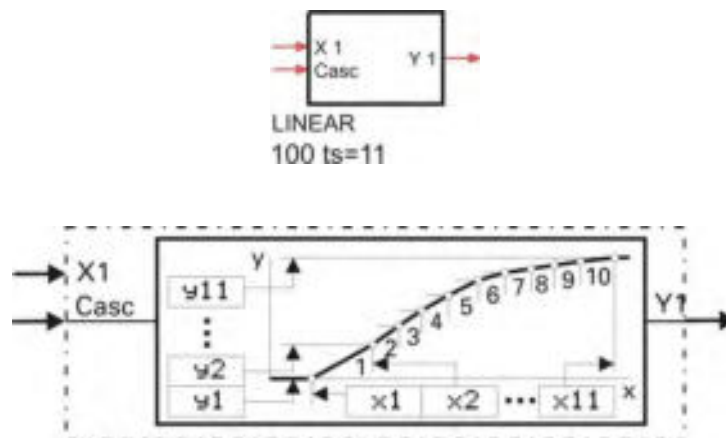
Examples:

The result of input value **x1** = 63 is an output value of **y1** = 1,799340549.

The result of an input value **x1** = 2,71828182845904 is an output value of **y1** = 1.

## 3.2. Non-linear functions

### 3.2.1. LINEAR (linearization function (No. 07))



Block LINEAR can be used for calculation of  $y = f(x)$ .

Max. 11 adjustable segment points for simulation or linearization of non-linear functions are available. Each segment point comprises input  $x(1)$  and output  $y(1)$ .

The segment points are connected automatically by straight lines. Each input value  $x1$  has a defined output value  $y1$ . With an input value  $x1$  smaller than parameter  $x(1)$ , the output value is equal to the  $y(1)$  value. If input value  $x1$  exceeds the highest parameter  $x(n)$  value, the output value is equal to the relevant  $y(n)$  value.

The condition for input of configuration parameters is that values are input in ascendant order ( $x(1) < x(2) < \dots < x(11)$ ). The end of value pairs is marked by the "OFF" value in the next input value  $x(n+1)$ .

This function block is cascable. It has 2 inputs: The 1st input provides the variable which must be linearized. The 2nd input (case) is used to connect the previous linear block.

#### Inputs/outputs

##### Analog inputs

$X_1$	Input variable which must be linearized
Casc	Cascade input

##### Analog outputs

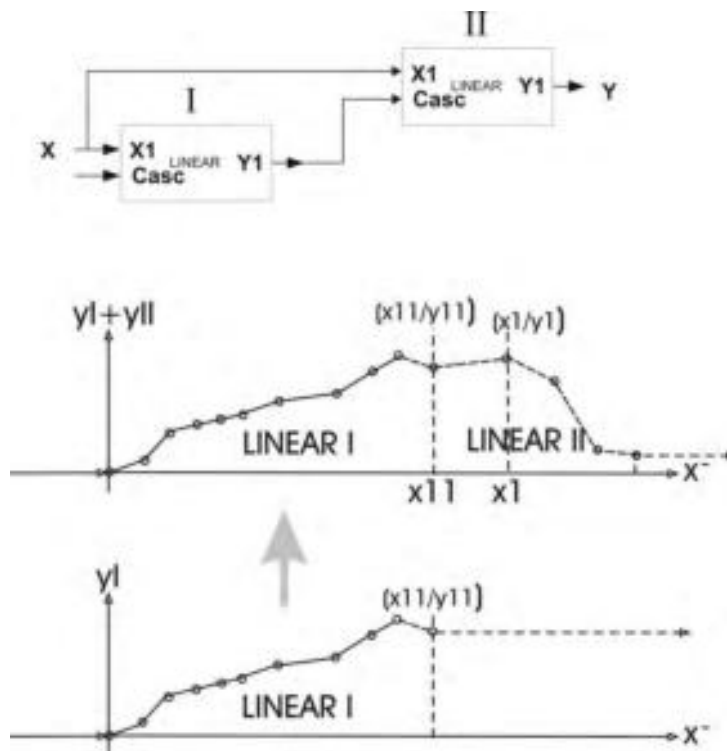
$Y_1$	Linearization result
-------	----------------------

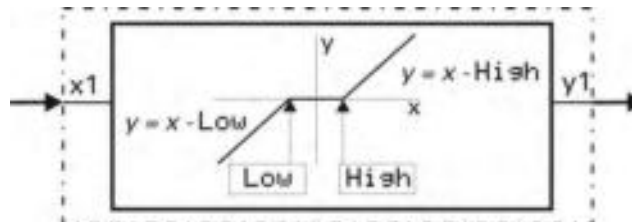
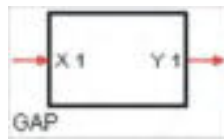
#### Parameter

Parameter	Description	Range	Default
$x(1) \dots x(11)$	Input value for segment point 1...11	-29999...999999, OFF $x(1) < x(2) < \dots < x(11)$	$x(1) = 0, x(2) = 1, x(3) = 2, \dots, x(11) = 10$
$y(1) \dots y(11)$	Output value for segment point 1...11	-29999 ... 999 999	$y(1) = 0, y(2) = 1, y(3) = 2, \dots, y(11) = 10$



**Example: linear as a cascade**



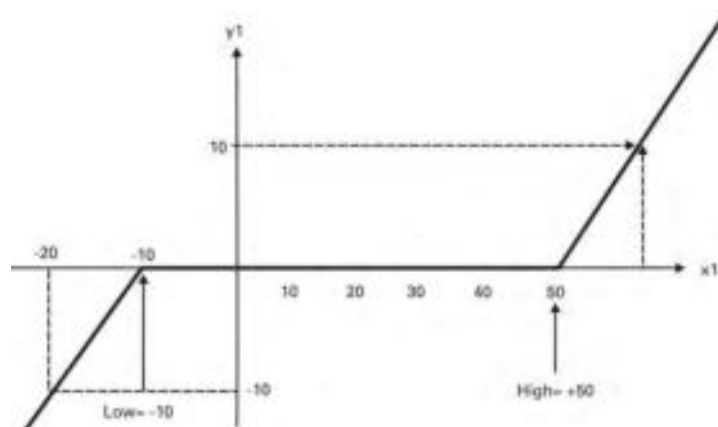
**3.2.2. GAP (dead band (No. 20))**

$$\begin{aligned}
 y_1 &= x_1 - \text{Low} && \text{bei } x_1 < \text{Low} \\
 y_1 &= 0 && \text{bei } x_1 = \text{Low} \dots \text{High} \\
 y_1 &= x_1 - \text{High} && \text{bei } x_1 > \text{High}
 \end{aligned}$$

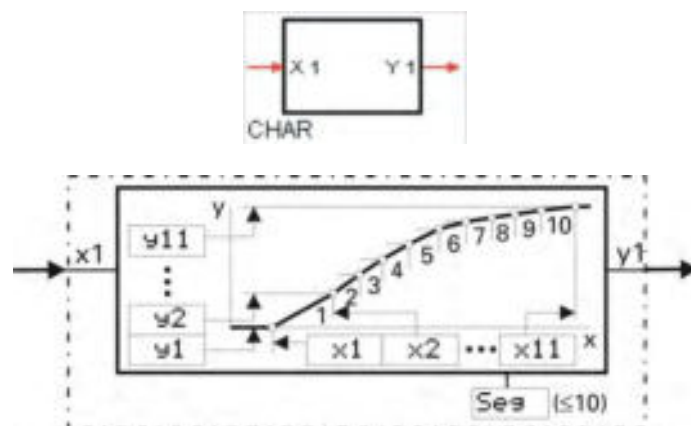
The range of the dead band is adjusted with parameters **Low** (lower limit) and **High** (upper limit). If input value  $x_1$  is within the dead band ( $\text{Low} \leq x_1 \leq \text{High}$ ), output value  $y_1 = 0$ . If  $x_1$  is not used, this is interpreted as  $x_1 = 0$ .

Example:

In the following example, -10 for **Low** and 50 for **High** was used.



Parameter	Description	Range	Default
Low	Lower limit value	-29 999...999 999	0
High	Upper limit value	-29 999...999 999	0

**3.2.3. CHAR (function generator (No. 21))**

With max. 11 adjustable value pairs, non-linear functions can be simulated or linearized. Each value pair comprises input  $x(1)$  and output  $y(1)$ . The number of value pairs is determined using configuration parameter **Seg** (number of segments +1 corresponds to the number of value pairs).

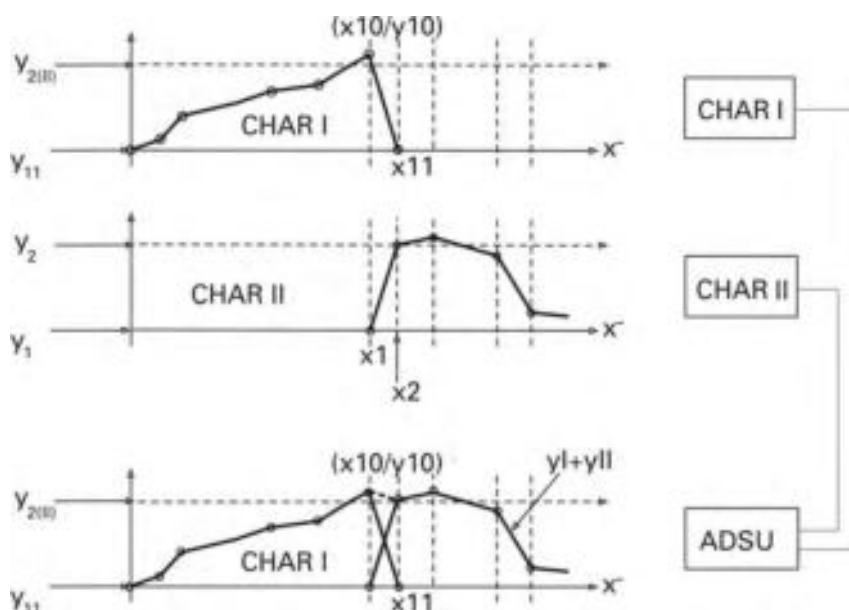
The value pairs are connected automatically with straight lines so that each input value  $x_1$  provides a defined output value  $y_1$ . If input value  $x_1$  is smaller than parameter  $x(1)$ , the output value is equal to the value of  $y(1)$ . If input value  $x_1$  is higher than the highest parameter  $x(n)$  the output value is equal to the corresponding  $y(n)$  value.

During entry of the configuration parameters, the condition is that the assigned values stand in ascending order ( $x(1) < x(2) < \dots < x(11)$ ).

Parameter	Description	Range	Default
<b>Seg</b>	Number of segments	1...10	2
<b>x(1) . . . (11)</b>	Input value for curve point	-29 999...999 999	0...10*
<b>y(1) . . . (11)</b>	Output value for curve point	-29 999...999 999	0...10*
*0 for $x(1)$ and $y(1)$ , 1 for $x(2)$ and $y(2)$ ... 10 for $x(11)$ and $y(11)$ .			

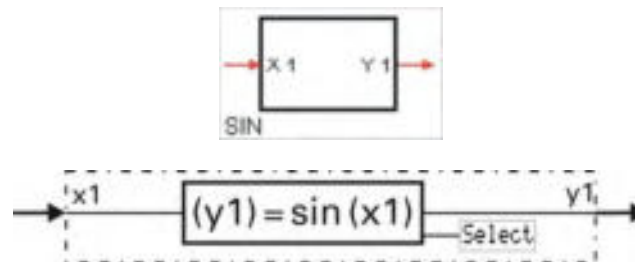
Unless one CHAR is sufficient, the following tip might be helpful:

whereby  $x_{10}$  of CHAR I =  $x_1$  of CHAR II and  $x_{11}$  of CHAR I =  $x_2$  of CHAR II



### 3.3. Trigonometric functions

#### 3.3.1. SIN (sinus function (No. 80))



$$y_1 = \sin(x_1)$$

The function provides the sinus of the input value, i.e.  $x_1$  is the angle the sinus of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian

Example degree of angle:

$$y_1 = \sin(x_1), x_1 = 30^\circ \triangleq y_1 = 0,5$$

Example radian:

$$y_1 = \sin(x_1), x_1 = 90\text{rad} \triangleq y_1 = 0,89399666$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian

$$1 \text{ rad} = 180^\circ/\pi = 57,296^\circ$$

$$1^\circ = \pi/180^\circ = 0,017453 \text{ rad}$$

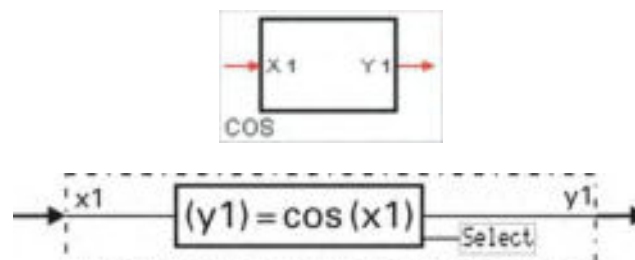
Control with the pocket calculator:

The function for the calculation in "rad" with the pocket calculator is limited to e.g.  $\pm 8$

$$\rightarrow 90/\pi = 28,6479: \sin(0,6479 \cdot \pi) = 0,893996664$$

Also during input in "°" usually a limitation is effective in the pocket calculator (e.g.  $< 1440^\circ$ )!

#### 3.3.2. COS (cosinus function (No. 81))



$$y_1 = \cos(x_1)$$

The function provides the cosinus of the input value, i.e.  $x_1$  is the angle the cosinus of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

Example degree of angle:

$$y_1 = \cos(x_1), x_1 = 60^\circ \triangleq y_1 = 0,5$$

## Trigonometric functions

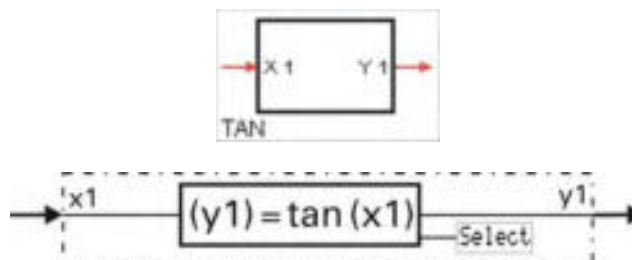
Example radian:

$$y1 = \cos(x1), \quad x1 = 45\text{rad} \quad \triangleq \quad y1 = 0,525321988$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default)	Ang. deg.
	Unit: radian	Radian

Important: When controlling with the pocket calculator see → page 84

### 3.3.3. TAN (tangent function (No. 82))



$$y1 = \tan(x1)$$

$$\text{Valid for } x1: \quad -90^\circ < x1 < +90^\circ \text{ bzw. } \left(-\frac{\pi}{2} < x1 < \frac{\pi}{2}\right)$$

The function provides the tangent of the input value, i.e. **x1** is the angle the tangent of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

For calculation clarity, the argument range is limited to the 1st or 4th quadrant ( $-90^\circ \dots 90^\circ$  oder  $-\pi/2 \dots \pi/2$ ). If input value **x1** is out of this range, output **y1**  $-1,5 \cdot 10^{37}$  ( $x1 \leq -90$  [ $-\pi/2$ ]) or  $1,5 \cdot 10^{37}$  ( $x1 \geq 90$  [ $\pi/2$ ]) is set

Example degree of angle:

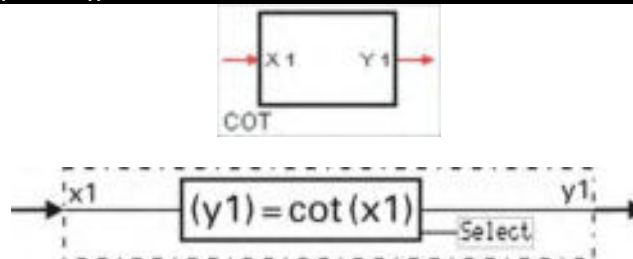
$$y1 = \tan(x1) \quad x1 = 60^\circ \quad \triangleq \quad y1 = 1,73205$$

Example radian:

$$y1 = \tan(x1) \quad x1 = 1,53\text{rad} \quad \triangleq \quad y1 = 24,498$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default)	Ang. deg.
	Unit: radian	Radian

Important: When controlling with the pocket calculator see → page 84

**3.3.4. COT (cotangent function (No. 83))**

$$y_1 = \cot(x_1)$$

Valid for  $x_1$ :  $0 < x_1 < 180^\circ (0 < x_1 < \pi)$

The function provides the cotangent of the input value, i.e.  $x_1$  is the angle the cotangent of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [ $^\circ$ ] or in radian.

For calculation clarity, the range for the argument is limited to the 1st and 2nd quadrant ( $> 0^\circ \dots < 180^\circ$  or  $> 0 \dots < \pi$ ). If input value  $x_1$  is out of this range, output  $y_1$  is set to  $1,5 \cdot 10^{37} (x_1 \leq 0)$  or  $-1,5 \cdot 10^{37} (x_1 \geq 180) [x_1 > \pi]$ .

Example degree of angle:

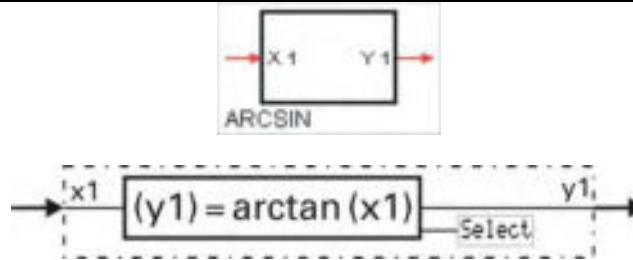
$$y_1 = \cot(x_1) \quad x_1 = 45^\circ \quad \triangleq \quad y_1 = 1$$

Example radian:

$$y_1 = \cot(x_1) \quad x_1 = 0,1 \text{ rad} \quad \triangleq \quad y_1 = 9,967$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian

Important: When controlling with the pocket calculator see → page 84

**3.3.5. ARCSIN (arcus sinus function (No. 84))**

$$y_1 = \arcsin(x_1)$$

Valid for  $x_1$ :  $-1 \leq x_1 \leq +1$

The function provides the arcus sinus of the input value, i.e.  $x_1$  is the angle the arcus sinus of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

The calculation is output as degree of angle  $[-90^\circ \dots 90^\circ]$  or as radian  $[-\pi/2 \dots \pi/2]$ . With arguments out of the function validity range, output  $y_1$  is limited to  $-1,5 \cdot 10^{37}$  ( $x_1 < -1$ ) or  $1,5 \cdot 10^{37}$  ( $x_1 > 1$ ).

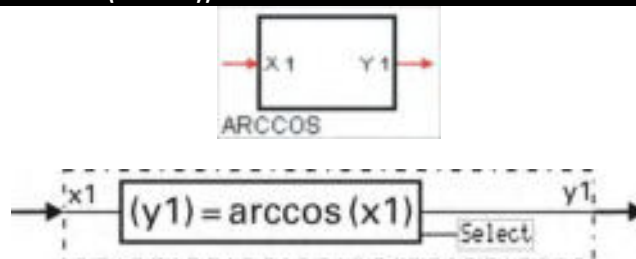
Example degree of angle:

$$y_1 = \arcsin(x_1) \quad x_1 = 0,5^\circ \quad \triangleq \quad y_1 = 30$$

Example radian:

$$y_1 = \arcsin(x_1) \quad x_1 = 1\text{rad} \quad \triangleq \quad y_1 = 1,571$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian

**3.3.6. ARCCOS (arcus cosinus function (No. 85))**

$$y_1 = \arccos(x_1)$$

Valid for  $x_1$ :  $-1 \leq x_1 \leq +1$

The function provides the arcus sinus of the input value, i.e.  $x_1$  is the angle the arcus sinus of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

Calculation is either as degree of angle [0° ... 180°] or as radian [0... $\pi$ ]. With arguments out of the function validity range, output  $y_1$  is set to  $1,5 \cdot 10^{37}$  ( $x_1 < -1$ ) or  $-1,5 \cdot 10^{37}$  ( $x_1 > 1$ )

Example degree of angle:

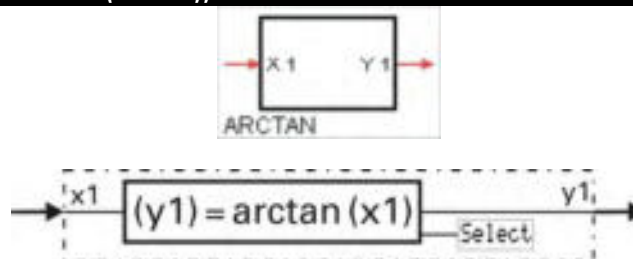
$$y_1 = \arccos(x_1) \quad x_1 = 0,5^\circ \quad \triangleq \quad y_1 = 60$$

Example radian:

$$y_1 = \arccos(x_1) \quad x_1 = 0,5\text{rad} \quad \triangleq \quad y_1 = 1,047$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian



**3.3.7. ARCTAN (arcus tangent function (No. 86))**

$$y_1 = \arctan(x_1)$$

The function provides the arcus tangent of the input value, i.e. **x1** is the angle the arcus tangent of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

The calculation is output either as degree of angle or as radian.

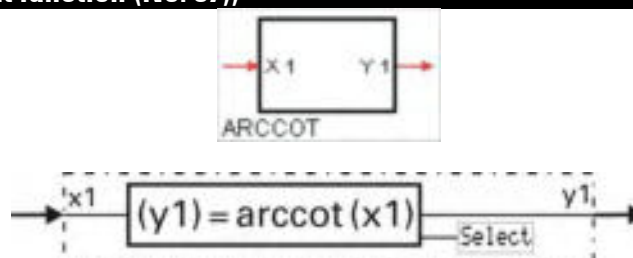
Example degree of angle:

$$y_1 = \arctan(x_1) \quad x_1 = 1 \quad \triangleq \quad y_1 = 45$$

Example radian:

$$y_1 = \arctan(x_1) \quad x_1 = 12 \quad \triangleq \quad y_1 = 1,488$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian

**3.3.8. ARCCOT (arcus cotangent function (No. 87))**

$$y_1 = \operatorname{arccot}(x_1)$$

The function provides the arcus cotangent of the input value, i.e. **x1** is the angle the arcus cotangent of which is calculated. Parameter **Select** is used to adjust, if the angle is provided in degree of angle [°] or in radian.

The calculation is output in degree of angle [0° ... 180°] and in radian [0 ... π].

Example degree of angle:

$$y_1 = \operatorname{arccot}(x_1) \quad x_1 = 1 \quad \triangleq \quad y_1 = 45^\circ$$

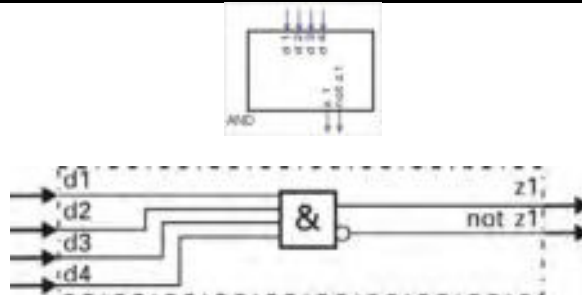
Example radian:

$$y_1 = \operatorname{arccot}(x_1) \quad x_1 = -12 \quad \triangleq \quad y_1 = 3,058$$

Parameter	Description	Controller display
<b>Select</b>	Unit: degree of angle (default) Unit: radian	Ang. deg. Radian

### 3.4. Logic functions

#### 3.4.1. AND (UND-gate (Nr. 60))

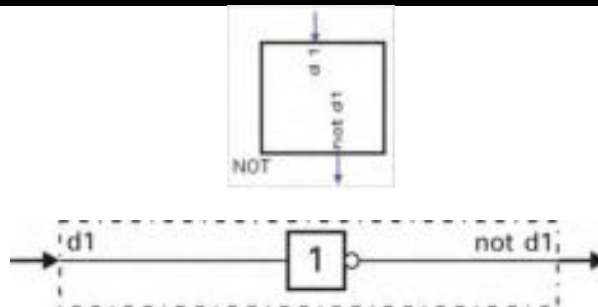


$$z_1 = d_1 \text{ AND } d_2 \text{ AND } d_3 \text{ AND } d_4$$

Logic function AND combines inputs **d1**...**d4** according to the truth table given below. Unused inputs are interpreted as logic 1.

d1	d2	d3	d4	z1	not z1
0	0	0	0	0	1
0	0	0	1	0	1
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	0	1
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	0

#### 3.4.2. NOT (inverter (No. 61))



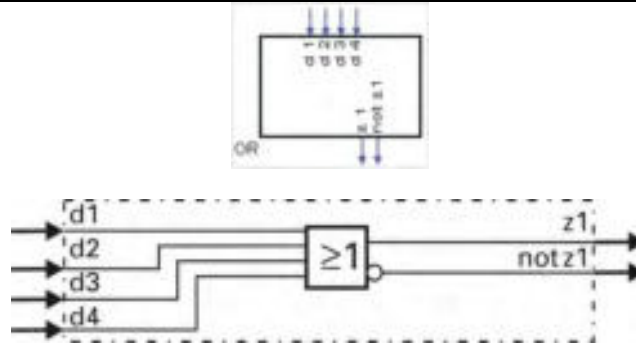
Logic input signal **d1** is output invertedly at **y1**. If **d1** is not wired, this is interpreted as logic 0.

d1	not z1
0	1
1	0

Not behaves different, dependent from

- Download e.g. POWER ON (RAM-Buffer empty)
- POWER ON (RAM-Buffer o.k.)

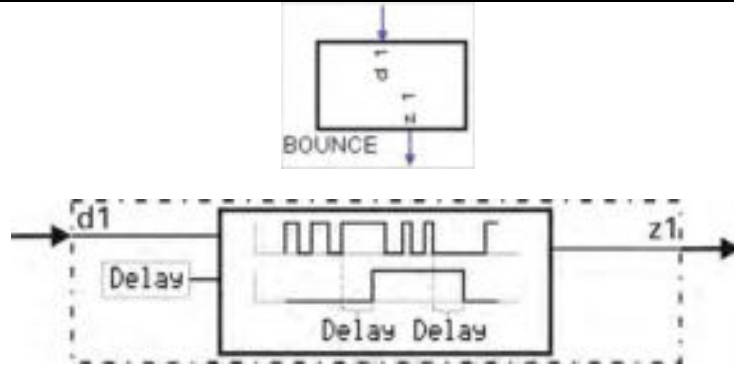
z1 at...	initialization	first calculation
Download or online → offline	z1 = 0	z1 = 1
POWER ON and RAM o.k.	z1 = 1	z1 = 1

**3.4.3. OR (OR gate (No. 62))**

$$z_1 = d_1 \text{ OR } d_2 \text{ OR } d_3 \text{ OR } d_4$$

Logic function OR combines inputs d1...d4 according to the truth table given below. Unused inputs are interpreted as logic 0.

d1	d2	d3	d4	z1	not z2
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	1	0
0	1	0	0	1	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	0	1	1	1	0
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	1	0

**3.4.4. BOUNCE (debouncer (No. 63))**

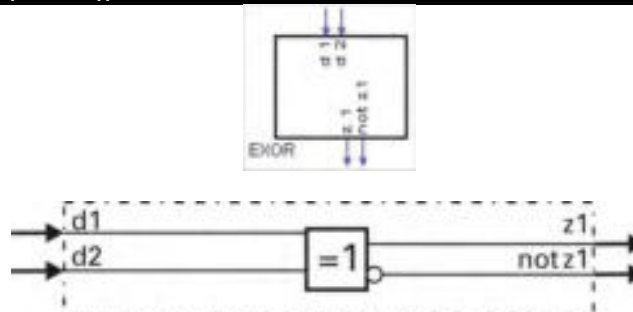
This function is used for de-bouncing a logic signal. The change of input signal **d1** is transferred to output **z1** only, when it remained constant for the time adjusted in parameter **Delay**. The time-out accuracy is dependent of the sampling interval assigned to the function.

Example:

**Delay** = 0,5s for assignment to

- sampling interval 100ms means that the signal is transferred only after  $\geq 0,5$ s.
- sampling interval 200ms means that the signal is transferred only after  $\geq 0,6$ s.
- sampling interval 400ms means that the signal is transferred only after  $\geq 0,8$ s.
- sampling interval 800ms means that the signal is transferred only after  $\geq 0,8$ s.

Parameter	Description	Range	Default
<b>Delay</b>	Switch-on and off delay time	0...999 999 [s]	0

**3.4.5. EXOR (exclusive OR gate (No. 64))**

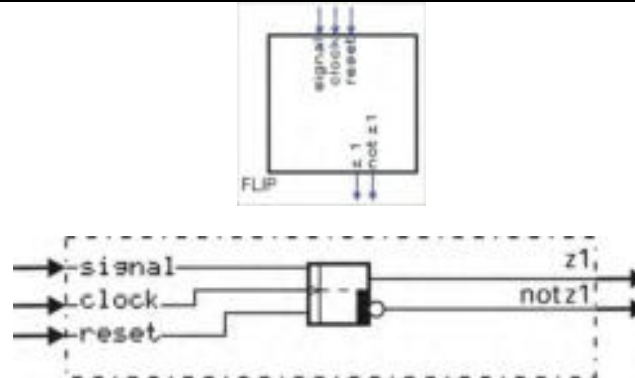
$$z_1 = d_1 \text{ EXOR } d_2$$

Logic inputs **d1** and **d2** are combined into **z1** according to the truth table given below. Unused inputs are interpreted as logic 0.

Output **z1** is 0, when the two inputs are equal (both 0 or both 1).

d1	d2	z1	not z1
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

### 3.4.6. FLIP (D flipflop (No. 65))




The digital signal status at static input **signal** is transferred to output **z1** when the signal at **clock** input clock changes from 0 to 1 (positive flank), and when input **reset** is logic 0.

With **reset** = 1, output **z1** is forced to 0 independent of inputs **signal** and **clock**.

**reset** has priority!

Input signals **signal**, **clock** and **reset** must be available at least for the duration of sampling interval  $T_r$  selected for this block (100, 200, 400 or 800ms).

In the switch-on status (initial condition),  $z1 = 0$ ! Unused inputs are interpreted as logic 0.

 This function has a "memory". This means: after power-on, it continues operating with the statuses at **z1** and **not z1**, which existed at power-off, provided that the RAM data are still unchanged.

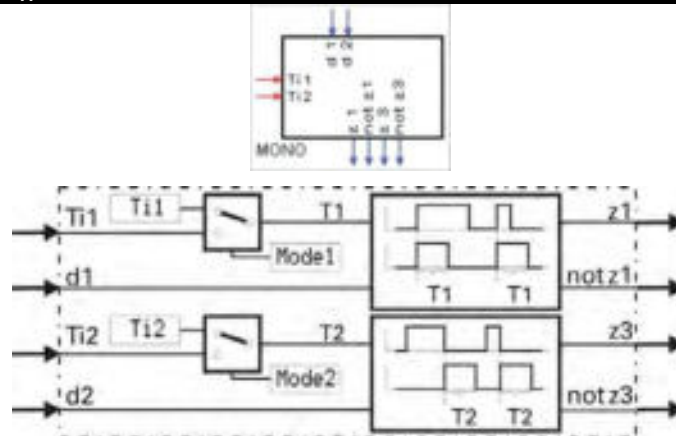
#### Inputs/outputs

##### Digital inputs

<b>signal</b>	D input - This signal is output via <b>z1</b> by the positive flank ( $0 \rightarrow 1$ ) of <b>clock</b> , when <b>reset</b> is not 1.
<b>clock</b>	Clock input - A positive flank transfers the instantaneous status at input <b>Signal</b> to output <b>z1</b> , when <b>reset</b> is not 1.
<b>reset</b>	Reset-input - sets <b>z1</b> to 0

##### Digital outputs

<b>z1</b>	Flip-Flop-output
<b>not z1</b>	Flip-Flop-output NOT <b>z1</b>

**3.4.7. MONO (monoflop (No. 66))**

The function generates a positive pulse of length  $Ti_1$  at output  $z1$ , when a positive flank at trigger input  $d1$  is detected. It generates a positive pulse of length  $Ti_2$  at output  $z3$ , when a negative flank at trigger input  $d2$  is detected.

Pulse duration  $Ti_1$  is adjusted either as parameter  $Ti_1$  or read in via inputs  $Ti_1$ . The origin of pulse duration is selected via parameter **Mode1**.

The duration of an output pulse is matched to the new values with changes at inputs  $Ti_1/Ti_2$ . With input values  $Ti_1/Ti_2 \leq 0$ , the pulse is output for the duration of one scanning cycle. The function is re-triggerable. I.e., if a new trigger condition is detected during a pulse output, the remaining pulse time to be output is prolonged to a full pulse length.

The pulse duration accuracy is dependent of the sampling time, which is assigned to the function.

Example:

$Ti = 0,9s$  for assignment to

- sampling interval 100ms means that the signal is output during = 0,9s.
- Sampling interval 200ms means that the signal is output during = 1,0s.
- Sampling interval 400ms means that the signal is output during = 1,2s.
- Sampling interval 800ms means that the signal is output during = 1,6s.

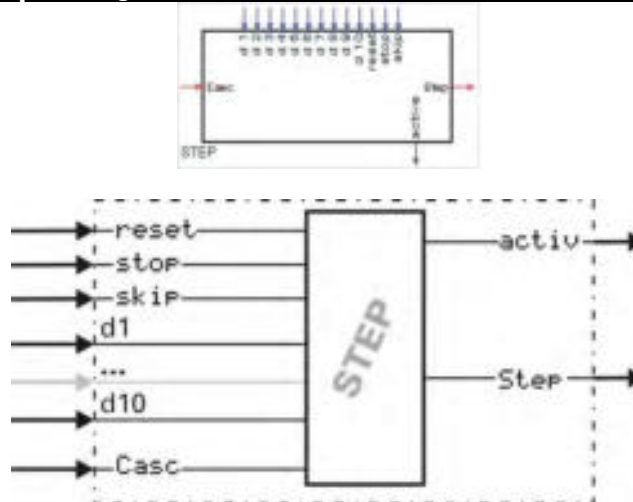
**Inputs/outputs**

Digital inputs	
d1	Triggerinput: Pulse at <b>z1</b> and <b>not z1</b> with positive flank $0 \rightarrow 1$ .
d2	Triggerinput: Pulse at <b>z3</b> and <b>not z3</b> with positive flank $1 \rightarrow 0$ .
Analog inputs	
Ti1	Pulse duration $Ti_1$ [s] of the pulse generated by d1, when <b>Mode1</b> = <b>Para . Ti1</b> .
Ti2	Pulse duration $Ti_2$ [s] of the pulse generated by d1, when <b>Mode2</b> = <b>Para . Ti2</b> .
Digital outputs	
z1	Positive pulse of length $Ti_1$ , when a positive flank at input d1 was detected.
not z1	Negative pulse of length $Ti_1$ , when a positive flank at input d1 was detected.
z3	Positive pulse of length $Ti_2$ , when a positive flank at input d2 was detected.
not z3	Negative pulse of length $Ti_2$ , when a positive flank at input d2 was detected.

**Parameters:**

Parameter	Description	Range	Default
<b>Mode 1</b>	Source of pulse duration at z1	Parameter Ti1	←
		Eingang Ti1	
<b>Mode 2</b>	Source of pulse duration at z3	Parameter Ti2	←
		Eingang Ti2	
<b>Ti1</b>	Duration of the pulse generated by d1,when Mode 1 = Para.Ti1 is entered.	0,1...999 999 [s]	1
<b>Ti2</b>	Duration of the pulse generated by d2, when Mode 2 = Para.Ti2 is entered.	0,1...999 999 [s]	1

### 3.4.8. STEP (step function for sequencing (No. 68))



The STEP function realizes the individual steps for sequencing.


The function starts with RESET at step 1 and remains at this step, until the relevant condition input d1 or the skip input is set from 0 to 1. This is followed by switch-over to step 2. The procedure for all further steps is identical.

The step number is output as a value at output Step

#### Example:

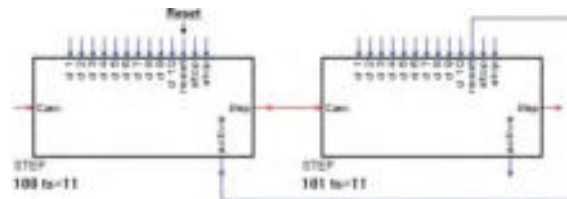
Switch-over from step 3 (**Step** = 3) to step 4 (**Step** = 4) is only after the condition at d3 was met (**d3** = 1). The condition at d4 is checked only when calling up the function for the next time. Thus immediate switch-over is prevented. As long as d3 = 0, the value of output **Step** remains 3.

Alternatively, a positive flank at input **skip** also leads to switch-over to the next step (independent of the status at input d1..d10).

 The function has a 'memory'. This means: after power-on, it continues operating with the step at power-off, provided that the RAM data are still unchanged.

When several switch-over conditions are 1 simultaneously (e.g. d1, d2, d3, d4 and d5), only the instantaneously effective input is handled. I.e. in each calculation cycle, switch-over is only by one step. For realizing a sequencing with more than 10 steps, the STEP function can be cascaded:

The wiring example shows how 2 STEP functions are cascaded. With cascading, step number 1...n is output always as a value at output **Step** of the last follow-up step.



For resetting the cascaded stepping control, reset wiring at the 1st function block is required.

#### Inputs/outputs

Digital inputs	
d1...d10	Condition inputs for switching over to the next step
reset	With input <b>reset</b> = 1, output <b>Step</b> is set to 1 (only with individual function or at the first step of a cascade). With the follow-up steps of a cascade, output y1 = the <b>Casc</b> input is set. <b>reset</b> has the highest priority of all digital inputs.
Stop	With input <b>Stop</b> = 1, the function block remains in the instantaneous step ( <b>y1</b> and <b>z1</b> remain unchanged, unless <b>reset</b> is switched to 1).
skip	This input reacts only to a positive flank, i.e. on a change from 0 to 1. At this flank the STEP function switches over to the next step without taking the status at the relevant di input into account.

## Logic functions

### Analog input

**casc**

For the first STEP function of a cascade, this input must not be connected. The RESET condition switches the entire chain to step 1 on the first STEP.

### Digital output

**activ**

**activ** =1 indicates that the STEP function is still in the active status or in reset.

**activ** =0 indicates that the STEP function has elapsed.

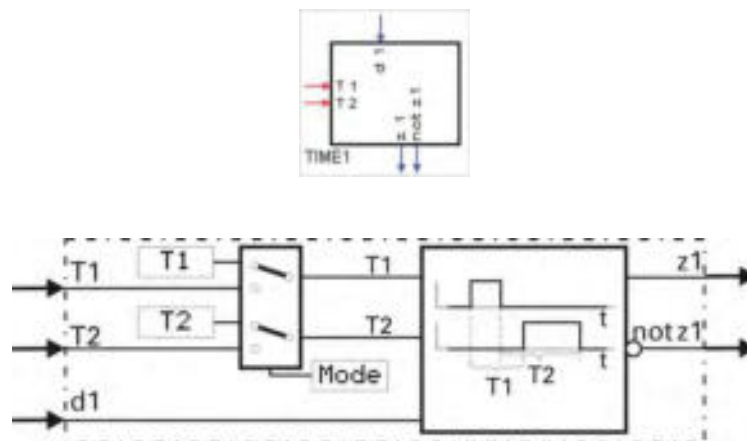
### Analog output

**Step**

The value at **Step** indicates the current step of the STEP function. With cascading, the value at **Casc** is added to this value.

*No parameters!*

## 3.4.9. TONOFF (timer (No. 69))



The function outputs the change of signal status at **d1** with a delay at **z1**.

The delay time can be adjusted separately for each change direction of the signal status! (positive and negative flank)!

With change from 0 to 1 at input **d1**, output **z1** is switched to 1 with a delay of time **T1**. With change from 1 to 0 at input **d1**, output **z1** is switched to 0 with a delay of time **T2**.

Time **T1** is adjusted either as parameter **T1** or read in via input **T1**.

Time **T2** is adjusted either as parameter **T2** or read in via input **T2**.

The time origin is selected via parameter **Mode**.

## Inputs/outputs

### Digital input

**d1**

This signal is output with a delay at output **z1** and negated at output **notz1**.

### Analog inputs

**T1**

Delay time **T1** [s], by which the positive signal of **d1** is delayed, when **Mode** = **Inputs**.

**T2**

Delay time **T2** [s], by which the negative signal of **d2** is delayed, when **Mode** = **Inputs**.

### Digital outputs

**z1**

Delayed input signal **d1**.

**not z1**

Inverted delayed input signal **d1**.



**Configuration:**

Configuration	Description	Range	Default
Mode	Source of delay times parameters T1 and T2 Eingänge T1 und T2	Parameter Inputs	←

*Parameters:*

Parameter	Description	Value range	Default
T1	Delay time T1 [s] by which the positive signal of d1 is delayed, if in <b>mode = parameter</b> is entered .	0,1...999 999 [s]	0
T2	Delay time T2 [s] by which the positive signal of d2 is delayed, if in <b>mode = parameter</b> is entered.	0,1...999 999 [s]	0

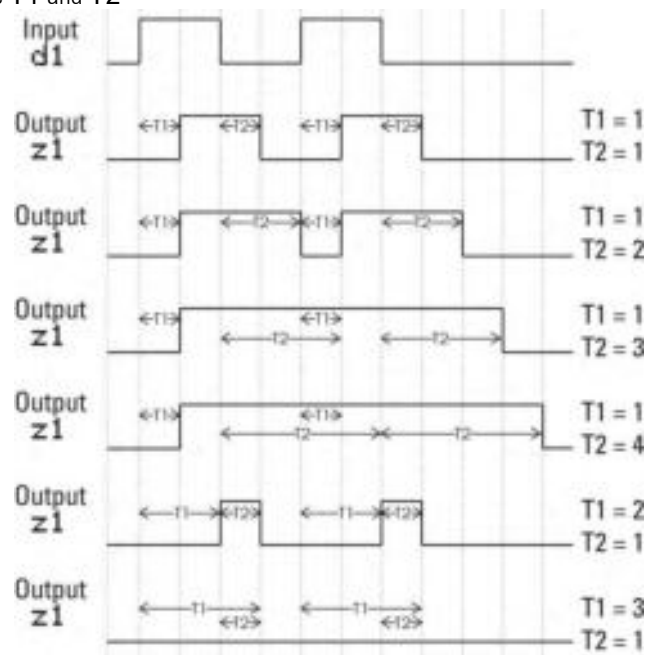
The pulse duration accuracy is dependent of the time group to which the function is assigned.  
It is an integer multiple of the sampling interval adjusted for this block (100, 200, 400, 800ms).

**Example:**

T1 = 0,7s with assignment to

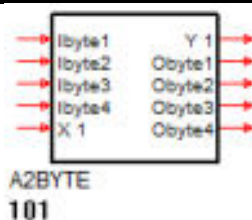
- sample time 100ms means, delay time of the positive flank is 0,7s.
- sample time 200ms means, delay time of the positive flank is 0,8s.
- sample time 400ms means, delay time of the positive flank is 1,2s.
- sample time 800ms means, delay time of the positive flank is 1,6s.

Example with different delay time T1 and T2



### 3.5. Signal converters

#### 3.5.1. A2BYTE (data type conversion (No. 02))



## Signal converters

Function A2BYTE converts an analog value (**X1**) into the individual bytes (**Oct1-4**) of a data type as used e.g. for transmission via the CAN bus ( see CPREAD / CPWRIT ). In the CAN notation, the bytes are transmitted in Intel format. Unless connected instruments are in compliance with this notation, word or bitwise exchange of the bytes may be necessary.

The function works in both directions simultaneously ( analog > bytes / bytes > analog ) with separate data type adjustment in the parameters.

### Analog inputs :

<b>X1</b>	analog input value
<b>Ibyte1..4</b>	analog input byte value 1

### Analog outputs :

<b>Y1</b>	analog output value
<b>Obyte1..4</b>	analog output byte value 1

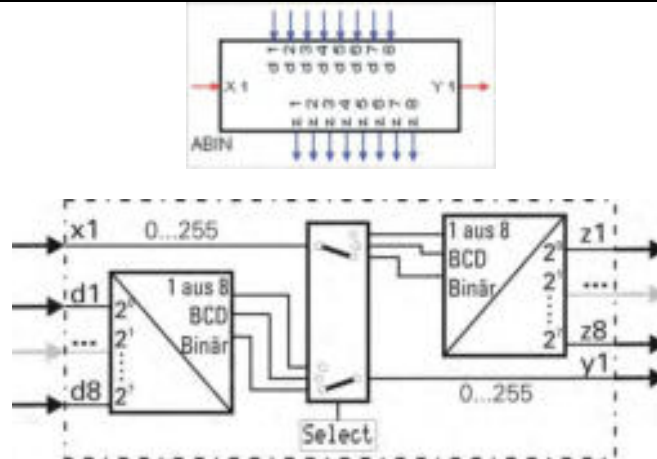
### Parameters:

Parameter	Description	Value range	Default
loct	data type of analog > byte conversion	0...999 999 [s]	0
Oct	data type of byte > analog conversion		

The following data types are available

0	1	2	3	4	5	6
UInt8	Int8	UInt16	Int16	UInt32	Int32	Float

### 3.5.2. ABIN (analog i binary conversion (No. 71))



Analog input variable **x1** is converted into a binary number, a BCD number or a selection "1 out of 8". Thereby, **x1** is always rounded off (down for values  $< 0,5$ , up for values  $\geq 0,5$ ).

Simultaneously, binary input values **d1...d8** (considered as a binary number or a BCD number) can be converted into an analog output variable. The conversion mode is determined by configuration parameter **Select**.

#### Analog/binary conversion - binary/analog conversion (**Select = ana<->bin**)

Conversion analog value into binary number:

The analog input value at **x1** is converted into a binary variable, which is output in binary form at outputs **z1...z8** ( $z1=2^0 \dots z8=2^7$ ). The range is within 0...255.

Out of the range, the output allocation is:

Input	z1	z2	z3	z4	z5	z6	z7	z8
$x1 \leq 0$	0	0	0	0	0	0	0	0
$x1 \geq 255$	1	1	1	1	1	1	1	1

Conversion binary number into analog value:

The analog input value at **x1** is converted into a binary variable, which is output in binary form at outputs **z1...z8** ( $z1=2^0 \dots z8=2^7$ ). The range is within 0...255.

BCD - conversion (**Select = ana<->BCD**) Converting a value into a BCD number

The analog input value at **x1** (range 0...99) is output as a BCD number at outputs **z8...z5** and **z4...z1**.

Example:  $x1 = 83 \rightarrow$  the output allocation is:

Input	z1	z2	z3	z4	z5	z6	z7	z8
	$2^0$			$2^3$	$2^0$			$2^3$
$x1 = 83$	1	1	0	0	0	0	0	1
BCD			3			8		

Out of the range, the output allocation is:

Input	z1	z2	z3	z4	z5	z6	z7	z8
$x1 \leq 0$	0	0	0	0	0	0	0	1
$x1 \geq 99$	1	0	0	1	1	0	0	1
		9				9		

## Signal converters

Converting a BCD number into an analog value

BCD input values at inputs **d1...d4** and **d5...d8** are converted into a floating-point number and available at output **y1**.

With a BCD number > 9 at inputs **d1...d4** or **d5...d8**, output variable **y1** is limited to 9. Out of the range, the output allocation is:

Output	d1	d2	d3	d4	d5	d6	d7	d8
	0	0	0	0	0	0	0	1
	$2^0$			$2^3$	$2^0$			$2^3$
y1=	1	1	0	1	1	1	0	1
y1=			9				9	

Converting a value into selection "1 out of 8" (**Select = ana<->1/8**)

An analog input value at **x1** (range 0...8) selects none or one of the 8 outputs **z1...z8**.

Example for conversion value (x1 = 5) into selection:

Input	z1	z2	z3	z4	z5	z6	z7	z8
x1 = 5	0	0	0	0	1	0	0	0

Out of the range, the output allocation is:

Input	z1	z2	z3	z4	z5	z6	z7	z8
x1 ≤ 0	0	0	0	0	0	0	0	0
x1 ≥ 8	0	0	0	0	0	0	0	0

Conversion Selection "1 out of 8" into analog value (**Select = ana<->1/8**)

Individual digital input allocation **d1...d8** result in an analog output variable at **y1** according to the allocated input value.

Example for conversion value (x1 = 5) into selection:

Output	d1	d2	d3	d4	d5	d6	d7	d8
y1 = 5	0	0	0	0	1	0	0	0

If more than one of inputs **d1...d8** is active, output variable **y1** is set to 0.

## Inputs/outputs

### Digital inputs

**d1 . . . d8** Digital inputs for binary value, BCD value or selection 1 out of 8.

### Analog input

**x1** Analog input for binary value, BCD value or selection 1 out of 8.

### Digital outputs

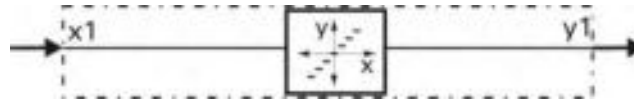
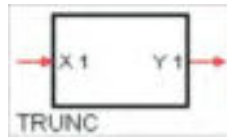
**z1 . . . z8** Converted binary value, BCD value or value selection.

### Analog output

**y1** Converted analog value.

## Configuration:

Configuration	Description	Range	Default
<b>Select</b>	Mode of conversion	analog/binary conversion and binary/analog conversion Analog/BCD conversion and BCD/analog conversion Selection 1 out of 8	ana<->bin ana<->BCD ana<->1/8 ←

**3.5.3. TRUNC (integer portion (No. 72))**

$$y_1 = INT(x_1)$$

The function provides the integer portion (integer) of input variable  $x_1$  without rounding off at output  $y_1$ .

Example:

$$\begin{array}{lll} x_1 = 1,7 & \rightarrow & y_1 = 1,0 \\ x_1 = -1,7 & \rightarrow & y_1 = -1,0 \end{array}$$

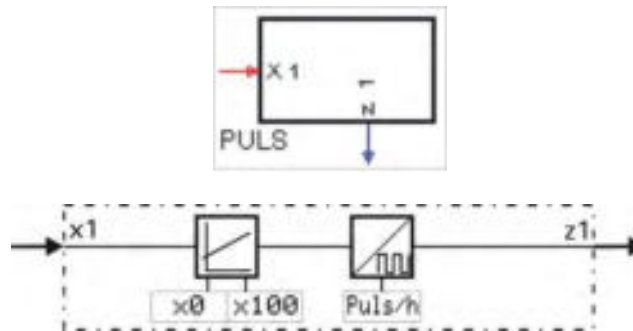
**Inputs/outputs****Analog input**

$x_1$  Input variable to be handled

**Analoger Ausgang**

$y_1$  Integer portion of  $x_1$

No parameters!

**3.5.4. PULS (analog pulse conversion (No. 73))**

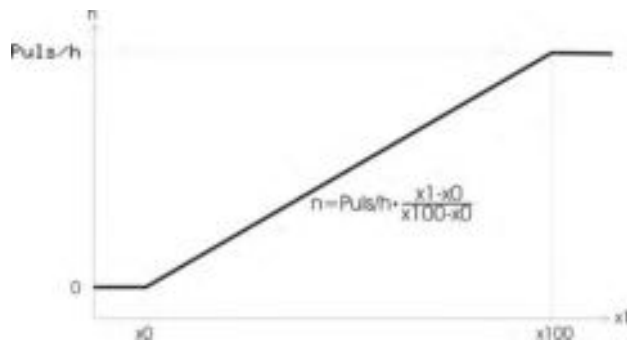
$$n = \text{Puls/h} \cdot \frac{x_1 - x_0}{x_{100} - x_0}$$

$n$  = Number of pulses per hour  $z_1$   
 $x_0$  = Parameter  
 $x_{100}$  = Parameter  
 $x_1$  = Analog input

Input variable  $x_1$  is converted into a number of pulses per hour. Parameter **Puls/h** is used for selecting the maximum number of pulses at  $x_1 \geq x_{100}$ . For  $x_1 \leq x_0$  no pulses are output.

Within range  $x_0 - x_{100}$ , input value  $x_1$  is converted linearly into pulses per hour.

**Puls/h** = max. Pulsenumber /h  
**x0** = % of Puls/h  
**x100** = 100% of Puls/h



The parameter settings result in a straight line between 0 and 3600 pulses /h according to input  $x_1$ . The pulse length corresponds to the sampling interval (100, 200, 400 or 800ms) adjusted for this block. The length of switch-off time between pulses is not always equal and dependent of the configured sampling interval.

The sampling interval allocation also determines the maximum number of pulses/hour, which can be realized. If higher values than can be output due to the sampling interval are entered in parameter Puls/h, limiting is to the maximum possible number of pulses.

Maximum number of pulses / h	
100 ms	= 18 000 Pulse/h
200 ms	= 9 000 Pulse/h
400 ms	= 4 500 Pulse/h
800 ms	= 2 250 Pulse/h

**Inputs/outputs****Analog input**

**x1** Input variable to be converted

**Digital output**

**z1** Pulse output

*No configuration parameters!*

**Parameter:**

Parameter	Description	Range	Default
x0	Span start (0 % von Puls/h)	-29 999...999 999	0
x100	Span end (100 % von Puls/h)	-29 999...999 999	1
Puls/h	Number of output pulses per hour for x1 ≥ x100	0...18 000	0

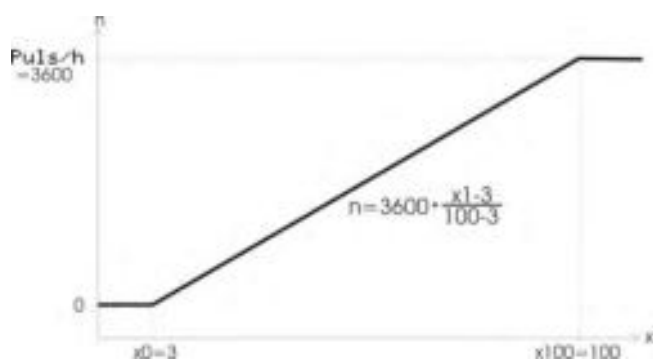
Equation for calculating the momentary impulse number of n per hour

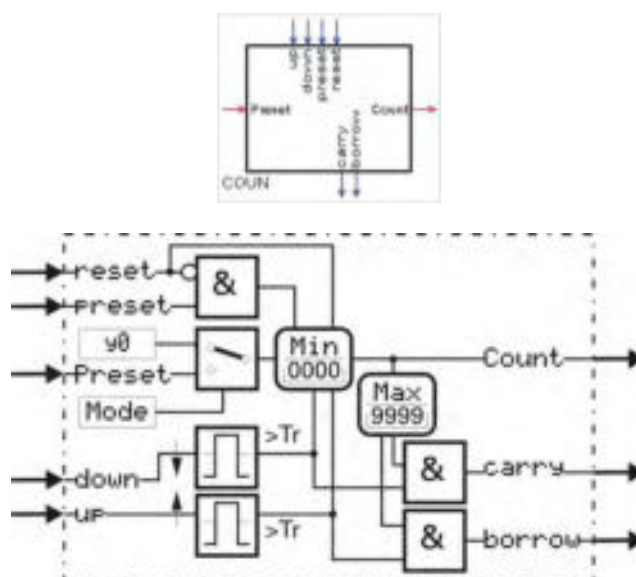
$$n = \text{Puls/h} \cdot \frac{x_1 - x_0}{x_{100} - x_0}$$

n = momentary impulse number/hour  
 x0 = Parameter. With analog input x1 & x0 no pulses are produced (area start, creeping flow suppression)  
 x100 = If the analog input is x1 & x100 n remains = constant = Puls/h  
 Puls/h = Parameter. Pulse number/hour for analog input x1 = x100

Example:

x1 = 3...100% ≙ 0...3600/h  
 x0 = 3  
 x100 = 100  
 Puls/h = 3600  
 sampling period ≤ 400 ms

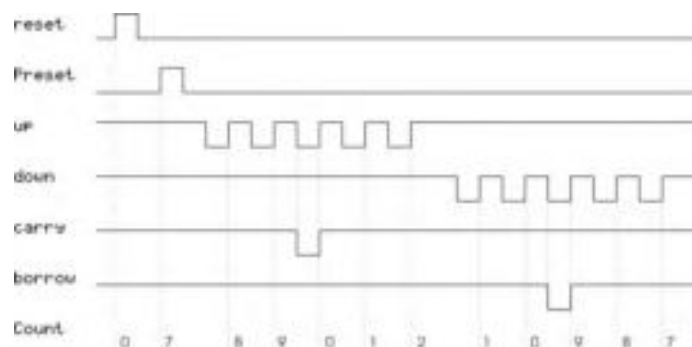


**3.5.5. COUN (up/down counter (No. 74))**

'COUN' is an up/down counter and counts the events at input up or down, which are available at the up or down input for at least the duration of the time group in which the function runs.

reset	preset	Mode
0	0	GO (default)
0	1	Preset
1	0	Reset (first run)
1	1	Reset (first run)

*Pulse diagram of the up/down counter:*




*"up, down, Carry und borrow" are inactive with status 1.*

Example: Max-limit = 9; Min-limit = 0; Preset = 7.

An unwired clock input is set to value 1 internally. If both clock inputs go from 0 to 1 signal simultaneously, counting is omitted. If one of clock inputs (up or down) are set from 0 to 1 signal, without the other one being already set to 1, counting is omitted.


If parameters for the min. or max. limit are changed during operation, the counter can be out of this new range. In order to prevent faulty functions, the counter must be set to a new, defined output status with 'reset' or 'preset'.

 The function has a 'memory'. This means: after power-on, it continues operating with counter state and internal states at power-off, provided that the RAM data are still unchanged.




### Function up counter:

At each positive flank (0 → 1) at input **up**, output **Count** is increased by 1, until the max. limit is reached. **Carry** output carry is set to 0 for the duration of the applied pulse. With the next pulse, output **Count** returns to the min. value and continues counting with the next pulses.

 If the **down**-input is wired, the **up** counter is prepared by signal 1 at input **down**. If not, counting is not possible. I.e. there must be a 1 signal at input **down** prior to input **up**, if the pulse shall be counted.

### Function down counter:

With each positive flank (0 → 1) at input **down**, output **Count** is decreased by 1, until the min. limit is reached. Subsequently, borrow output **borrow** is set to 0 for the duration of the applied pulse 0. With the next pulse, output **Count** returns to the max. value and continues counting down with the next pulses.

 If the **up**-input is wired, the **down** counter is prepared by signal 1 at input **up**. If not, counting is not possible. I.e. there must be a 1 signal at input **up** prior to input **down**, if the pulse shall be counted.

### Function reset:

A 1 signal at input **reset** has priority over all other inputs. **reset** resets the count to the min. value.

### Function preset:

A 1 signal at input **preset** has priority over inputs **up** and **down**. **preset** resets the count to the preset value. The origin of the preset value is selected with parameter **Mode**.

- **Mode** = **Para.y0** means that the preset value corresponds to parameter **y0**.
- **Mode** = **InpPreset** means that the preset value corresponds to analog input **Preset**.

With a preset value higher than the max. limit, output **Count** is set to the max. limit. A preset value smaller than the min. limit is set to the min. limit. A preset value which is not an integer is rounded off.

### Inputs/outputs

#### Digital inputs

<b>up</b>	Input for clock up - pulse count up
<b>down</b>	Input for clock down - pulse count down
<b>preset</b>	Input for the preset mode - output <b>Count</b> goes to value <b>Reset</b>
<b>reset</b>	Input for the reset mode - output <b>Count</b> goes to value <b>Min</b>

#### Analog input

<b>Preset</b>	Analog input for external preset value
---------------	--

#### Digital outputs

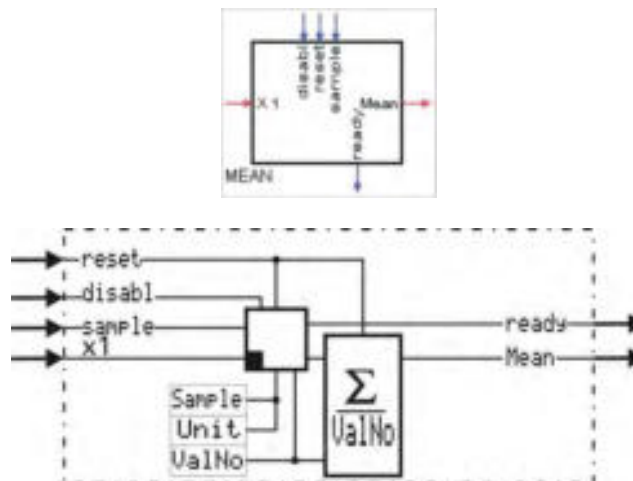
<b>Carry</b>	Carry output (clock - up)
<b>borrow</b>	Borrow output (clock - down)

#### Analog output

<b>Count</b>	Count output
--------------	--------------

Parameter:

Parameter	Description	Value range	Default
<b>Mode</b>	Source of preset-value	0: Para <b>y0</b> 1: <b>InpPreset</b>	←
<b>y0</b>	Preset-value	-29 999...999 999	0
<b>Max</b>	Max. limit	-29 999...999 999	1
<b>Min</b>	Min. limit	-29 999...999 999	0


**3.5.6. MEAN (mean value formation (No. 75))****General**

Function MEAN forms the floating, arithmetic mean value of the number (**ValNo**) of the values detected last at input **x1** for output at output **y1**.

The interval between the individual samplings (interval) is adjustable with **Sample** and **Unit**.

Unit is used to specify the measurement interval (**sec** = seconds, **min** = minutes or **h** = hours).

**Sample** is used to specify the number of 'Unit' intervals for measurement.

 With the sample input wired, the adjusted sample and unit parameter are ineffective. Only the sample-pulse is used

Example 1: mean value of the past minute with sampling per second.

**Sample** = 1 and **Unit**=**sec** → value sampling per second.

**ValNo** = 60 → the past 60 values form the mean value (1 minute)).

Example 2: mean value of the past day with sampling per hour.


**Sample** = 1 and **Unit**=**h** → value sampling per hour.

**ValNo** = 24 → the past 24 values for the mean value (1 day).

Example 3: mean value of the past day with sampling per quarter of an hour.

**Sample** = 15 and **Unit**=**min** → value sampling at intervals of 15 minutes.

**ValNo** = 96 → the past 96 values form the mean value (1 day)).

 If the **sample** input is wired, sampling is triggered by a positive flank at this input. The adjusted sampling interval is invalid.

**disabl** = 1 interrupts the sampling, **reset** = 1 deletes the mean value.

Internal calculation:

The number of input values entered in **ValNo** is stored, totalized and divided by the number.

$$y_1 = \frac{Wert_1 + Wert_2 + Wert_3 + \dots + Wert_n}{n}$$

Example: **ValNo** = 5

x1=	11	24	58	72	12
-----	----	----	----	----	----

$$y_1 = \frac{11 + 24 + 58 + 72 + 12}{5} = 35,4$$

**reset**

## Signal converters

Analog input **Mean** goes to value 0 for the duration of the applied **reset** signal.  
The stored values are deleted.

Example:

**ValNo** = 5 output **Mean** at reset:

x1=	x	x	x	x	x
-----	---	---	---	---	---

Detection that no valid values are available is made. Value 0 is output at output **y1**.

**ValNo** = 5 1st Sample after Reset:

x1=	55	x	x	x	x
-----	----	---	---	---	---

Detection that only one valid value is available is made. The only valid value **y1** = 55 is available at output **y1**.

**ValNo** = 5 2nd Sample after Reset:

x1=	44	55	x	x	x
-----	----	----	---	---	---

Detection that two valid values are available is made. The mean value of these valid values **y1** = 49,5 is output at output **y1**.

After all memory cells with a value are occupied (**ValNr** = 5), with every sample a new input value is added, the at this time oldest value subtracted and the result divided by **ValNr** = 5. The input values are shifted (like with a shift register).

## Inputs/outputs

### Digital inputs

<b>disabl</b>	The disable input interrupts sampling
<b>reset</b>	The reset input clears the memory and resets the mean value to 0.
<b>sample</b>	A positive flank (0 → 1) is used for sampling a new value.

### Analog input

<b>x1</b>	Process value, of which the mean value is formed.
-----------	---

### Digital output

<b>ready</b>	Display for an elapsed overall cycle
--------------	--------------------------------------

### Analog output

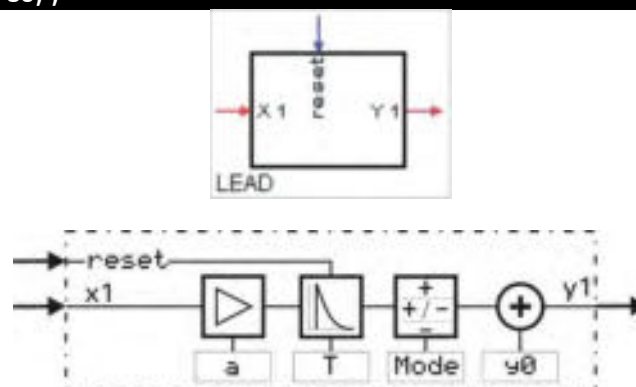
<b>Mean</b>	Calculated mean value
-------------	-----------------------

## Configuration:

Parameter	Description	Value	Default
<b>ValNo</b>	Number of values which can be aquired	1...100	100
<b>Unit</b>	Unit of time for "Sample"	Seconds Minutes Hours	←
<b>Sample</b>	Interval time for averaging	0,1...999 999	1

### 3.6. Time functions

#### 3.6.1. LEAD ( differentiator (Nr. 50) )



The differentiator forms the difference quotient according to equation:

$$y_{1(t)} = \frac{T}{T + t_s} \cdot \left[ y_1(t - t_s) + a \cdot \{x_{1(t)} - x_{1(t-t_s)}\} \right] + y_0$$

ts	sampling interval	y0	output offset	y1(t)	instantaneous y1
T	time constant	x1(t)	instantaneous x1	y1(t-ts)	previous y1
a	gain	x1(t-ts)	previous x1		

$$C = \frac{T}{T + t_s} < 1 \quad (\text{differentiation constant})$$

The complex transfer function reads:  $F(p) = \frac{a \cdot T \cdot p}{T \cdot p + 1}$

#### Inputs/outputs:

##### Digital input

**reset** = 1 causes that  $y_1 = y_0$  and the difference quotient is set to 0.  
= 0 starts differentiation automatically.

##### Analog input

**x1** Input variable to be differentiated

##### Output

**y1** Differentiator output

#### Parameter:

Parameter	Description	Range	Default
<b>a</b>	Gain factor	-29 999...999 999	1
<b>y0</b>	Output offset	-29 999...999 999	0
<b>T</b>	Time constant in s	0...199999	1

#### Configuration:

Configuration	Description	Value	Default
<b>Mode</b>	Differentiating all changes	0	0
	Differentiating only positive changes $dx/dt > 0$	1	
	Differentiating only negative changes $dx/dt < 0$	2	

### Step response:

After a step change of input variable  $x_1$  by  $\{x = x_t - x(t - t_s)\}$ , the output changes to maximum value  $y_{max}$ .

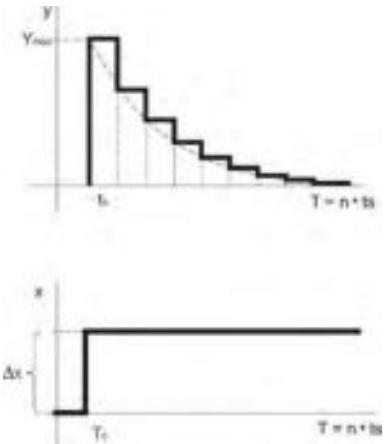
$$Y_{max} = C \cdot a \cdot \Delta x + Y_0$$

and decays to 0 according to function

$$Y_n \cdot t_s = C^n \cdot a \cdot \Delta x + Y_0 = Y_{max} \cdot C^{n-1}.$$

Thereby,  $n$  is the number of calculation cycles  $t_s$  after the input step change. Number  $n$  of required calculation cycles  $t_s$  until output variable decaying to  $y(n \cdot T_s)$  is

$$n = \frac{\lg \frac{Y(n \cdot t_s)}{Y_{max}}}{\lg C} + 1 \quad \text{Surface area } A \text{ under the decaying function is: } A = Y_{max} \cdot \left( \frac{T}{t_s} - 1 \right) = a \cdot \Delta x$$



### Ramp response:

After ramp starting, output variable  $y$  runs towards the final value of differentiation quotient

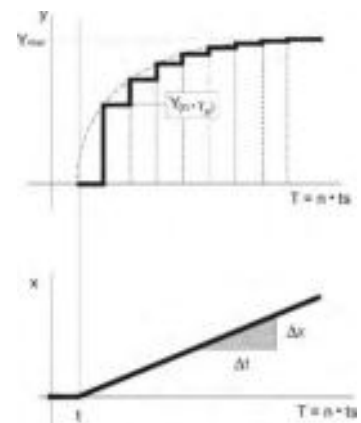
$$Y_{max} = m \cdot a \cdot T$$

according to function  $Y_{(n \cdot t_s)} = m \cdot a \cdot T \cdot (1 - C^n)$

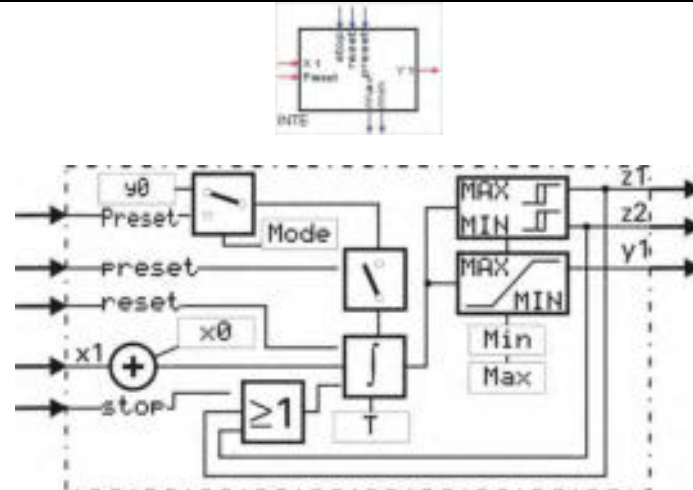
auf den endgültigen Wert des Differenzenquotienten

Thereby,  $m = \frac{dx}{dt}$  is the gradient factor of the input function. Relative error  $F$  after  $n$  calculating cycles  $T_s$  referred to the final value is calculated as follows

$F = C^n$  and the number of required calculating cycles, according to which function  $Y_{(n \cdot t_s)}$  approaches final value  $Y = Y_{max}$  to error  $F$  is  $n = \frac{\lg F}{\lg C}$ .



### 3.6.2. INTE (integrator (No. 51))



The integrator forms the integral according to equation:

$$y1_{(t)} = y1_{(t-t_s)} + \frac{t_s}{T} \cdot [x1_{(t)} + x0]$$

$t_s$	sampling interval	$x1(t)$	instantaneous $x1$
$T$	Integration constant	$y1(t)$	$y1$ after $t=n \cdot t_s$
$n$	Number of calculation cycles	$y1(t-t_s)$	previous $y1$
$x0$	Input offset		

The complex transfer function is:

$$F(p) = \frac{1}{T \cdot p}$$

Unused control inputs are interpreted as logic "0". With simultaneous input of several control commands:

**reset** = 1 has priority over **preset** and **stop**  
**preset** = 1 priority over **stop**

Integrator output **y1** is limited to the preset limits (**Min**, **Max**):  $\text{Min} \leq y1 \leq \text{Max}$ . When exceeding **Min** or **Max**, the integrator is stopped automatically and the relevant control output min or max is set to logic 1. Limit value monitoring uses a fixed hysteresis of 1 % referred to operating range (**Max** - **Min**).

#### Ein-/Ausgänge

Inputs/outputs		
<b>stop</b>	= 1	The integrator is stopped for the duration of the stop command. Output <b>y1</b> does not change.
<b>reset</b>	= 1	The integration result is adjusted to lower limit ( <b>Min</b> ). After cancelation of <b>reset</b> , integration starts at lower limiting.
<b>preset</b>	= 1	The integration result is set either to a preset value <b>y0</b> ( <b>Mode</b> =0) or to a preset variable <b>Preset</b> ( <b>Mode</b> = 1). After cancelation of the preset command, integration starts with the actually effective preset value.
Analog inputs		
<b>x1</b>		Input variable to be integrated
<b>Preset</b>		External preset value
Digital outputs		
<b>max</b>	= 1	exceeded with max. limiting
<b>min</b>	= 1	exceeded with min. limiting
Analog output		
<b>y1</b>		Integrator output

**Parameters:**

Parameter	Description	Range	Default
<b>T</b>	Time constant in s	0.1...999 999	60
<b>x0</b>	Constant	-29 999...999 999	0
<b>y0</b>	Preset value	-29 999...999 999	0
<b>Min</b>	Min. limiting	-29 999...999 999	1
<b>Max</b>	Max. limiting	-29 999...999 999	0
<b>Mode</b>	Source of preset = Para y0	0	0
	Source of preset = InpPreset	1	

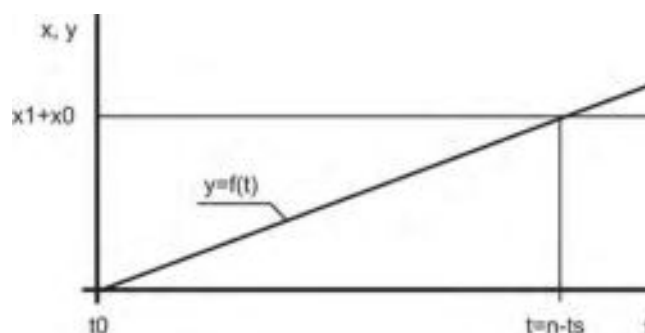
Ramp function:


With constant input  $x1+x0$ , the applicable formulas are

$$y1_{(t)} = y_{(t_0)} + n \cdot \frac{t_s}{T} \cdot (x1 + x0)$$

$$t = n \cdot t_s$$

"t" is the time required by the integrator for changing output  $y1$  linearly by value  $x1 + x0$  after integration start.

Ramp response:

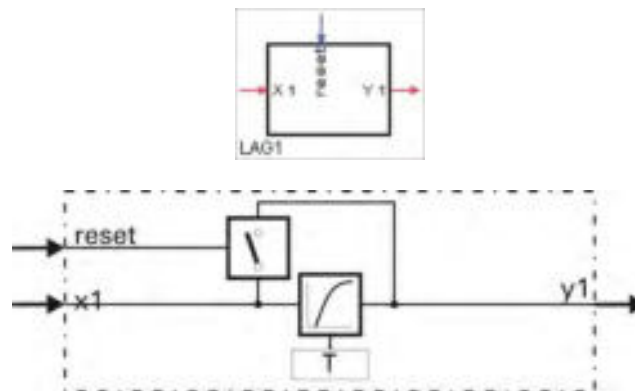
 The function has a 'memory'. This means: after power-on, it continues operating with values  $y1$ ,  $z1$  and  $z2$ , which existed at power-on, provided that the RAM data are still available.

Example: Which is the value of output variable  $y$  after  $t=20s$  with a time constant of  $100s$ , if a constant of  $x1 = 10$  Volt is preset. Sampling interval  $t_s$  is  $100ms$ .

$$n = \frac{t}{t_s} \quad n = \frac{20s}{0.1s} = 200$$

$$y = 0 + 200 \cdot \frac{0.1}{100} \cdot 10 = 2$$

This results in a gradient of  $2/20s$  oder  $0.1/1s$ .

**3.6.3. LAG 1 (filter (No. 52) )**

Dependent of control input **reset**, input variable **x1** is passed on to output **y1** with delay (reset= 0) or without delay (reset = 1). Delay is according to a 1st order e-function (1st order low pass) with time constant **T**(s). The output variable for reset= 0 is calculated according to the following equation:

$$y1_{(t)} = \frac{T}{T + t_s} \cdot y1_{(t-t_s)} + \frac{t_s}{T + t_s} \cdot x1_{(t)}$$

$t_s$  sampling interval

$T$  time constant

$n$  number of calculation cycles

$x1(t)$  instantaneous  $x1$

$x1(t-t_s)$   $y1$  after  $t = n \cdot t_s$

$y1(t-t_s)$  previous  $y1$

The complex transfer function is:

$$F(p) = \frac{1}{1 + p \cdot T}$$

**Inputs/outputs:****Digital input**

**reset** = 0 means that input signal **x1** is output without delay at output **y1**.  
 = 1 means that input signal **x1** is output at output **y1** according to the calculated e-function.

**Analog input**

**x1** Input variable to be calculated

**Analog output**

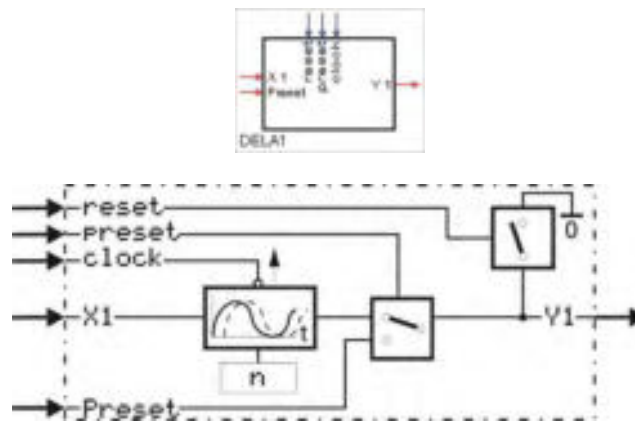
**y1** Delayed output variable

Parameter:

Parameter	Description	Range	Default
<b>T</b>	Time constant in s	0...199999	1

*No configuration parameters!*



**3.6.4. DELA1 (delay time (No. 53) )**

If the **clock** input is not wired, the function calculates  $y1(t) = x1_{(t-n \cdot t_s)}$ . ( $t_s$  = sampling interval, Delay = delay factor  $n$ )

Unless **clock** input clock is wired, the following is applicable: input variable **x1** is output with a delay by  $n$  times the amount of adjusted sampling interval  $t_s$  (phase shift by  $n \cdot t_s$ ). The effective delay time corresponds to integer multiples of the selected time group (sampling interval  $t_s$  100/200/400/800 ms).

The delay time range covers  $n = 0$  to  $255$  ( $0 \dots 255 \cdot t_s$ ).


With clock input **clock** wired, DELA1 acts like a shift register with a length of max.  $255 = \text{Parameter delay}$ . This register can be switched on by one step by an external event **preset**. Switching on is only with a positive flank (transition from  $0 \rightarrow 1$ ) at the **clock** input plus the adjusted delay factor (parameter delay)

**Example:**

With delay = 4 change-over at output  $y$  is only after 4 flank changes from  $0 \rightarrow 1$  at input clock.

**Preset:** The output provides the value applied to Preset. After  $(n+1)$  positive flanks at clock or  $(n+1)$  sampling cycles  $t_s$  (if clock isn't wired), the first input value  $x1$  appears at  $y1$

**reset:** The output provides value 0. After a positive flank at clock, value zero still is provided for the sampling interval  $t_s$ .

 The function has a 'memory'. This means: after power-on, it continues operating with values  $y1$ ,  $z1$  and  $z2$ , which existed at power-on, provided that the RAM data are still unchanged.

**Inputs/outputs****Digital inputs**

- clock** =  $0 \rightarrow 1$  clock for delaying
- preset** = 1 The preset value is taken to the output
- reset** = 1 Output  $y1$  is set to zero

**Analog inputs**

- x1** Input variable to be delayed
- Preset** Value output without delay by **preset** = 1

With several simultaneous control commands:

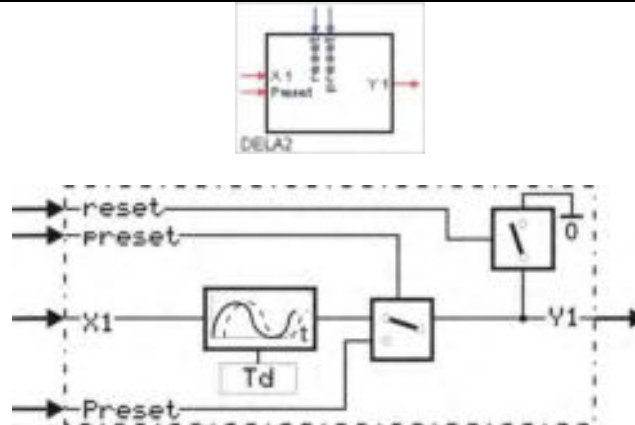
- reset** = 1 has priority over preset and stop
- preset** = 1 has priority over stop

**Analog output**

- y1** Delayed output variable

**Parameter:**

Parameter	Description	Range	Default
<b>Delay</b>	Delay factor $n$	0/1/.....255	0

**3.6.5. DELA 2 (delay time (No. 54))**

The function provides calculation  $y1(t) = x1(t-T_d)$ .

Input variable  $x1$  is output at  $y1$  with delay by time  $T_d$ . The accuracy  $T_d$  is dependent of the time group (sampling interval  $t_s$ ), to which the function is assigned.

The shift register has a maximum length of 255, which depends on the set parameter  $T_d$  and the selected sampling time  $t_s$ . The effective length is calculated from  $T_d/t_s$ .

(Rounding to the next higher natural number)

**Example:**

$T_d = 0,7s$  with assignment

- to time group 100ms means  $T_d = 0,7s$
- to time group 100ms means  $T_d = 0,8s$
- to time group 100ms means  $T_d = 0,8s$
- to time group 100ms means  $T_d = 0,8s$

The possible delay time is dependent of the configured time slot (sampling interval  $t_s$ ).

$T_d \text{ max} = 25,5s$  with  $t_s = 100ms$   
 $T_d \text{ max} = 51,0s$  with  $t_s = 200ms$   
 $T_d \text{ max} = 102,0s$  with  $t_s = 400ms$   
 $T_d \text{ max} = 204,0s$  with  $t_s = 800ms$

**Inputs/outputs****Digital input**

**preset** = 1 The preset value is taken to the output  
**reset** = 1 Output  $y1$  is set to zero

With several simultaneous control commands:

**reset** = 1 has priority over **preset** and **stop**  
**preset** = 1 priority over **stop**

**Analog output**

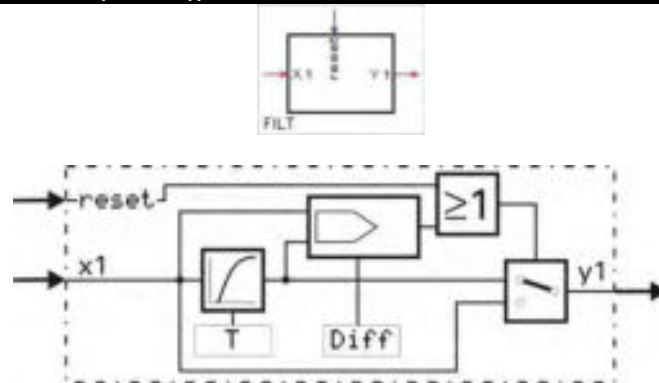
**x1** Input variable to be delayed  
**Preset** Value output with delay by  $\text{preset}=1$

**Analog output**

**y1** Delayed output variable

**Parameter:**

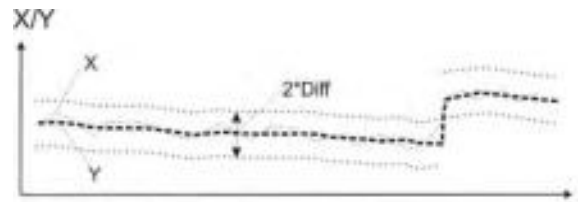
Parameter	Description	Range	Default
$T_d$	Delay in seconds	0.....204	0

**3.6.6. FILT (filter with tolerance band (No. 55))**

The complex transfer function of the filter within a tolerance band around the last output value  $|x1 - y1| \leq \delta$  is:

$$F_p = \frac{1}{1+p \cdot T}$$

With a difference higher than **Diff** or **reset** = 1 between input x1 and output y2, the filter stage is switched off and the output follows the input directly.



With a difference of input x1 and output y1 smaller than **Diff** and **reset** = 0, the output follows an e-function with time constant T. The output variable is calculated according to the following equation:

$$y1(t) = \frac{T}{T + t_s} \cdot y1(t-t_s) + \frac{t_s}{T + t_s} \cdot x1(t)$$

$t_s$  sampling interval       $x(t)$   
 $T$  time constant           $x1(t-t_s)$

**Inputs/outputs****Digital input**

<b>reset</b> = 0	$ x1 - y1  < \text{Diff}$	delay effective
	$ x1 - y1  > \text{Diff}$	delay switched off
<b>reset</b> = 1	$ x1 - y1  \leq \text{Diff}$	delay switched off
	$ x1 - y1  > \text{Diff}$	delay switched off

**Analog input**

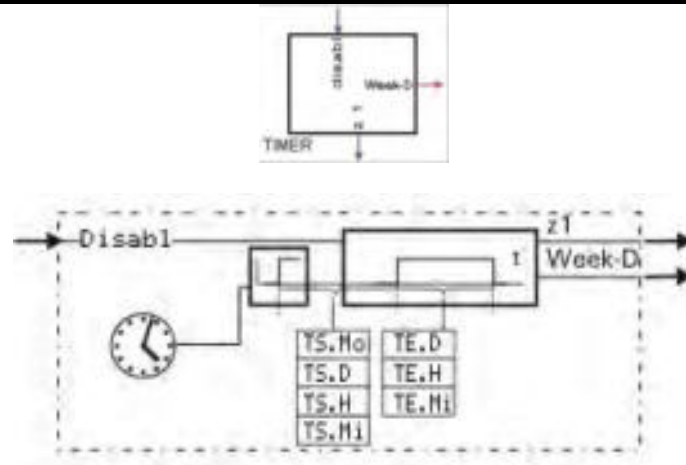
**x1**      Input variable to be delayed

**Analog output**

**y1**      Delayed output variable

**Parameters:**

Parameter	Description	Range	Default
<b>T</b>	Time constant in s	0...199999	1
<b>Diff</b>	Tolerance band $\varnothing$	0...999999	1

**3.6.7. TIMER (timer (No. 67))**

The function timer can only be used with real-time clock (9407-9xx-2xxx). Output **z1** is switched on at absolute time **TS** and switched off again after **TE**. This switching operation can be unique or cyclical (parameter adjustment). Output **Week-D** indicates the actual weekday (0...6 = Su...Sa). **TS.Mo** = 0 and **TS.D** = 0 means actual day.

When the time defined with **TS.H** and **TS.Mi** has elapsed, the 1st switching operation occurs on the following day.

With **TS.Mo** = 0 and **TS.D** < actual day, the first switching operation occurs in the following month. With **TS.Mo** ≤ actual month and **TS.D** < actual day, the 1st switching operation occurs in the next year

**Inputs/outputs****Digital input**

**disabl** = 0 output z1 active. Becomes 1 when the time was reached.  
 = 1 output z1 switched off. The output behaves like "time not yet reached"

**Digital output**

**z1** z1 is logic 1 between the start and end time.

**Analog output**

**Week-D** indicates the actual weekday (0...6 = Su...Sa)

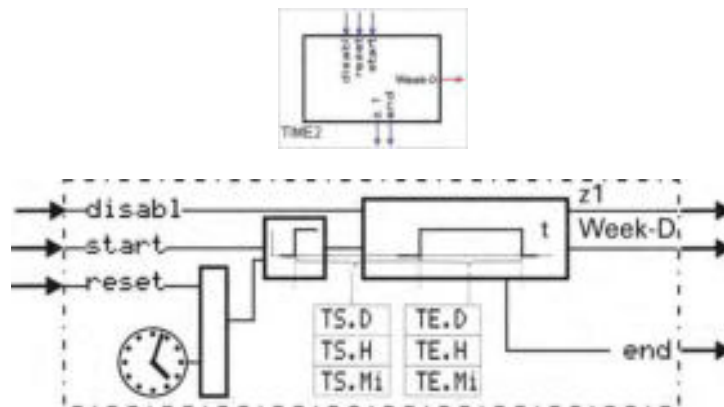
**Parameters:**

Parameter	Description	Wertebereich <sup>*)</sup>	Default
TS.Mo	Switch-on time month	0...12	0
TS.D	Switch-on time day	0...31	0
TS.H	Switch-on time hour	0...23	0
TS.Mi	Switch-on time minute	0...59	0
TE.D	Time duration days	0...255	0
TE.H	Time duration hours	0...23	0
TE.Mi	Time duration minutes	0...59	0

**Configuration:**

Configuration	Description	Value	Default
Func1	<b>cyclical</b> function runs cyclically	0	0
	<b>once</b> function runs once	1	
	<b>daily</b> function runs daily	0	
Func2	<b>Mo...Fr.</b> Function runs from Monday to Friday	1	0
	<b>Mo...Sa.</b> function runs from Monday to Saturday	2	
	<b>weekly</b> function runs weekly	3	

<sup>\*)</sup> with the engineering tool broken rational numbers can be used; however only the integral portion is taken over!

**3.6.8. TIME 2 (timer (No. 70))**

The function timer2 can only be used with real-time clock. With a positive flank at **start**, TIMER2 is started and output **z1** is switched to 1 after elapse of time **TS** and reset to 0 after elapse of time **TE**.

**Example:**

$TS.D = 2, TS.H = 1, TS.Mi = 30, TE.D = 0, TE.H = 2, TE.Mi = 2$

After the change from 0 to 1 at input **start**, output **z1** is set to 1 after 2 days, 1 hour and 30 seconds and reset to 0 after 2 hours and 2 seconds. Cyclic switching operations can be realized by feed-back of the **end** output to the **start** input.

**Inputs/outputs****Digital inputs**

- Disabl** = 1 suppresses the switching operation.  
**Reset** = 1 finishes an instantaneously running switching operation immediately.  
**Start** 0 → 1 switch-on duration start

**Digital outputs**

- z1** = 1 switching operation running.  
**end** = 1 switching operation end.

**Analog output**

- Week-D** indicates the actual weekday (0...6 ≙ Su...Sa)

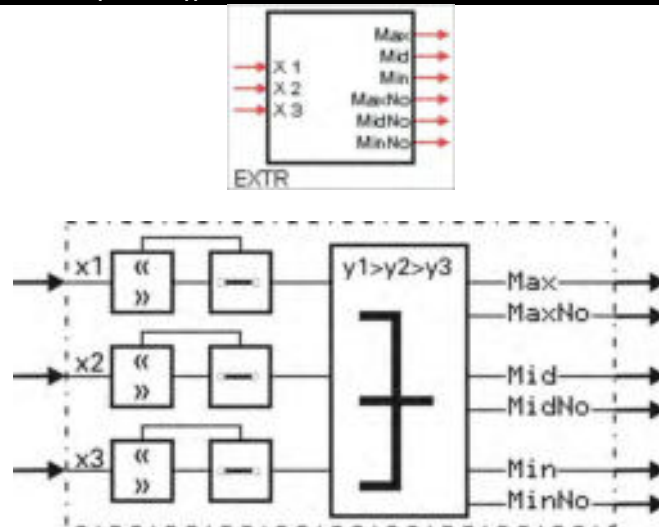
**Parameters:**

Parameter	Description	Range <sup>*1)</sup>	Default
TS.D	Switch-on delay day	0...255	0
TS.H	Switch-on delay hour	0...23	0
TS.Mi	Switch-on delay minute	0...59	0
TE.D	Switch-on duration days	0...255	0
TE.H	Switch-on duration hours	0...23	0
TE.Mi	Switch-on duration minutes	0...59	0

<sup>\*1)</sup> with the engineering tool broken rational numbers can be used; however only the integral portion is taken over!

### 3.7. Selecting and storage

#### 3.7.1. EXTR (extreme value selection (No. 30))




Analog inputs **x1**, **x2** and **x3** are sorted according to their instantaneous values and provided at outputs **Max**, **Mid** and **Min**. Input value output is at **Max** for the highest one, at **Mid** for the medium one and at **Min** for the smallest one.

The number of the input with the highest value is output at **MaxNo**.

The number of the input with the medium value is provided at output **MidNo**.

The number of the input with the smallest value is provided at output **MinNo**.

 With equality, the distribution is at random. Inputs are not included into the extreme value selection, if: the input is not wired or the input value is higher than  $1,5 \cdot 10^{37}$  or smaller than  $-1,5 \cdot 10^{37}$

Number of failed inputs	Max	Mid	Min	MaxNo	MidNo	MinNo
0	xmax	xmid	xmin	number of xmax	number of xmid	number of xmin
1	xmax		xmin	number of xmax		number of xmin
2	the valid value			number of the valid value		
3	$1,5 \cdot 10^{37}$	$1,5 \cdot 10^{37}$	$1,5 \cdot 10^{37}$	0	0	0

#### Inputs/outputs

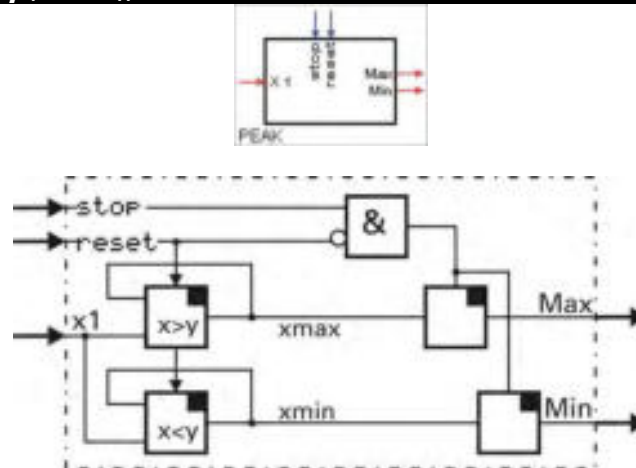
##### Analog inputs

**x1 . . . x3** Input variables to be compared

##### Analog outputs

<b>Max</b>	Maximum instantaneous input value
<b>Mid</b>	Mean instantaneous input value
<b>Min</b>	Minimum instantaneous input value
<b>MaxNo</b>	Number of maximum instantaneous input value (1 = <b>x1</b> , 2= <b>x2</b> , 3= <b>x3</b> )
<b>MidNo</b>	Number of mean instantaneous input value (1 = <b>x1</b> , 2= <b>x2</b> , 3= <b>x3</b> )
<b>MinNo</b>	Number of minimum instantaneous input value (1 = <b>x1</b> , 2= <b>x2</b> , 3= <b>x3</b> )


### 3.7.2. PEAK (peak value memory (No. 31))



Maximum input value  $x_{\max}$  and minimum input value  $x_{\min}$  are determined, stored and output at **Max** and **Min**. With the stop input set to 1, the extreme values determined last remain unchanged.

If the **reset** input is set to 1, the extreme value memory and any applied **stop** command are cancelled. ( $x_{\max}$  and  $x_{\min}$  are set to the instantaneous  $x1$  value and follow input  $x1$ , until the **reset** input returns to 0.).

Unused inputs are interpreted as 0 or logic 0.

 The function has a 'memory'. This means: after power-on, it continues operating with the Min- and Max values which existed at power-off, provided that the RAM data are still unchanged.

*No parameters!*

#### Inputs/outputs

##### Digital inputs

<b>stop</b>	With the stop input set to 1, instantaneous values Max and Min are unchanged.
<b>reset</b>	The reset input deletes the <b>Min</b> and <b>Max</b> values.

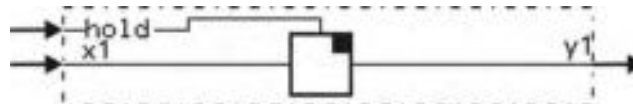
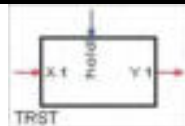
##### Analog inputs

<b>x1</b>	Process value, the min and max values of which are output.
-----------	--


##### Analog outputs

<b>Max</b>	Maximum value
<b>Min</b>	Minimum value

### 3.7.3. TRST (hold amplifier (No. 32) )



With control input hold set to 1, instantaneous input value x1 is stored and output at y1. With control input hold set to 0, output y1 follows input value x1.

 The function has a 'memory'. This means: after power-on, it continues operating with the y1 value which existed at power-off, provided that the RAM data are still unchanged.

*No parameters!*

#### Inputs/outputs

##### Digital input

<b>hold</b>	Storage signal for the x1 value
-------------	---------------------------------

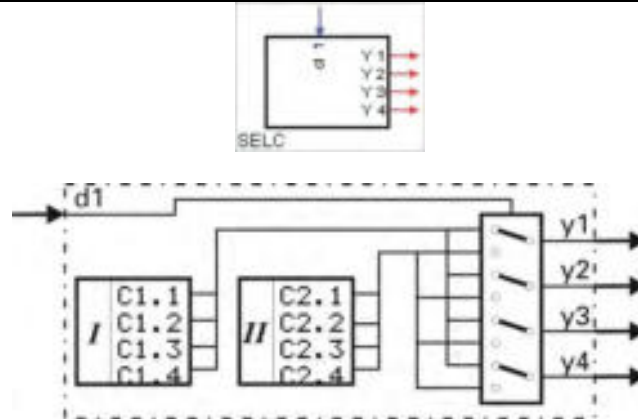
##### Analog input

<b>x1</b>	Process value which can be output stored
-----------	--

##### Analog output

<b>y1</b>	Function output
-----------	-----------------



**3.7.4. SELC (Constant selection (No. 33))**

Dependent of control signal d1, the four preset parameters of group 1 or of group 2 are output.

**Inputs/outputs****Digital input**

**d1** Selecting the constant group (0 = group 1; 1= group 2)

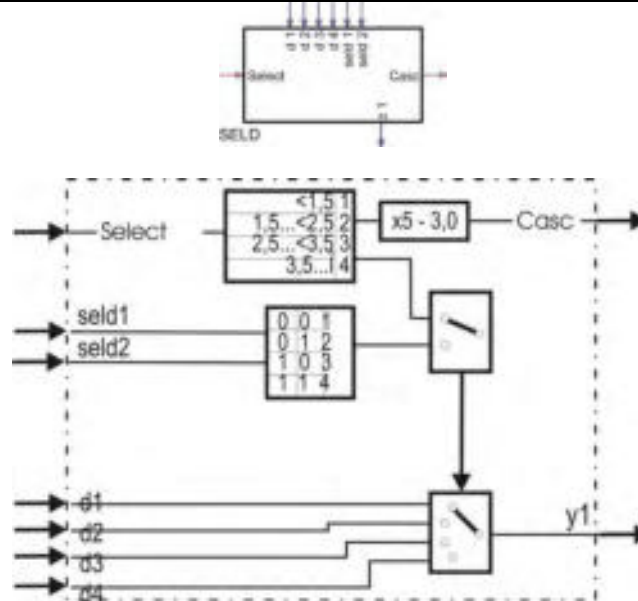
**Analog outputs**

	d1=0 $\triangleq$ group I	d1=1 $\triangleq$ group II
y1	C1 . 1	C2 . 1
y2	C1 . 2	C2 . 2
y3	C1 . 3	C2 . 3
y4	C1 . 4	C2 . 4

**Parameters:**

Parameter	Description	Range	Default
C1 . 1	1. Constant of group 1, output at y1 with d1 =0.	-29 999...999 999	0
C1 . 2	2. Constant of group 1, output at y2 with d1 =0.	-29 999...999 999	0
C1 . 3	3. Constant of group 1, output at y3 with d1 =0.	-29 999...999 999	0
C1 . 4	4. Constant of group 1, output at y4 with d1 =0.	-29 999...999 999	0
C2 . 1	1. Constant of group 2, output at y1 with d1 =1.	-29 999...999 999	1
C2 . 2	1. Constant of group 2, output at y2 with d1 =1.	-29 999...999 999	1
C2 . 3	3. Constant of group 2, output at y3 with d1 =1.	-29 999...999 999	1
C2 . 4	4. Constant of group 2, output at y4 with d1 =1.	-29 999...999 999	1

### 3.7.5. SELD (selection of digital variables - function no. 06))



Selection of one of the 4 digital inputs is either by an analog “Select” signal or by the two digital control signals seld1, seld2. If analog control signal Select is connected, selecting is done using this control signal. If the input is not connected, selection is by means of the two digital control inputs seld1, seld2.

This function block is cascadable. The select input can be linked with the cas output of another SELD block into a choice of 8 digital variables.

#### Inputs/outputs

##### Digital inputs

- d1** Input, is output at z1, when seld1=0 and seld2=0
- d2** Input, is output at z1, when seld1=0 and seld2=1
- d3** Input, is output at z1, when seld1=1 and seld2=0
- d4** Input, is output at z1, when seld1=1 and seld2=1
- seld1** 1st control signal for variable selection (least significant bit)
- seld2** 2nd control signal for variable selection (most significant bit)

##### Analog inputs

- Select** Dependent on input value, the relevant variable is output at the z1 output C1.4

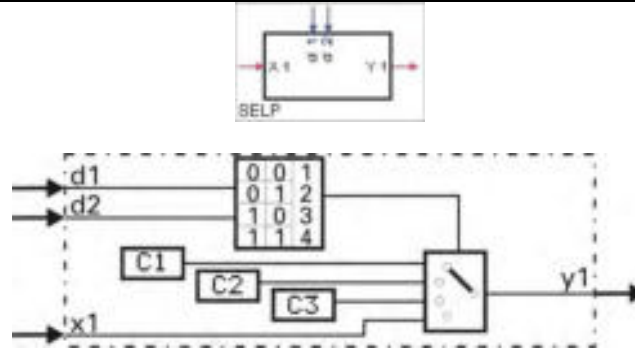
##### Digital outputs

- z1** d1, d2, d3. or d4  
According to the input value of Select (or to values seld1, seld2) the relevant input variable is output.

##### Analog outputs

- Casc** Cascade output = Select – 3.0

### 3.7.6. SELP (parameter selection (No. 34) )



Dependent of control signals d1 and d2, either one of the three preset parameters C1, C2, C3 or input variable x1 is connected with output y1. Unused inputs are interpreted as 0 or logic 0.

#### Inputs/outputs

##### Digital inputs

- d1 1st digital input for parameter selection
- d2 2nd digital input for parameter selection

##### Analog input

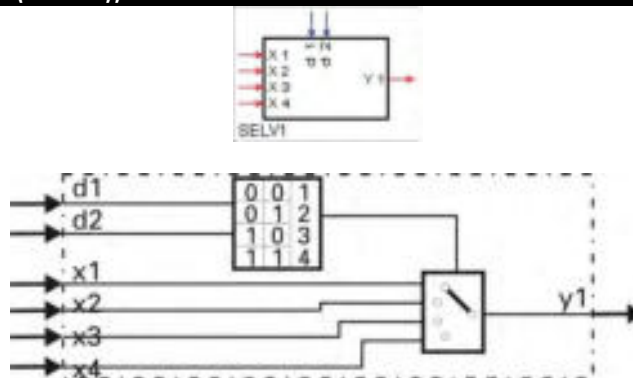
- x1 Input is output at y1, when d1 = 1 and d2 = 1

##### Analog outputs

	d1	d2
y1=C1	0	0
y1= C2	0	1
y1= C3	1	0
y1= x1	1	1

#### Parameters:

Parameter	Description	Range	Default
C1	1st constant, output at y1 with d1 = 0 and d2 = 0.	-29 999...999 999	0
C2	2nd constant, output at y1 with d1 = 0 and d2 = 1.	-29 999...999 999	0
C3	3rd constant, output at y1 with d1 = 1 and d2 = 0.	-29 999...999 999	0

**3.7.7. SELV1 (variable selection (No. 35))**

Dependent of control signals d1 and d2, one of four inputs x1...x4 is connected with output y1. Unused inputs are interpreted as 0 or logic 0.

**Inputs/outputs****Digital inputs**

- d1      1st digital input for parameter selection  
d2      2nd digital input for parameter selection

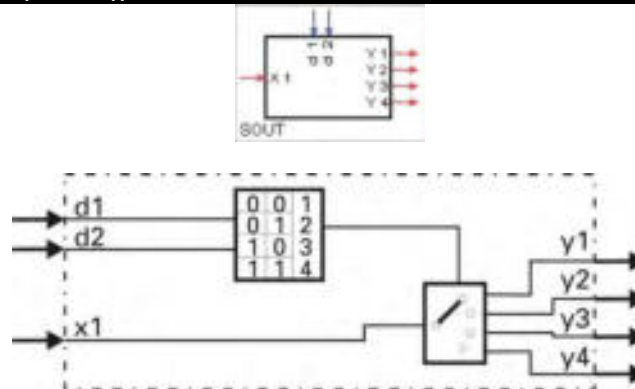
**Analog inputs**

- x1      Input is output at y1, when d1 = 0 and d2 = 0  
x2      Input is output at y1, when d1 = 0 and d2 = 1  
x3      Input is output at y1, when d1 = 1 and d2 = 0  
x4      Input is output at y1, when d1 = 1 and d2 = 1

**Analog outputs**

	d1	d2
y1= x1	0	0
y1= x2	0	1
y1= x3	1	0
y1= x4	1	1

No parameters:

**3.7.8. SOUT (Selection of output (No. 36))**

Dependent of control signals d1 and d2,, input variable x1 is connected to one of outputs y1, y2, y3 or y4. Unused inputs are interpreted as 0 or logic 0.

**Inputs/outputs****Digital inputs**

- d1 1st digital input for output selection  
d2 2nd digital input for output selection

**Analog input**

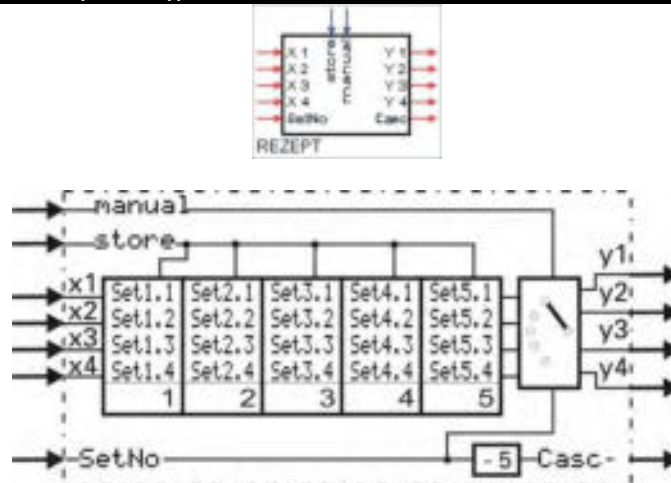
- x1 Input is output at y1, when d1 = 0 and d2 = 0

**Analoge Ausgänge**

	d1	d2
y1= x1	0	0
Y2= x1	0	1
Y3= x1	1	0
Y4= x1	1	1

No parameters:

### 3.7.9. REZEPT (recipe management (No. 37))



The function has 5 groups (recipe blocks) each with 4 memory locations. The recipes can be written via parameter setting and analog inputs. The function parameters are stored in EEPROM with back-up.

Selection which recipe block is output at y1...y4 is determined by the value applied to input **SetNo**.

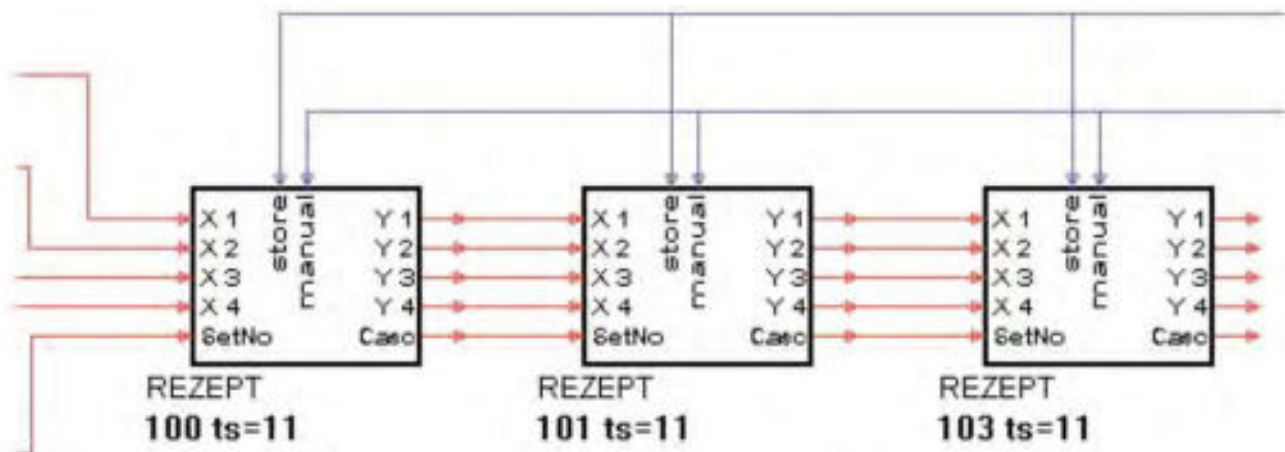
In mode STORE (store = 1), the values applied to x1... x4 are written into the memory addresses of the recipe block selected with input **SetNo**.

During manual mode (**manual** = 1), the inputs are directly connected with the outputs.

If more than 5 recipes are required, a corresponding number of recipe functions are simply cascaded.

**!** Values of the used analog inputs are stored as parameter values when detecting a positive edge at the store- input. This input should be activated only with relevant changes of the input values. Too frequent storing operations may lead to the destruction of the EEPROM! ( → page 311)

Example for 15 recipes



With cascading, the values for the overall recipe are available at outputs y1...y4 of the last stage.

### Inputs/outputs

#### Digital inputs

<b>store</b>	This input reacts only on a positive flank, i.e. on a change from 0 to 1. With this flank, input values x1...x4 are stored in the recipe block selected with <b>SetNo</b> . The values are stored in RAM and in EEPROM.
<b>dynamic</b>	With <b>store</b> = 0 or permanent 1, storage is omitted. The saving process is also carried out in manual mode (manual = 1).
<b>manual</b>	manual = 0: automatic mode: recipe function active manual = 1: manual mode: the values of inputs x1...x4 are applied to y1...y4 directly.

#### Analog inputs

<b>x1 . . . x4</b>	In mode STORE ( <b>store</b> = 1), the values applied to x1... x4 are written into the memory locations of the group selected with <b>SetNo</b> . The inputs are connected with the outputs directly in manual mode (manual = 1) and also when the <b>SetNo</b> input is beyond range 1...5. Selecting a recipe block:
<b>SetNo5</b>	The value of <b>SetNo</b> determines, which one of the 5 recipe blocks is selected. Selection is valid for reading and storage (→ <b>store</b> ). A recipe block is selected only with a value within 1...5 at <b>SetNo</b> . With <b>SetNo</b> out of range 1...5, the inputs are connected directly with the outputs (independent of the status at the A/M input manual). This is required for cascading.

#### Analog outputs

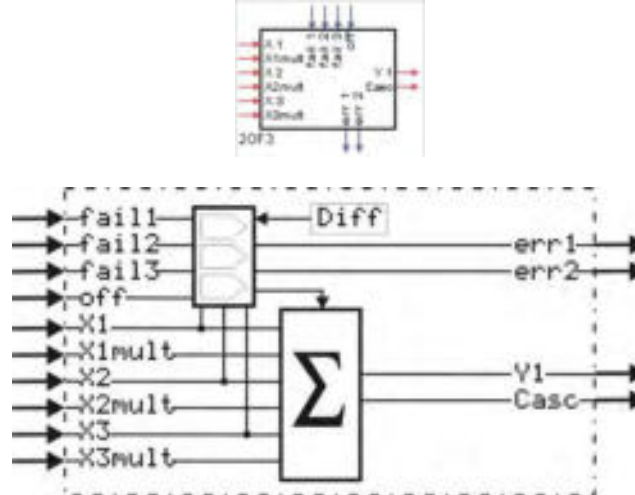
<b>y1 . . . y4</b>	The values at y(i) correspond either to the recipe block selected with <b>SetNo</b> or to inputs x(i) in manual mode ( <b>store</b> = 1).
<b>Casc</b>	The value at output <b>Casc</b> is the value of input <b>SetNo</b> reduced by 5 and is used for cascading.

### Parameters:

Via interface, 20 parameters (5 recipe blocks each with 4 values) can be preset:

Parameter	Description		Range	Default
Set1.1	Recipe block 1	Parameter 1 for recipe 1	-29 999...999 999	0
Set1.2		Parameter 2 for recipe 1	-29 999...999 999	0
Set1.3		Parameter 3 for recipe 1	-29 999...999 999	0
Set1.4		Parameter 4 for recipe 1	-29 999...999 999	0
Set2.1	Recipe block 2	Parameter 1 for recipe 2	-29 999...999 999	0
Set2.2		Parameter 2 for recipe 2	-29 999...999 999	0
Set2.3		Parameter 3 for recipe 2	-29 999...999 999	0
Set2.4		Parameter 4 for recipe 2	-29 999...999 999	0
Set3.1	Recipe block 3	Parameter 1 for recipe 3	-29 999...999 999	0
Set3.2		Parameter 2 for recipe 3	-29 999...999 999	0
Set3.3		Parameter 3 for recipe 3	-29 999...999 999	0
Set3.4		Parameter 4 for recipe 3	-29 999...999 999	0
Set4.1	Recipe block 4	Parameter 1 for recipe 4	-29 999...999 999	0
Set4.2		Parameter 2 for recipe 4	-29 999...999 999	0
Set4.3		Parameter 3 for recipe 4	-29 999...999 999	0
Set4.4		Parameter 4 for recipe 4	-29 999...999 999	0

### 3.7.10. 2OF3 ( 2-out-of-3 selection with mean value formation (No. 38))



Function 2OF3 forms the arithmetic mean value of input variables **x1**, **x2** and **x3**.

The difference of x1, x2 and x3 is formed and compared with parameter Diff. Inputs the value of which exceeds this limit value are not used for mean value formation. With 1 applied to fail1...fail3 (e.g. the fail signals of AINP), faulty inputs are not taken into account either for mean value formation. err1 = 1 indicates that 1 input failed and was not used for mean value formation. If at least 2 inputs do not participate in mean value formation, output err2 is set to 1. With input off set to 1 or if output err2 = 1 the x1 value is output at y1.

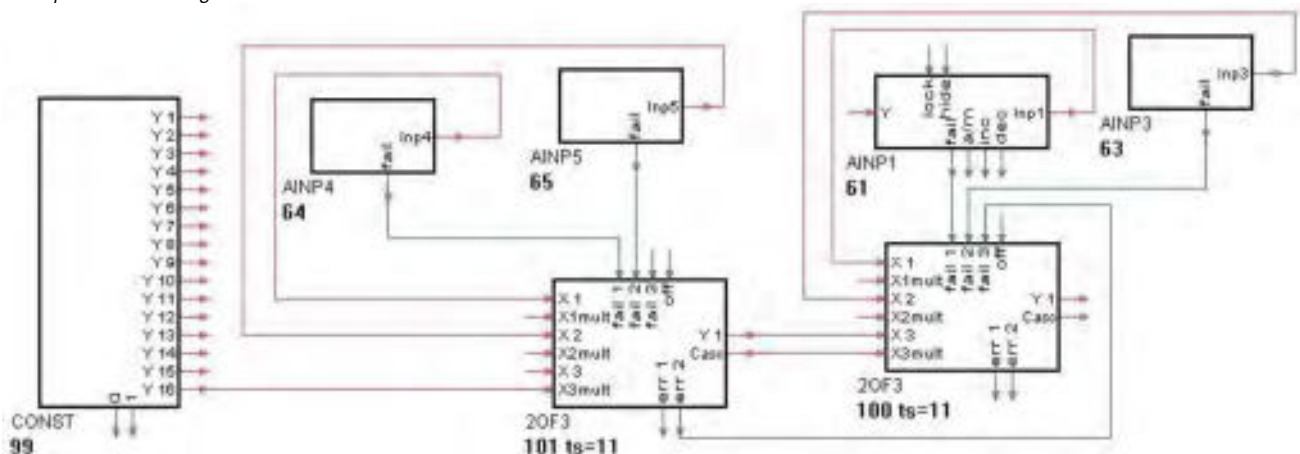
With more than 3 input variables, function 2OF3 can be cascaded.

Output Casc indicates the number of values used for mean value formation. This is important with 2OF3 function cascading.

With unwired factor inputs (x1mult...x3mult) factor 1 is used automatically. If one of inputs x1...x3 is not used, the relevant x-mult must be set to 0.

The x-mult input of the following function block is wired with factor output Casc of the previous function block.

*Example of cascading*



In this example, CONST output y16 = 0 is set.

The following formulas are calculated:

The left 2OF3:  $y1 = \frac{x1 \cdot 1 + x2 \cdot 1 + x3 \cdot 1}{2}$  and the right 2OF3:  $y1 = \frac{x1 \cdot 1 + x2 \cdot 1 + x3 \cdot 2}{4}$



### Inputs/outputs

#### Digital inputs

<b>fail1</b>	Error message for input <b>x1</b> . With <b>fail1</b> = 1, input <b>x1</b> is not taken into account for mean value formation.
<b>fail2</b>	Error message for input <b>x2</b> . With <b>fail2</b> = 1, input <b>x2</b> is not taken into account for mean value formation.
<b>fail3</b>	Error message for input <b>x3</b> . With <b>fail3</b> = 1, input <b>x3</b> is not taken into account for mean value formation.
<b>off</b>	Function switch-off: with <b>off</b> = 1, input <b>x1</b> is output at <b>y1</b> .

#### Analog inputs

<b>x1</b>	Measurement input 1 Factor input, pertaining to measurement input 1. Determination is made of how many measurement inputs the <b>x1</b> consists (required with function block cascading or input <b>x1</b> not connected). Non-connected input <b>x1mult</b> is evaluated as value 1.
<b>x2</b>	Measuring input 2 Factor input, pertaining to measurement input 1. Determination is made of how many measurement inputs the <b>x2</b> consists (required with function block cascading or input <b>x2</b> not connected). Non-connected input <b>x2mult</b> is evaluated as value 1.
<b>x3</b>	Measuring input 3 Factor input, pertaining to measurement input 1. Determination is made of how many measurement inputs the <b>x3</b> consists (required with function block cascading or input <b>x3</b> not connected). Non-connected input <b>x3mult</b> is evaluated as value 1.

#### Digital outputs

<b>err1</b>	Error message: <b>err1</b> = 1 indicates that at least one of inputs <b>x1</b> ... <b>x3</b> is not taken into account with mean value formation.
<b>err2</b>	Error message: <b>err2</b> = 1 indicates that mean value formation is omitted. Either several inputs ( <b>fail</b> or difference > <b>Diff</b> ) are disturbed or function was switched off by input <b>off</b> .

#### Analog outputs

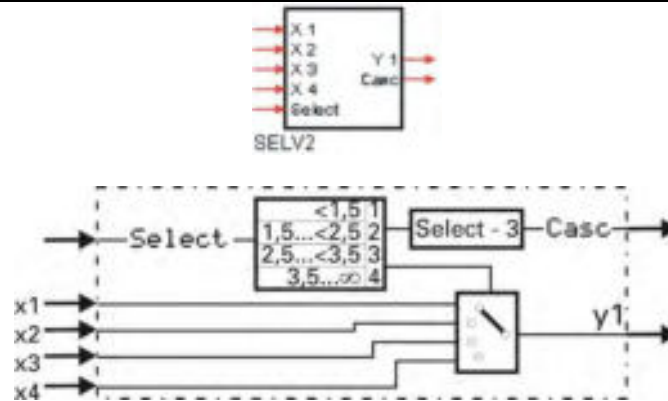
<b>y1</b>	arithmetic mean value or switching to <b>x1</b> occurred ( <b>off</b> = 1 or several inputs defective).
<b>Casc</b>	Factor: number of the values used for mean value formation. <b>Casc</b> = <b>x1mult</b> + <b>x2mult</b> + <b>x3mult</b> .

### Parameter:

Parameter	Description	Range	Default
<b>Diff</b>	Limit value for comparison of differences between inputs <b>x1</b> ... <b>x3</b> for determination of faulty inputs	0...999 999	1

No configuration parameters:

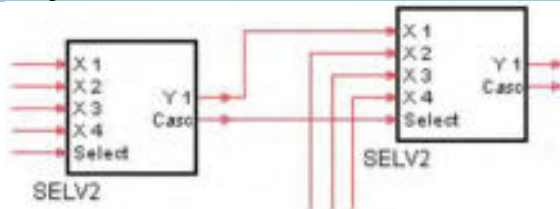
### 3.7.11. SELV2 (cascadable selection of variables (No. 39))



Dependent of input **Select**, one of the four inputs  $x1...x4$  is connected with output  $y1$ . Unused inputs are interpreted as 0. Output **Casc** = input **Select** -3.

The function can be cascaded as shown in the example given below. Dependent of input signal **Select** at the 1st SELV1, the corresponding variable is output at **Y1** of the 2nd SELV2.

#### Cascading



2. SELV2	y1 1output 2nd SELV2
$\text{Select} < 1,5$	$x1$ vom 1. SELV2
$1,5 \leq \text{Select} < 2,5$	$x2$ vom 1. SELV2
$2,5 \leq \text{Select} < 3,5$	$x3$ vom 1. SELV2
$3,5 \leq \text{Select} < 4,5$	$x4$ vom 1. SELV2
$4,5 \leq \text{Select} < 5,5$	$x2$ vom 2. SELV2
$5,5 \leq \text{Select} < 6,5$	$x3$ vom 2. SELV2
$\text{Select} \geq 6,5$	$x4$ vom 2. SELV2

#### Inputs/outputs

##### Analog inputs

$x1$	Input is output at $y1$ with $\text{Select} < 1,5$ .
$x2$	Input is output at $y1$ with $1,5 < \text{Select} < 2,5$ .
$x3$	Input is output at $y1$ with $2,5 < \text{Select} < 3,5$ .
$x4$	Input is output at $y1$ with $\text{Select} < 3,5$ .
<b>Select</b>	Dependent of input value, the relevant variable is output at $y1$ .

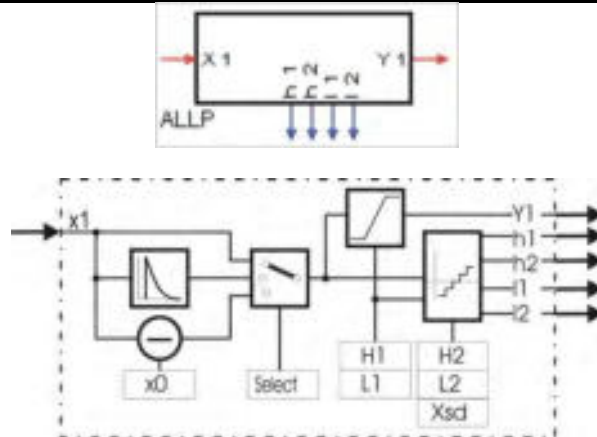
##### Analog outputs

$y1$	According to the input value of <b>Select</b> , the relevant input variable is output.
<b>Casc</b>	Cascade output = <b>Select</b> - 3

No parameters:

### 3.8. Limit value signalling and limiting

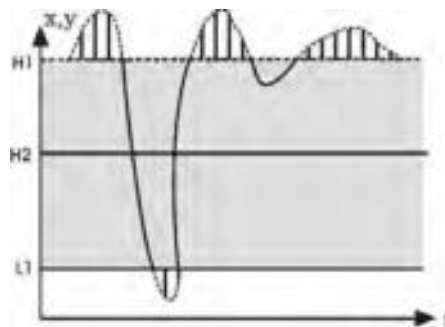
#### 3.8.1. ALLP (alarm and limiting with fixed limits(No. 40))



##### Signal limiting:

Parameter L1 determines the minimum, H1 the maximum limiting.  $y_1$  is limited to the range between L1 and H1. ( $L1 \leq y1 \leq H1$ ). With parameter H1 smaller than L1, a higher priority is allocated with H1. This means that  $y1 \leq H1$

*Limiting at  $H1 < L1$*



##### Limit signaller

The limit signaller has two 2 low and high alarms (L1, L2, H1 and H2). Configuration parameter **Select** can be used to select the variable to be monitored ( $x1$ ,  $dx1/dt$ ,  $x1 - x0$ ). The limit values are freely adjustable as parameters and have an adjustable hysteresis of  $\geq 0$ .

The smallest separation between a minimum and a maximum limit value is 0. When an alarm is triggered, the corresponding output (L1, L2, H1 and H2) is logic "1"

##### D -alarm ( $dx1/dt$ )

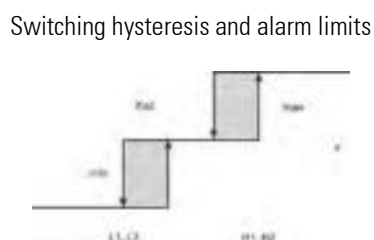
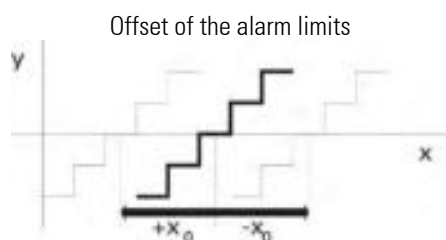
Value  $x1(t-1)$  measured one sampling interval before is subtracted from instantaneous value  $x1(t)$ . This difference is divided by calculation cycle time  $Tr$  (100, 200, 400, 800ms).

Thus input variable  $x1$  can be monitored for its rate of change

## Limit value signalling and limiting

### Alarm with offset ( $x_1 - x_0$ ):

$x_1$  can be shifted by means of  $x_0$ . This corresponds to the offset of the adjusted alarm limits (L1, L2, H1 and H2) in parallel to the x-axis



## Inputs/outputs

### Analog input

$x_1$  Input value to be monitored

### Digital outputs

L1 Low alarm 1 - becomes logic 1, if  $x_1 < L1$   
 L2 Low alarm 2 - becomes logic 1, if  $x_1 < L2$   
 H1 High alarm 1 - becomes logic 1, if  $x_1 > H1$   
 H2 High alarm 2 - becomes logic 1, if  $x_1 > H2$

### Analog output

$y_1$  Calculated and limited input signal  $x_1$ .

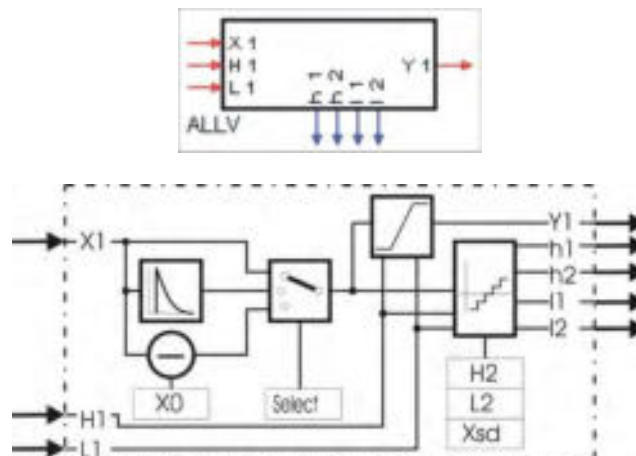
## Configuration parameter:

Parameter	Description	Range	Default
Select	Selection of the variable to be monitored	$x_1$	$x_1$
		D -alarm	$\frac{dx_1}{dt}$
		Alarm with Offset	$x_1 - x_0$

## Parameters:

Parameter	Description	Range	Default
H1	High-alarm 1	-29 999...999 999	9999
H2	High-alarm 2	-29 999...999 999	9999
L1	Low-alarm 1	-29 999...999 999	9999
L2	Low-alarm 2	-29 999...999 999	-9999
$x_0$	Offset $x_0$	-29 999...999 999	0
Xsd	Switching hysteresis	0...999 999	1

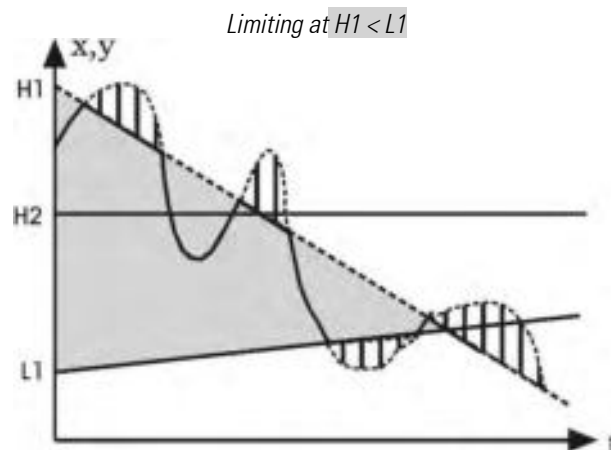
### 3.8.2. ALLV (alarm and limiting with variable limits (No. 41))



#### Signal limiting:

Analog input H1 determines the maximum limiting, L1 determines the minimum limiting. y1 is limited to the range between L1 and H1 ( $L1 \leq y1 \leq H1$ ).

As both H1 and L1 come from analog inputs, H1 can be smaller than L1. In this case, H1 is assigned a higher priority. This means that signal y1 is  $\leq H1$ !



#### Limit signaller:

The limit signaller has 2 low and high alarms (L1, L2, H1 and H2). The variable to be monitored can be selected with configuration parameter Select ( $x1$ ,  $dx1/dt$ ,  $x1 - x0$ ).

The limit values are freely adjustable via the analog inputs H1 and L1 and have an adjustable hysteresis of  $\geq 0$ . The smallest separation between a minimum and a maximum limit value is 0. With an alarm triggered, the relevant output (L1, L2, H1 and H2) is logic "1".

#### D -alarm ( $dx1/dt$ )

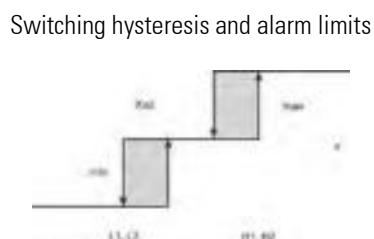
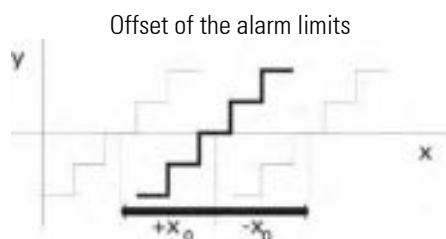
Value  $x1(t-1)$  measured one sampling interval before is subtracted from instantaneous value  $x1(t)$ . This difference is divided by calculation cycle time  $T_r$  (100, 200, 400, 800ms).

Thus input variable  $x1$  can be monitored for rate of change.

## Limit value signalling and limiting

### Alarm with offset ( $x1 - x0$ ):

$x1$  can be shifted by means of  $x0$ . This corresponds to the offset of alarm limits (L1, L2, H1 and H2) in parallel to the x-axis.



## Inputs/outputs

### Analog inputs

$x1$	Input value to be monitored
H1	High-alarm
L1	Lo-alarm

### Digital outputs

L1	Low - alarm 1 - is logic 1 with $x1 < L1$
L2	Low - alarm 2 - is logic 1 with $x1 < L2$
H1	High - alarm 1 - is logic 1 with $x1 > H1$
H2	High - alarm 2 - is logic 1 with $x1 > H2$

### Analog output

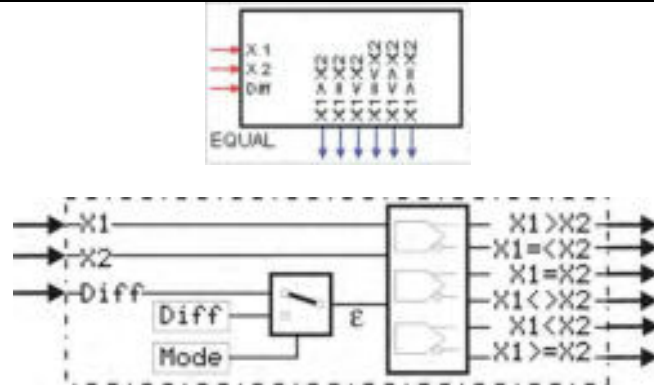
$y1$	Calculated and limited input signal $x1$ .
------	--

## Configuration parameters:

Parameter	Description	Range	Default
Select	Selection of variable to be monitored	$x1$ D -alarm alarm with Offset	$x1t$ $dx1/dt$ $x1 - x0$

## Parameters:

Parameter	Description	Range	Default
H2	High-alarm 2	-29 999...999 999	9999
L2	Low-alarm 2	-29 999...999 999	-9999
$x0$	Offset $x0$	-29 999...999 999	0
Xsd	Switching hysteresis	0...999 999	1

**3.8.3. EQUAL (comparison (No. 42))**

The function checks the two analog input values  $x1$  and  $x2$  for equality.

The values are equal, if the amount of their difference is smaller than or equal to the preset tolerance.

Comparison conditions	z1	z2	z3	z4	z5	z6
$x2 + \text{Diff} < x1$	1	0	0	0	1	1
$x2 - \text{Diff} \leq x1 \leq x2 + \text{Diff}$	0	1	0	1	0	1
$x2 - \text{Diff} > x1$	0	0	1	1	1	0

The tolerance can be adjusted either as parameter **Diff** (**Mode** = **Para.Diff**) or entered at analog input **Diff** (**Mode** = **Inp.Diff**).

**Inputs/outputs****Analog inputs**

<b>x1</b>	1st input value to be compared
<b>x2</b>	2nd input value
<b>Diff</b>	Tolerance for comparison operations

**Digital outputs**

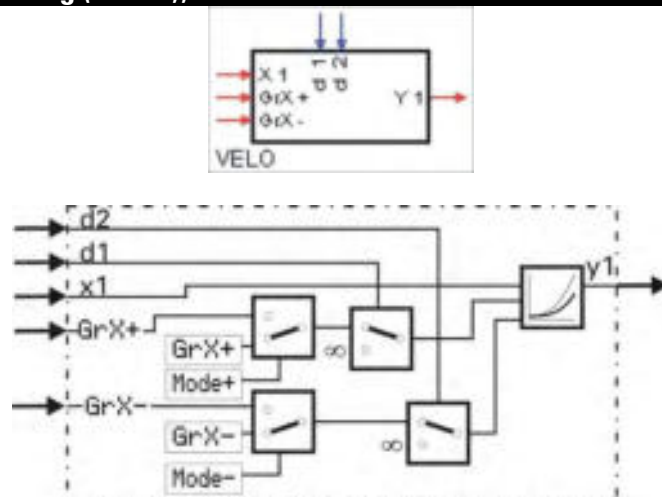
Diff = 0

<b>z1</b>	$z1 = 1$ , with $x2 + \text{Diff} < x1$	$x1 > x2$
<b>z2</b>	$z2 = 1$ , with $x2 - \text{Diff} \leq x1 \leq x2 + \text{Diff}$	$x1 = x2$
<b>z3</b>	$z3 = 1$ , with $x2 - \text{Diff} > x1$	$x1 < x2$
<b>z4</b>	$z4 = 1$ , with $x2 + \text{Diff} \geq x1$	$x1 \leq x2$
<b>z5</b>	$z5 = 1$ , with $x2 - \text{Diff} > x1 > x2 + \text{Diff}$	$x1 <> x2$
<b>z6</b>	$z6 = 1$ , with $x2 - \text{Diff} \leq x1$	$x1 \geq x2$

No configuration parameters!

**Parameters:**

Parameter	Description	Range	Default
<b>Mode</b>	Tolerance source	Parameter <b>Diff</b> analog input <b>Diff</b>	←
<b>Diff</b>	Tolerance for comparison operation	0... 999 999	0

**3.8.4. VELO (rate-of-change limiting (No. 43))**

The function passes input variable  $x_1$  to output  $y_1$  and limits its rate of change  $dx_1/dt$  to a positive and negative gradient.

The gradients can be adjusted either as parameter  $GrX+$  and  $GrX-$  or preset at analog inputs  $GrX+$  and  $GrX-$ .

Switch-over between the gradient sources is by parameter **Mode+** for the positive gradient and by **Mode-** for the negative gradient.

Via digital inputs  $d_1$  and  $d_2$ , limiting can be switched off separately for positive and negative rates of change. When using the analog inputs for gradient adjustment, the following is applicable:  $GrX+ \geq 0$  or  $GrX- \leq 0$  otherwise the relevant gradient is set to 0.

The function has a 'memory'. This means: after power-on, it continues operating with the value of  $y_1$  which existed at power-off, provided that the RAM data are still unchanged.

**Inputs/outputs****Digital inputs**

$d_1$	Control of positive gradient (0 = the selected gradient is effective. 1 = gradient = $\infty$ )
$d_2$	Control of negative gradient (0 = the selected gradient is effective. 1 = gradient = $\infty$ )

**Analog inputs**

$x_1$	Input variable to be limited
$GrX+$	positive gradient [1/s] with parameter <b>Mode+</b> = Inp. $GrX+$
$GrX-$	negative gradient [1/s] with parameter <b>Mode-</b> = Inp. $GrX-$

**Analog output**

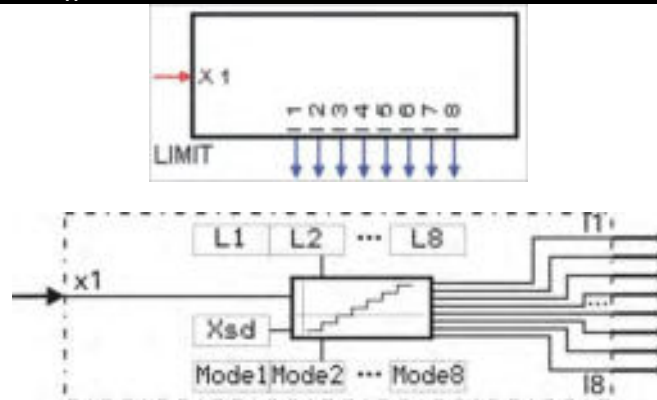
$y_1$	Limited input value $x_1$
-------	---------------------------

*No configuration parameters!*

**Parameters:**

Parameter	Description	Range	Default
<b>Mode+</b>	Source of positive gradient	Parameter $GrX+$ analog input $GrX+$	←
<b>Mode-</b>	Source of negative gradient	Parameter $GrX-$ analog input $GrX-$	←
$Grx+$	positive gradient [1/s] with parameter <b>Mode+</b> = Para. $GrX+$	0...999 999	0
$Grx-$	negative gradient [1/s] with parameter <b>Mode-</b> = Para. $GrX-$	-29 999...0	0

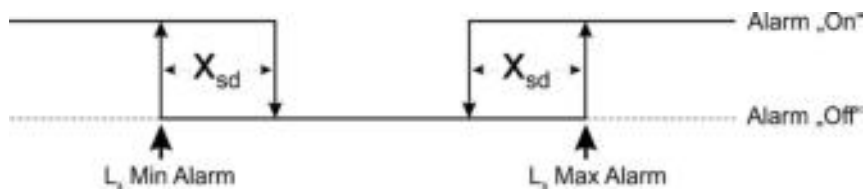


**3.8.5. LIMIT (multiple alarm (No. 44))**

The function checks input variable  $x1$  for 8 alarm values  $L1 \dots L8$ . Dependent of configuration by **Mode 1 ... Mode 8**, the relevant alarm value is evaluated as MAX or MIN alarm.

With MAX alarm configuration, the alarm is triggered when the input signal is higher than the alarm value and finished when it is lower than (alarm value - hysteresis  $X_{sd}$ ).

With MIN alarm configuration, the alarm is triggered when the input signal is lower than the alarm value and finished when it is higher than (alarm value + hysteresis  $X_{sd}$ ).

**Inputs/outputs****Analog input**

$x1$  Input variable to be monitored

**Digital outputs**

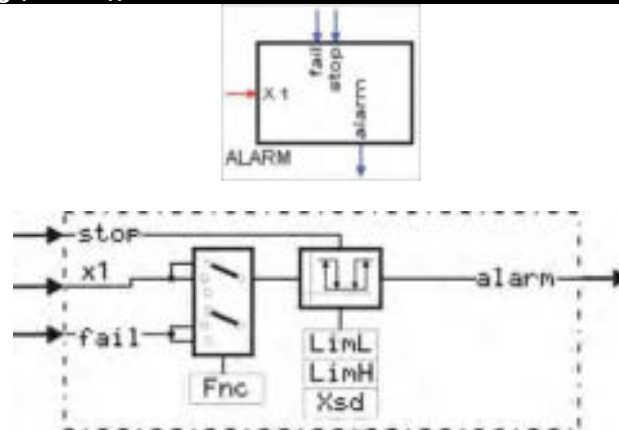
$I1 \dots I8$  The alarm statuses of alarm 1 to alarm 8: 0 = no alarm; 1 = alarm case

**Configuration parameters:**

Parameter	Description	Range	Default
Mode1...Mode8	alarm functions of the 8 alarms	Max-Alarm Min-Alarm	MAX-Alarm MIN-Alarm

**Parameters:**

Parameter	Description	Range	Default
$L1 \dots L8$	Alarm values of alarm 1 to alarm 8	-29 999 ... 999 999	0
$X_{sd}$	Switching hysteresis $X_{sd}$	0 ... 999 999	0

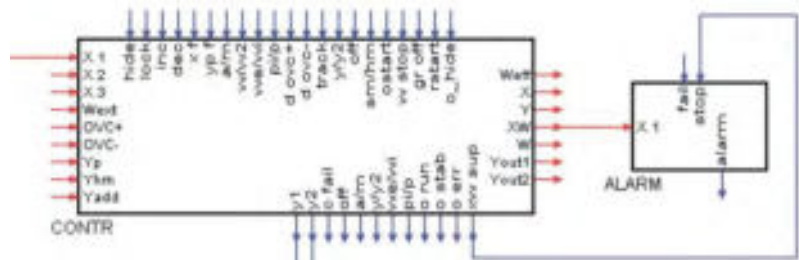
**3.8.6. ALARM (alarm processing (No. 45))**

**x1** is checked for a lower and an upper alarm value. Additionally, digital alarm input **fail** can be used. Configuration parameter **Fnc** can be used to select which signal shall be monitored (**x1**, **x1 + fail** or **fail**).

With input **stop** = 1, alarms (**fail** and **x1**) are suppressed. After removal of this signal, suppression lasts, until the monitored value is again within the limits. This can be used e.g. for suppressing an alarm message with change.

Alarm suppression with change

During value change at the exit **xw sup** a pulse with the length of a scanning cycle **Ts** is sent.

**Inputs/outputs****Digital inputs**

<b>fail</b>	Digital alarm signal e.g. fail signal of AINP
<b>stop</b>	<b>stop</b> = 1, alarms ( <b>fail</b> and <b>x1</b> ) are suppressed. After stop returned to 0, suppression lasts, until the monitored value is again within the limits.

**Analog input**

<b>x1</b>	Input variable to be limited
-----------	------------------------------

**Digital output**

<b>alarm</b>	Alarm status: 0 = no alarm; 1= alarm
--------------	--------------------------------------

**Configuration parameter:**

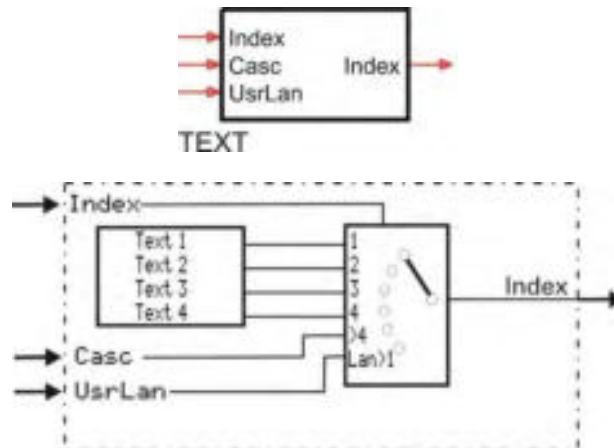
Parameter	Description	Range	Default
<b>Fnc</b>	only <b>x1</b> is monitored	<b>Messw.X1</b>	←
	<b>x1</b> and <b>fail</b> are monitored	<b>X1+fail</b>	
	only <b>fail</b> is monitored	<b>fail</b>	

**Parameters:**

Parameter	Description	Range	Default
<b>LimL</b>	lower limit for the alarm	-29 999 ... 999 999	-10
<b>LimH</b>	upper limit for the alarm	-29 999 ... 999 999	10
<b>Lxsd</b>	Switching hysteresis Xsd	0 ... 999 999	0

### 3.9. Visualization

#### 3.9.1. TEXT (text container with language-dependent selection (No. 79))



The text block contains a list of user texts which can be displayed by various operating pages (programmer, VVERT and ALARM). These texts can be displayed and adjusted as a selection list on a VVERT page (e.g. for plain text selection of recipes).

The function block is cascable, if more than 4 texts are available for selection.

Texts can be entered only via engineering tool: 4 texts of 16 characters

#### Inputs/outputs

##### Analog inputs

<b>Index</b>	Input for text selection
<b>Casc</b>	Cascade input for further text blocks in the same language
<b>UsrLan</b>	Input for a text block with texts in a further language

##### Analog outputs

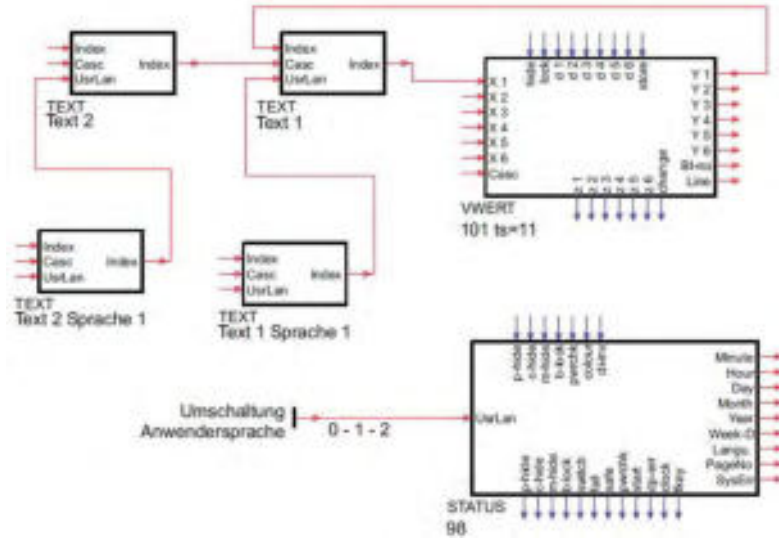
<b>Index</b>	Number of selected text in the text block
--------------	---

Output "Index" of the last text block in a text cascade must be wired to the block on the operating page of which the texts must be used, e.g. VVERT. The number of the display text is allocated to the index input of this text block.

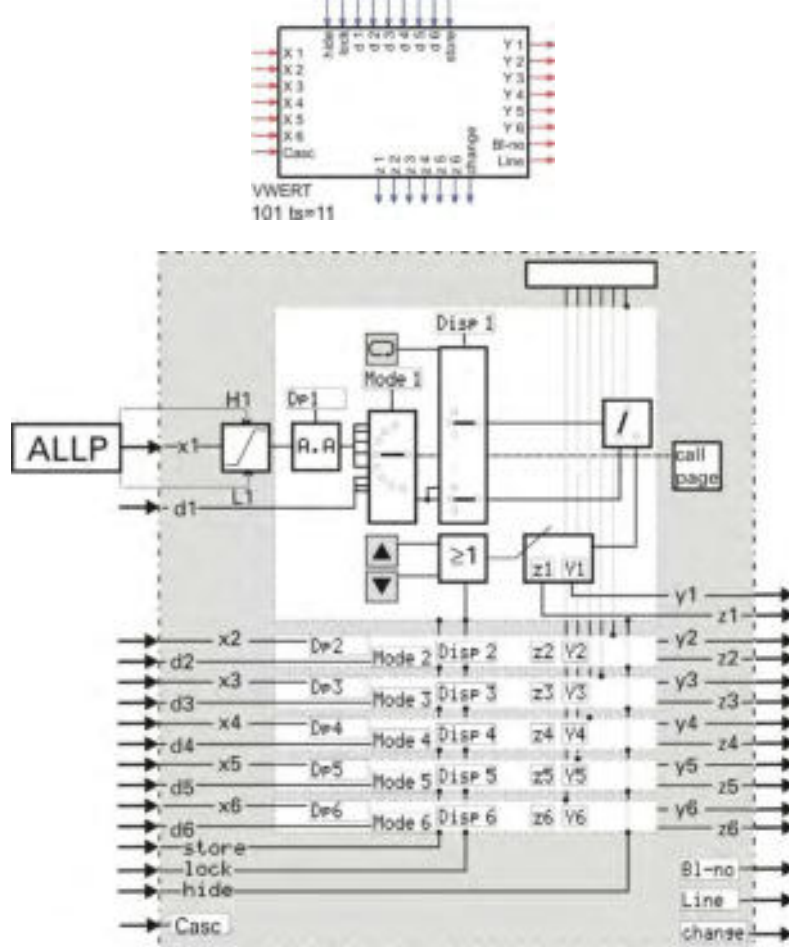
The texts can be extended into any number of texts via the cascade input (Casc). For this, wire the index output of the subordinated block (texts 5 ...8) to input "Casc" of the next text block. The index for text selection is adjustable only at the index input of the last block (see example below).

For user language selection, wire the index output of the (language) text block to language input UsrLan of the used text block. During selection of the user language, its texts will replace the texts of the first text block. User language selection is central at the status block.

Wiring of cascaded text blocks. User language selection is via the status block.



### 3.9.2. VWERT (display / definition of process values (No. 96))




#### General

This function permits display or definition of 6 analog or digital process values in 6 display lines. These values can be changed also via the KS98-1 communication interface. The function block can be cascaded, whereby a scroll field with more than 6 lines can be implemented on the operating page.



- Determination if the display line has digital or analog functions, or if it is switched off is made via configurations (generation of an empty line in the display).
- Possible display functions are: analog, digital, text, menu, push-button, switch and radio button
- Normally, the values applied to the inputs are displayed.
- A value adjustable at the front panel is output at the relevant function output.
- Only adjustable lines are selectable.
- The change of these values from the operating level can be switched off (**lock**)
- Parameters **z1 ... z6** or **y1 ... y6** are used as initial value for the outputs at power-on.
- The output value is displayed only, if the output is fed back to the relevant input, or if the display for this value is in the adjustment mode.
- With a positive flank at the **store** input, the values applied to the remaining inputs are stored in parameters
- **z1 ... z6** and **y1 ... y6** and thus used as output values.

Value changes are stored as parameters **z1 ... z6** or **y1 ... y6** in non-volatile EEPROM. With digital input **lock** set, no values can be changed. With digital input **hide** set, the operating page cannot be displayed. The engineering tool can be used to configure a 16-digit text for the display header and further texts for identification of value and unit, or for the two digital statuses.

 Values of the used analog inputs are stored as parameter values when detecting a positive edge at the store- input. This input should be activated only with relevant changes of the input values. Too frequent storing operations may lead to the destruction of the EEPROM! (→ page Fehler! Textmarke nicht definiert.)

### Inputs/outputs

#### Digital inputs:

<b>hide</b>	Display suppression (with <b>hide</b> = 1 the page is not displayed in the operation.
<b>lock</b>	Adjustment locking (with <b>lock</b> = 1 the values are not adjustable by means of keys   .
<b>d1 ... d6</b>	Process statuses to be displayed. (Default = 0)
<b>store</b>	With a positive flank (0→1) the input values are used as output values.

#### Digital outputs:

<b>z1 ... z6</b>	Valid process values.
<b>change</b>	If a value is changed during operation, the change-output of the VVERT-block is set to 1 for one calculating cycle.

#### Analog inputs:

<b>x1 ... x6</b>	Process values to be displayed (default = 0)
<b>casc</b>	By wiring the casc-input with the bl-no output of another VVERT, cascades can be set up.

#### Analog outputs:

<b>y1 ... y6</b>	Valid process values.
<b>Bl - no</b>	Block number of this output
<b>line</b>	If a value is changed during operation, for one calculating cycle the line-output of the VVERT-block is set to that value that was changed (1-6).

### Parameter and configuration data

Parameter	Description	Range	Default
<b>z1...z6</b>	Start values for digital outputs 1...6 at power-on	0/1	0
<b>y1...y6</b>	Start values for the analog outputs 1...6 at power on	-29999...999999	0

Configuration	Description	Values	Default
<b>Disp1 ... Disp6</b>	Function of display line 1...6	display line, value ajustable	<b>adjustable</b>
		only display line	<b>display</b> ←
		line = empty line	<b>empty</b>
		analog value display	<b>analog</b> ←
		digital value display	<b>digital</b>
<b>Mode1 ... Mode6</b>	Type of display line 1...6	value display in time format	<b>time</b>
		selection group (radio button)	<b>radio</b>
		toggle function	<b>switch</b>
		push-button function (pressed =1)	<b>push-button</b>
		text selection	<b>text</b>
<b>Dp1 ... Dp6</b>	Digits behind decimal point in analog line 1...6	menu function (page changing)	<b>menu</b>
			<b>0 ... 3</b> 0

### Entry and display of texts

Changing the texts displayed in the unit is only possible in the engineering tool! Max. 16 characters can be entered into each text parameter. Dependent on whether a line was configured as an analog, digital, radio, switch, push-button or menu line, all 16 characters (e.g. Mode x = **digital**) or only the first 6 characters (e.g. mode x= **analog**) are shown in the device. Further detailed information on the various display types is given at the end of the section




With digital displays (digital, switch, push-button and radio):

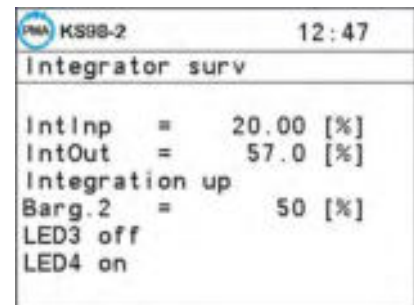
signal = 0: dependent of line of 'Text1 a ... Text6 a'

signal = 1: dependent of line of 'Text1 b ... Text6 b'

## VWERT operating page

The VWERT operating page can be selected in the operating page menu with non-activated 'hide' input. For information on the operation, see section Operating pages on page 39.

-  If a line is configured as display, the value can't be changed.
-  Operation of line modes radio, switch and push-button is described in section I.9.4 "Adjusting values".
-  When using the VWERT block in an engineering, operation and function must be described separately.



### VWERT block cascade connection

For linking several VWERT operating pages, the BI-no output of a further VWERT must be wired with the Casc input of the calling VWERT. Thereby, the last page which must be connected can be linked also to the start page (ring structure).

Cascade connection of a VWERT block is indicated by arrows ▲▼ on the display page. A previous block (BI-no output wiring) above the first line and a next block (Casc input wiring) below the last line are marked, otherwise, these arrows are omitted. When setting the cursor onto one of these arrows and pressing key Enter, a change to the relevant VWERT page occurs. With standard exit from the called up VWERT page a change to the selection list of the operating pages is made.

## Selectable display modes in detail

### ① Data type "analog"

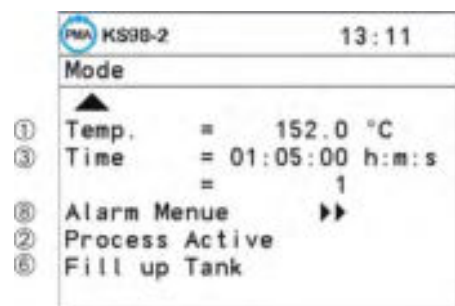
The line contains 2 static texts (6 characters) and the analog value connected to X1...X6. Changing the value is done as described above, if changeability is configured.

*VWERT page with the line modes: marking of previous page, analog, time, text selection, menu, digital, switch*

With the corresponding input Xn wired via a type ALLP function block, its limits H1 (max. limit) and L1 (min. limit) are used as adjustment limits for this value. Unless an ALLP block is connected to the input, limits -29999 to 99999 remain valid.

Example: Value with limits:

Apart from its maximum number of digits behind the decimal point, each value can have its own adjustment limits, which are determined by parameter values L1 and H1 of a previous ALLP block. Unless the source of the display value is the VWERT itself, the ALLP uses these parameter values for limiting the value.



### ② Data type "digital"

Dependent digital input bit value of the relevant line, text "0" (Name\_n) or text "1" (Unit\_n) is displayed. With a static input value, static text output can be generated (e.g. headline).

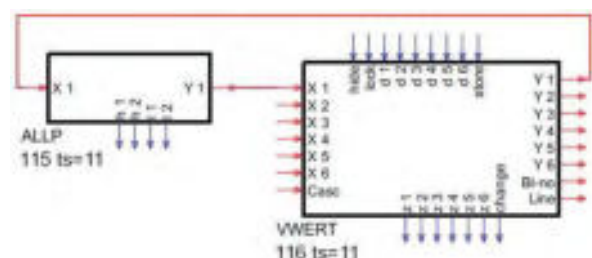
Example: value with limits:

In addition to its maximum number of digits behind the decimal point, each value can have its adjustment limits which are determined by parameter values L1 and H1 of a connected ALLP block. Unless the display value source is the VWERT itself, the ALLP limits the value with these parameter values.

### ③ Data type "time" (analog output)

Data type "time" can be used to display or to adjust times in HH:MM:SS or HH:MM, whereby the last decimal digit indicates the full minutes. The digits behind the decimal point are for display of seconds.

- With the digits behind the decimal point DP set to 0, adjustment of seconds is not possible. Only hours and



minutes can be adjusted. If the relevant configuration value DP is 2, adjustment of seconds is also possible. Wird die entsprechende Nachkomma- Konfiguration DP auf 0 gesetzt, so ist eine Einstellung der Sekunden nicht möglich. Es können nur Stunden und Minuten verstellt werden. Ist der Wert des entsprechenden Konfigurationswertes DP gleich 2, so ist eine Verstellung der Sekunden ebenfalls möglich.

- From a time above 100 hours, seconds are not displayed any more.
- The adjustment range is within 00:00:00 - 15999:59 hours. Due to the limited resolution of a float value, adjustment is possible only in steps of 6 seconds from value 16:40:00.

### ④ Data type "radio" (radio button; digital output)

Data type "radio button" can be used for changing over between combined selection fields.

- After selecting, changing is done directly without starting with function key **7**.
- "Radio button" arranged successively in one VVERT form a common group. Radiobutton.
- Only one element of this group is activated.
- By actuating the function key, the radio button on which the cursor is standing is activated. All relevant other ones are de-activated.
- A new group starts, if another data type is defined between 2 radio buttons.
- Unless a radio button is activated during data transmission to VVERT, all radio buttons remain inactive. If more than 1 button is active, the 1st one of the group is activated, the other ones are inactive. Wird bei der Übertragung der Daten zum VVERT kein Radiobutton eingeschaltet, so bleiben alle ausgeschaltet. Ist mehr als 1 Button aktiv, so wird der 1. der Gruppe aktiviert, die weiteren sind inaktiv.

### ⑤ Data type "switch" (digital output)

Data type "switch" can be used to implement switch-on/switch-off (toggle) functions.

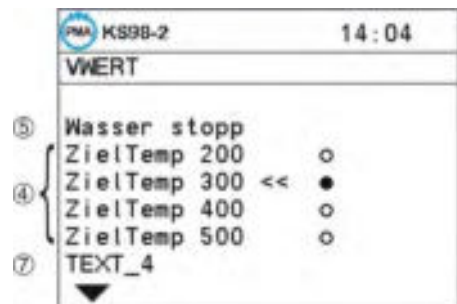
- After selection, adjusting is done directly without starting with function **7**.
- Pressing the function key will activate a de-activated switch and de-activate an activated switch.

*VVERT-Seite mit den Zeilenmodi: Taster, Radio, Textliste und Folgeseitenmarkierung*

### ⑥ Data type "push-button" (digital output)

Data type "push-button" can be used to realize short switch-on/switch-off (hold) functions.

- After selection, adjusting is done directly without starting with function key **7**.
- As long as the function key is pressed, the output is activated. When releasing the key, the output is de-activated.



### ⑦ Data type "text" (analog output, see also: Function block TEXT)

Data type "text" can be used for display of indexed texts for analog integer signals. Moreover, an analog value can be allocated to a text when adjusting.

- The corresponding input must be connected with the index output of a text block.
- The number of the text to be selected (VVERT output Y1...Y6) is applied to the index input of the first (next to VVERT) text block.
- The text blocks cascaded by wiring the index output of another text block with the Casc input of the relevant text block. Text selection is always via the index input of the text block next to VVERT.
- Via the UsrLan input, text blocks of different language can be appended. Language switchover (language index) is defined by the value at the UsrLan input of status block 98. Unless an appropriate text block for the language is available (e.g. language index too high), the corresponding text is output in the last language block found.
- When selecting a text in the VVERT to be displayed, the number of selectable texts is limited by the number of connected text blocks.
- If the index for text selection comes from a different origin, no text is displayed if the index is beyond the possible text selection ( 0 or >max ). VVERT marks the line with "\_\_\_\_\_".
- With text selection at VVERT, the initial value ( parameter Y1...Y6) should be set > 0 to avoid a start value of "\_\_\_\_\_".

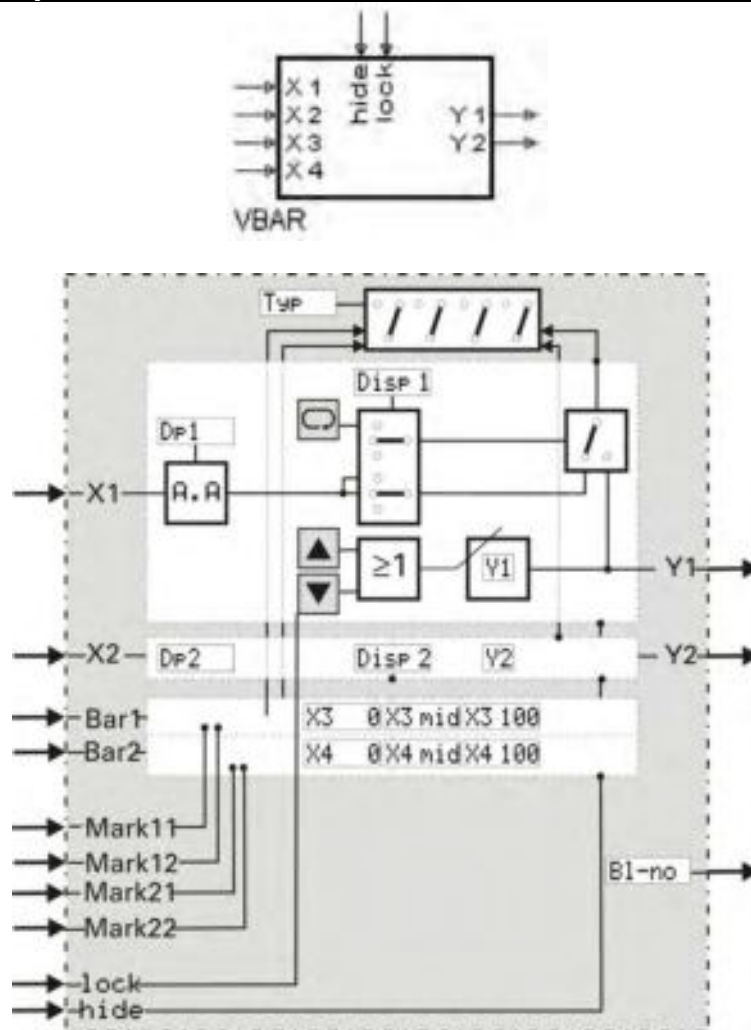


### ⑧ Data type "menu"

Data type "menu " can be used for changing to other operating pages (menu without sub-menu items, linking not possible).

- The value applied to the corresponding input is interpreted as block number of the operating page to which changing is required.
- Changing to the specified page is done by pressing the Enter key. Unless the page is accessible, a change to the operating page selection list is made. This list contains all the blocks which are available for selection. If a page is not accessible, the reasons may be:
  1. Block number not defined
  2. Block number does not have an operating page
  3. Block cannot be displayed instantaneously because hide = 1.
- When making a standard exit from the operating page, return is to the calling VVERT page.
- When using this procedure for changing to a VVERT operating page which also contains a menu type line, no further change will occur.

### 3.9.3. VBAR (bargraph display (No. 97))




#### General

This function permits the display of 2 analog input signals as bargraphs, and of 2 analog input signals as numeric values. Moreover, two analog output signals can be defined. Another 4 analog inputs can be used to position 2 markings at each side of the bar within the bargraph range. These markings can be e.g. alarm limits or reference values. With open marker inputs, or out-of-range marker values, display of markers is suppressed.



- Determination of horizontal or vertical bargraph is via configuration (type)
- Determination if value displays are visible or switched off
- Configuration of start values **x3mid** or **x4mid** determines, if the bargraph points only in one direction (from top or bottom) or in 2 directions from the middle.
- The values applied to the inputs are displayed.
- A value which is adjustable via the front panel is output at the relevant analog output.
- Changing these values from the operating level can be suppressed.
- Parameters **Y1** / **Y2** are the initial values for Power-On.
- The output value is displayed only with the output fed back to the relevant input, or if the display for this value is in the adjustment mode.
- Value changes are stored as parameters **Y1** / **Y2** in non-volatile EEPROM.
- With a positive edge at the **store** input, the values applied to the signal inputs are stored as parameters **y1** and **y2**, i.e. as output values.
- When an ALLP is connected at inputs x1 and x2, its limits L1 and H1 are used for parameter adjustment.

With digital input **lock** set, no values can be changed. With digital input **hide** set, the operating page cannot be displayed during operation. A 16-digit text for the display header can be adjusted user-specifically via the engineering tool. The same is applicable for further texts for identification of value and unit.

 Values of the used analog inputs are stored as parameter values when detecting a positive edge at the store- input. This input should be activated only with relevant changes of the input values. Too frequent storing operations may lead to the destruction of the EEPROM!! (→ Seite Fehler! Textmarke nicht definiert.)

### Inputs/outputs

#### Digital inputs:

<b>hide</b>	Display suppression (with hide = 1 the operating page is not displayed).
<b>lock</b>	Blockierung der Verstellung (Bei <b>lock</b> = 1 sind die Werte nicht mittels der Tasten   verstellbar).
<b>store</b>	With a positive edge (0→1), the input values are stored in the EEPROM and adopted as output values.

#### Analog inputs:

<b>X1 / X2</b>	Process values to be displayed as values (default = 0)
<b>X3 / X4</b>	Process values to be displayed as bargraph (default = 0)
<b>Mark 11</b>	Mark on the first bar
<b>Mark 12</b>	Mark on the second bar
<b>Mark 21</b>	Mark on the third bar
<b>Mark 22</b>	Mark on the fourth bar

#### Analog outputs:

<b>y1 / y2</b>	Valid process values.
<b>BL-no</b>	own block number

### Parameter and configuration data

Parameter	Description	Range	Default
<b>Y1 / Y2</b>	Start values at power-on.	-29999...999999	0

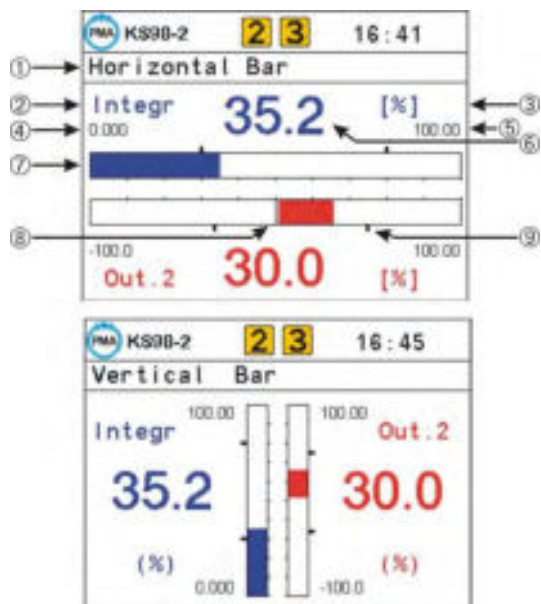
Configuration	Description	Values	Default
<b>Disp1</b>	Function of numeric display 1 and 2	display x1 / x2, value adjustable	<b>Display+adj</b>
<b>Disp2</b>		only display x1 / x2	<b>Display</b>
		X1 / x2 blank line	<b>Empty</b>
<b>Dp1 / Dp2</b>	Digits behind the decimal point in numeric display 1 / 2	0 ... 3	0
<b>Typ</b>	Position of bargraphs	Both bargraphs horizontal	<b>horizont.</b>
		Both bargraphs vertical	<b>vertical</b>
<b>X3 0</b>	Display scaling bargraph 1, 0% (left or bottom end)	-29999...999999	0
<b>X3 100</b>	Display scaling bargraph 1, 100% (right or upper end)	-29999...999999	100
<b>X3 mid</b>	Display scaling bargraph 1, start value (middle)	-29999...999999	0
<b>X4 0</b>	Display scaling bargraph 2, 0% (left or bottom end)	-29999...999999	0
<b>X4 100</b>	Display scaling bargraph 2, 100% (right or upper end)	-29999...999999	100
<b>X4 mid</b>	Display scaling bargraph 2, start value (middle)	-29999...999999	0

## VBAR operating page

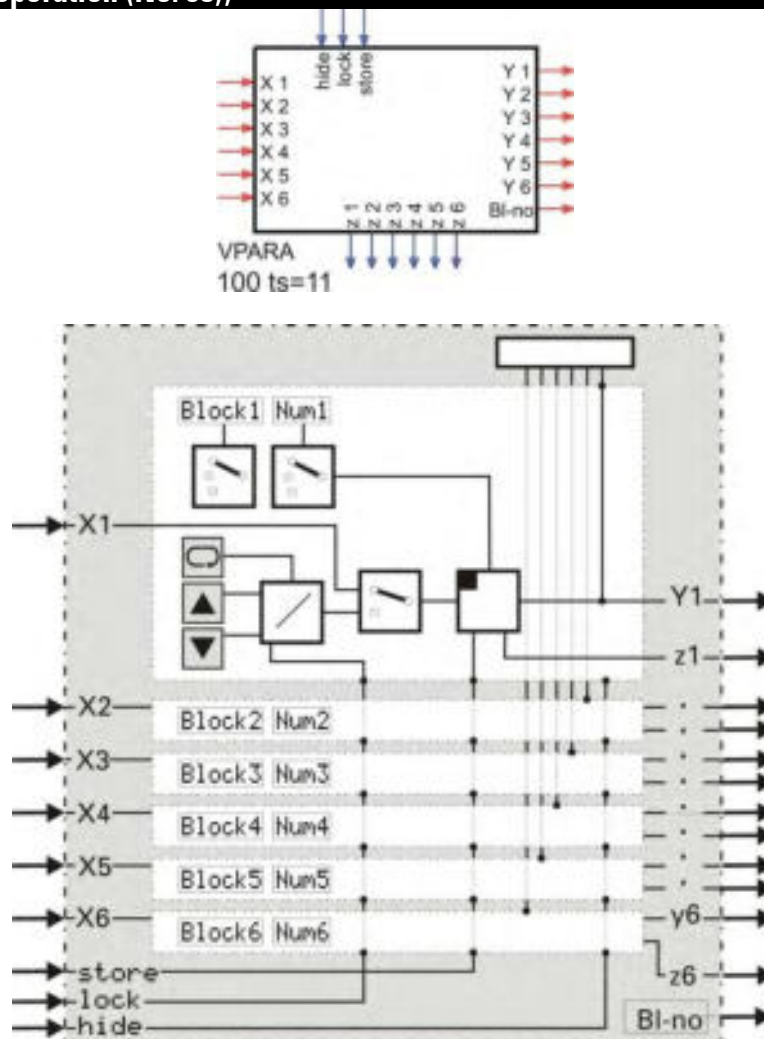
VBAR has an operating page, which can be selected in the operating page menu with the 'hide' input not used. Changing the texts displayed in the unit is only possible in the engineering tool! Max.16 characters can be entered in each text parameter. A value configured as a display cannot be changed.

If a value is configured as a display, this value can not be changed.

- ① Title
- ② Process value name for X1  
(first 6 characters of 'Name 1')
- ③ Unit for X1  
(first 6 characters of 'Unit 1')
- ④ Scale start of bar for value X1
- ⑤ Scale end of bar for value X1
- ⑥ Process value display/input field
- ⑦ Bar for value X1
- ⑧ Middle of bar X2 (starting point)
- ⑨ Marker at right/bottom bar for X2 (the same applies accordingly to the other bar)



### 3.9.4. VPARA (parameter operation (No. 98))



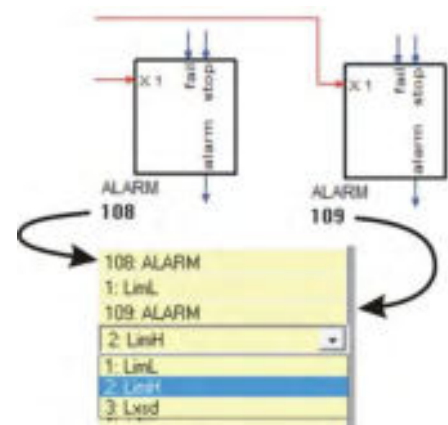
## General

Function VPARA provides an operating page which can be used for changing max. 6 parameters of other function blocks available in the engineering from the operating level.

Each parameter to be displayed is made known to the display function with block number and parameter number by means of two configuration data. The engineering tool supports parameter setting by a special operating sequence in which the parameter numbers of the selected block are selected by means of the parameter descriptions (→ see figure opposite).



Additionally, an identifier and a unit text can be specified. Values of the analog inputs used are accepted as parameter values if a positive edge is detected at the **store** input.

☞ Activation of this input must be organized so that it occurs only with relevant input value changes. Too frequent storage can lead to EEPROM destruction (→ page Fehler! Textmarke nicht definiert.)



### Inputs/outputs

#### Digital inputs:

<b>hide</b>	Display suppression (with <b>hide</b> = 1 the page is not displayed in the operation).
<b>lock</b>	Adjustment blocking (with <b>lock</b> = 1 the values are not adjustable by means of keys   ).
<b>store</b>	With a positive flank (0→1) the input values are stored as parameter values.

#### Digital outputs:

<b>z1 ... z6</b>	The outputs provide a status, which shows if the last storage of the values taken over from the inputs was successful (z1 ... z6 = 0). Errors may occur due to exceeded limits of the parameter value or due to non-existing parameters (z1 ... z6 = 1).
------------------	--

#### Analog inputs:

<b>X1 ... X6</b>	Process values to be stored as parameter values (default = 0)
------------------	---

#### Analog outputs:

<b>y1 ... y6</b>	The values of the 6 parameters are output at the analog outputs. Unused parameters provide value '0'.
<b>BL - no</b>	Block number of this output

### Parameter and configuration data

Configuration	Description	Values	Default
<b>Block1 ... Block6</b>	Block number of parameter to be displayed	*	*
<b>Num1 ... Num6</b>	Parameter number	*	*

\* To avoid confusions and thus operating errors, we recommend adjusting block numbers and parameters exclusively via the engineering tool, where the parameters with their short-form descriptions must also be specified. Text entry is only possible via the engineering tool.

### Entry and display of texts

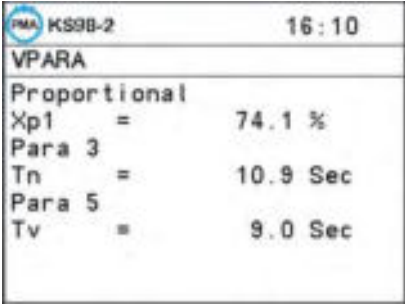
Changing the texts displayed in the unit is possible only in the engineering tool! Max. 16 characters can be entered in each text parameter. Dependent of whether a line is allocated to a block number or defined as a text line, all characters (**Blockx = Text**) or only the first 6 characters (**Blockx = #xxx**) are displayed in the unit. If parameter number (**Numx**) or block number (**Blockx**) are undefined, '?????' is displayed as a value.

Parameter allocation to the display lines:

Block1; Num1; Text1; Einh.1 → Zeile 1 .... Block6; Num6; Text6; Einh.6 → Zeile 6

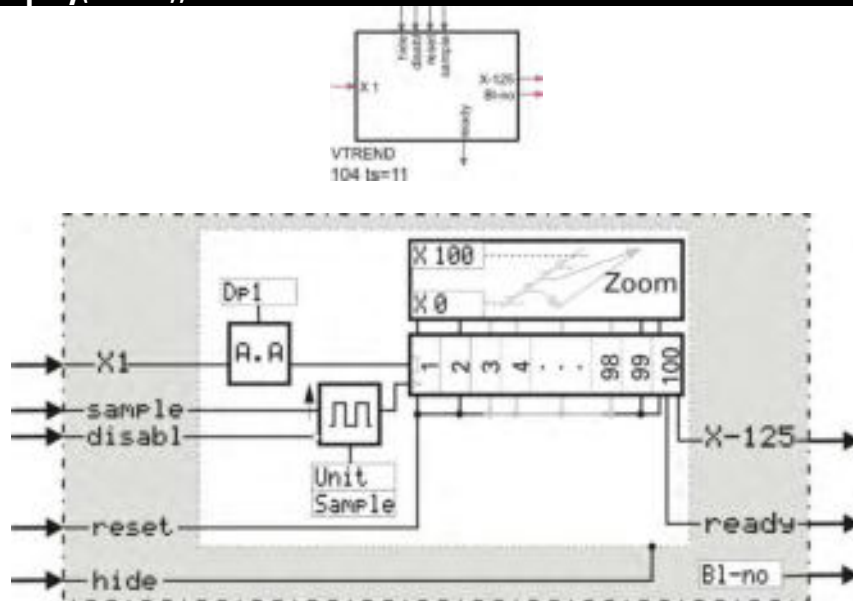
### VPARA operating page

VPARA has an operating page, which can be selected in the operating menu with input 'hide' not used.



KSSB-2		16:10
VPARA		
Proportional		
Xp1	=	74.1 %
Para 3		
Tn	=	10.9 Sec
Para 5		
Tv	=	9.0 Sec

### 3.9.5. VTREND (trend display(No. 99))






#### General


Function VTREND collects 125 values of the analog input 'x1' in a shift register and permits value display as a trend curve. When the shift register is filled with 125 values, the value 125 samples ago is overwritten by a new value. With input sample 'sample' not used, data recording is synchronous with the time units specified in the configuration. Trigger pulses at the 'sample' input permit asynchronous data recording.

The properties of KS98-1 function block VTREND are:

1. The resolution of the KS98-1 Y axis is 60 pixels.
2. The resolution of the X axis is 125 pixels.
3. With further trend blocks connected at the output of a trend block (cascade), these blocks can be viewed by shifting the time axis (scrolling the time axis)
4. The Y resolution can be magnified by factor 4, and scrolling throughout the range is possible in steps of 12,5%. The zero shift remains unchanged in the background when returning to normal resolution.
5. The old settings also remain unchanged when leaving the operating page and calling it up again.
6. The lower scanning time limit is set to 0,01 for unit hours.
7. Output Bl-no provides the operating page block number.

5 accesses via the communication interface, each providing data packages of 25 trend data from KS 98-1, are available.

-  When connecting 2 trend blocks to one trend output by mistake in the case of a cascade, the one with the lower number is ignored. The number of cascaded blocks is not limited.
-  If blocks in the chain have different scanning times or different ranges, data display is faulty. No warning is output. The trend display continues when scrolling in the time axis (paging into the past) rather than being stopped.
-  With voltage failure, the sampled values remain unchanged.

 The changing of the texts displayed, is only possible in the engineering-tool! Max. 16 digits can be entered for every textparameter.

### Inputs/outputs

#### Digital inputs:

hide	Display suppression (with hide = 1 the page in the operation is not displayed).
disable	The digital input can be used to interrupt automatic or triggered sampling (high-active).
reset	The digital input deletes the shift register and resets trend measurement.
sample	If the digital input is wired, sampling is triggered by a positive flank (0 → 1) at this input. In this case, the adjusted sampling interval (configuration) is not effective.

#### Digital outputs:

ready	After filling the shift register with 100 values first, the digital output is set to high.
-------	--

#### Analog inputs:

x1	Process value to be displayed as trend (default = 0)
----	--


#### Analog outputs:

X-100	The value of the shift registers which is overwritten by the next sample value is provided at the analog output (value 100 samples ago).
BL-no	Block number of this output

### Configurations data

Configuration	Description	Values	Default
Unit	Unit of sampling interval	Seconds (s) Minutes (m) Hours (h)	sec. min. h. ←
Sample	Value of sampling interval in the unit defined with 'Unit'	0,2...3600	1
Dp	Digits behind the decimal point for value displays	0 ... 3	0
X 0	Display scaling start value (0%)	-29999...999999	0
X100	Display scaling end value (100%)	-29999...999999	100

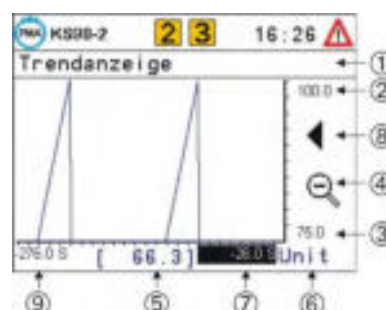
### Text display and entry

 Changing the texts displayed in the unit is possible only in the engineering tool!  
Max. 16 characters can be entered into each text parameter

#### VTREND operating page

VTREND has an operating page which can be selected in the operating page menu with input 'hide' no connected. The operating page is provided only for display of trend data. Making changes in the input fields will change only the visible trend display section, without making changes to the data.

- ① Title
- ② ③ scale end values
- ④ Zoom switchover
- ⑤ Value at time ⑦ / actual input value
- ⑥ Unit of value
- ⑦ Origin (start) of time axis related to the actual value (=0) shift of time axis (scrolling into the past)
- ⑧ Axis shift signalling
- ⑨ End of time axis / earliest value in the displayed trend





## Examples:

### Trend recording with 2 curves

Although distinction of different curves is not possible, display of two values on a trend page may be purposeful (e.g. controller and process value, or one value and zero, in order to have a curve).

In the example, a clock triggering switch-over between the values together with the SELV1 is generated by means of a pulse.

In the example, a TONOFF can be used to generate a clock for switching over between values with SELV1.

E.g. for making a record in VTREND at intervals of a second, unit is s and sample is 1.

In order for the TONOFF to change between 0 and 1 once per second, T1 and T2 must be set to 0,9 s. One cycle (0.1s) is lost for detection of the own output change).

In the following example, a pulse is used to create a clock which generates switchover between values with SELV1.

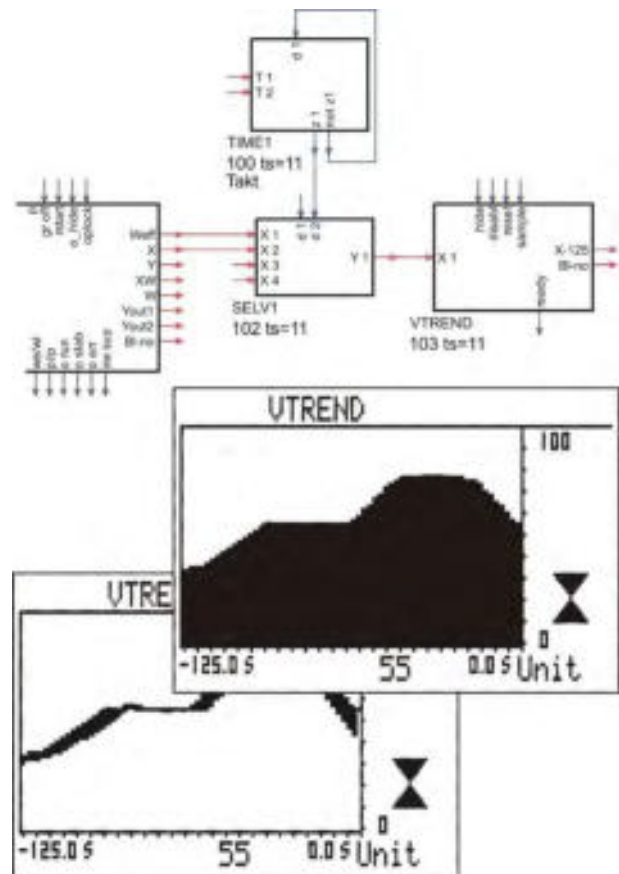
For making a record at intervals of a second e.g. in VTREND, Unit is set to s and Sample is set to 1..

### Settings:

Unit = s and S

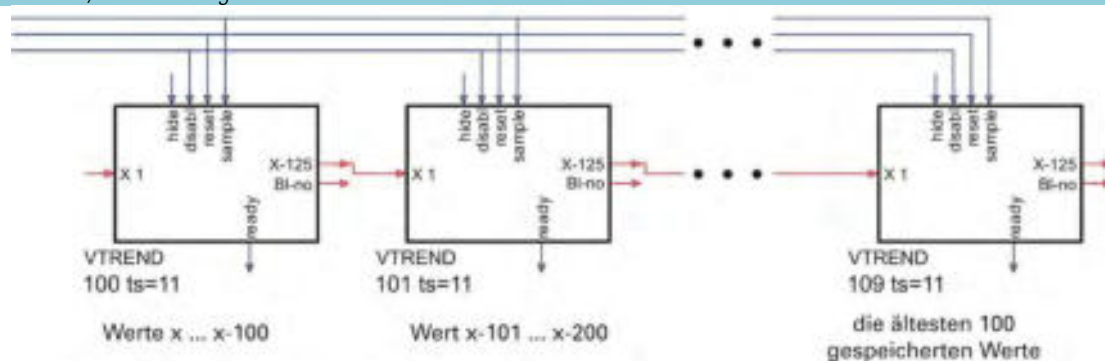
ample =  $1 \triangleq 1/s = 3600/h$

x0=0, x100 and pulse/h to 3600, 1/2 sample interval = 1800 must be applied to pulse input x1.



## Cascading

### Example of trend- /datarecording with n values



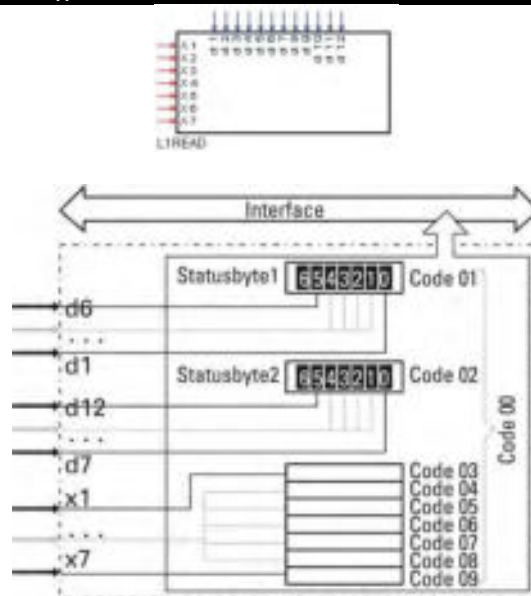
Trend or data recording with any number of values can be realized by cascading VTREND function blocks. Limiting refers only to the number of available block numbers and to the calculation time. The data sequence is dependent of VTREND function block wiring. In wiring direction, the block numbers must be ascending.

## 3.10. Communication

### ISO 1745

In total, max. 20 L1READ and L1WRIT functions can be configured (blocks 1...20 ), any combination of functions is possible. Any number of data can be used in the functions.

#### 3.10.1. L1READ (read level1 data(No. 100))



#### General

Any 7 analog process values (x1...x7) and any 12 digital status informations (d1...d12) of the engineering are composed into a data set for the digital interface. The digital interface can read the data set as a complete block with code 00, function number 0, or the individual values with codes 01...09, function number 0.

#### Inputs/outputs

##### Digital inputs:

- d1 ... d6 Digital process values, which can be read via interface (status byte 1). (Default = 0)
- d7 ... d12 Digital process values, which can be read via interface (status byte 2). (Default = 0)

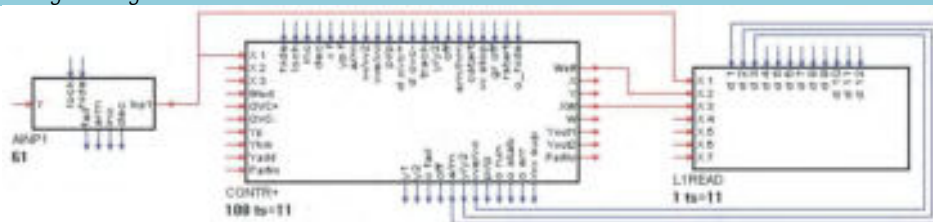
##### Analog inputs:

- x1 ... x7 Analog process values, which can be read via interface. (Default = 0)

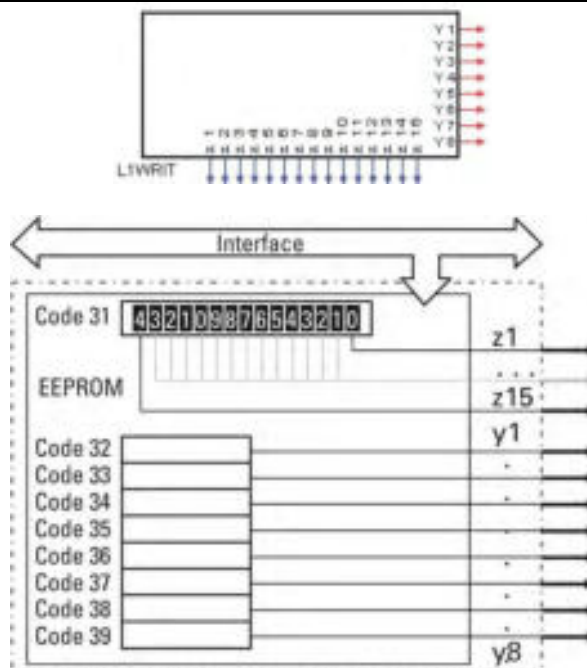
#### Engineering example

In the following example, several process data (process value, effective and control deviation) and controller statuses (automatic/ manual, Wint/Wext and y/Y2) are connected with the L1READ function block. Now, these data can be read in a message via interface.

##### Example for L1READ engineering



### 3.10.2. L1WRIT (write level1 data (No. 101))



#### General

This function is used to provide a data set transmitted by the interface to the engineering. The digital interface describes EEPROM cells with codes 31...39, function number 0. The data set comprises 8 analog process values (y1...y8) and 15 digital control informations (z1...z15), which are provided to the engineering.

- i** The transmitted data are stored in the EEPROM. After power failure, start is with the data rather than with the default values.

#### Inputs/outputs

Digital outputs:

**z1 ... z15** Digital process values, which can be written via the interface (default = 0)

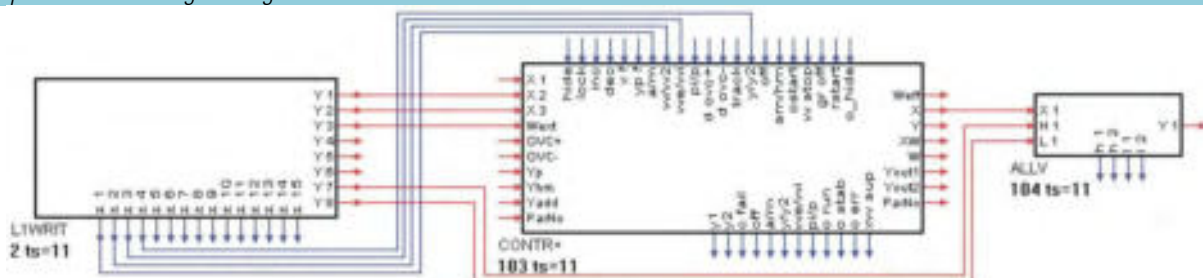
Analog outputs:

**y1 ... y8** Analog process values, which can be written via the interface (default = 0)

#### Engineering example

In the following example, the L1WRIT function block is used to make several process data (process values x2, x3, external and two alarm limits) and the control information (automatic/manual, w/W2, Wint/Wext and y/Y2) available to the engineering. These data can be written in a message via interface.

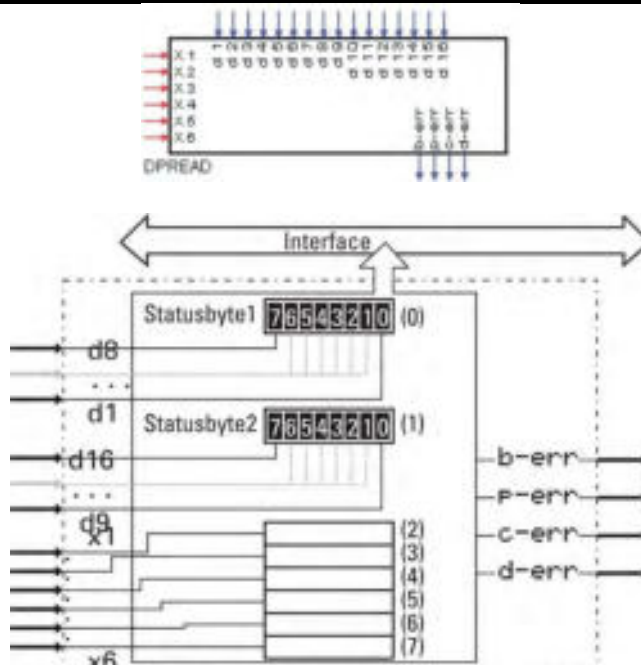
#### Example for L1WRIT engineering



## PROFIBUS

Max. 4 functions DPREAD and DPWRIT can be configured (blocks 1...4 or 11...14 ). Any combination of functions is possible. Any data can be used in the functions.

### 3.10.3. DPREAD (read level1 data via PROFIBUS (No. 102))



#### General

Block numbers 1...4. Any 6 analog process values (x1...x6) and any 16 digital process values (d1...d16) of the engineering are composed for scanning via a PROFIBUS data channel. Block number 1 provides the data for channel 1, block number 2 provides the data for channel 2, etc.

The PROFIBUS module reads the data of two channels at intervals of 100 ms. The digital outputs indicate the PROFIBUS status

**i** Further information on communication with PROFIBUS is given in the interface description (order no.: 9499 940 52711).

#### Inputs/outputs

##### Digital inputs:

- d1 ... d8 Digital process values, which can be read via the PROFIBUS (status byte 1)
- d9 ... d16 Digitale process values, which can be read via the PROFIBUS (status byte 2)

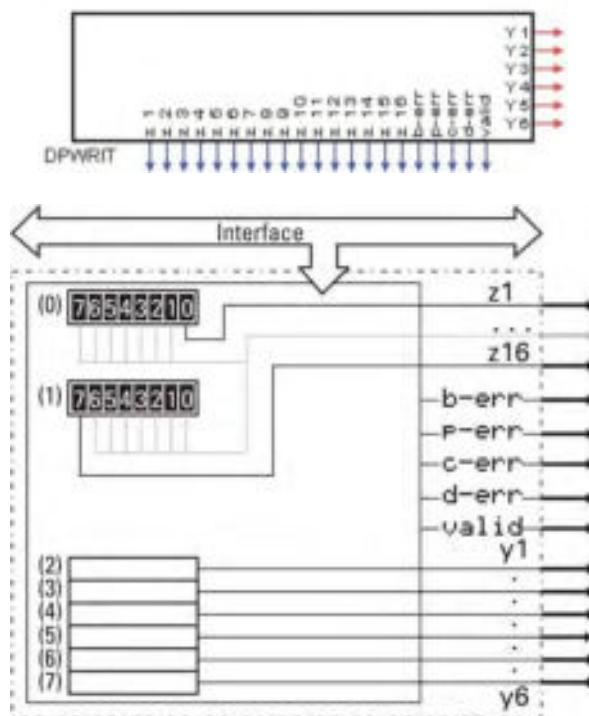
##### Digital outputs:

- b-err PROFIBUS status: 1 = bus access not successful
- p-err PROFIBUS status: 1 = faulty parameter setting
- c-err PROFIBUS status: 1 = faulty configuration
- d-err PROFIBUS status: 1 = no data communication

##### Analoge Eingänge:

- x1 ... x6 Analog process values, which can be read via the PROFIBUS

### 3.10.4. DPWRIT (write level1 data via PROFIBUS (No. 103))



#### General

Block numbers 11...14. The data of a PROFIBUS data channel are transmitted into the memory. Block number 11 transmits the data of channel 1, block number 12 transmits the data of channel 2, etc. The PROFIBUS module writes the data of two channels at intervals of 100ms. The data set comprises 6 analog process values (y1...y6) and 16 digital status informations (z1...z16), which are available to the engineering. The digital outputs (b-err, p-err, c-err, d-err and valid) indicate the PROFIBUS status.

**i** Further information on communication with PROFIBUS is given in the interface description (order no.: 9499 940 52711).

#### Inputs/outputs

##### Digital outputs:

z1 ... z16	Digital process values, which can be written via the Profibus.
b-err	PROFIBUS status: 1 = bus access not successful
p-err	PROFIBUS status: 1 = faulty parameter setting
c-err	PROFIBUS status: 1 = faulty configuration
d-err	PROFIBUS status: 1 = no data communication
valid	PROFIBUS status: 1 = data o.k.

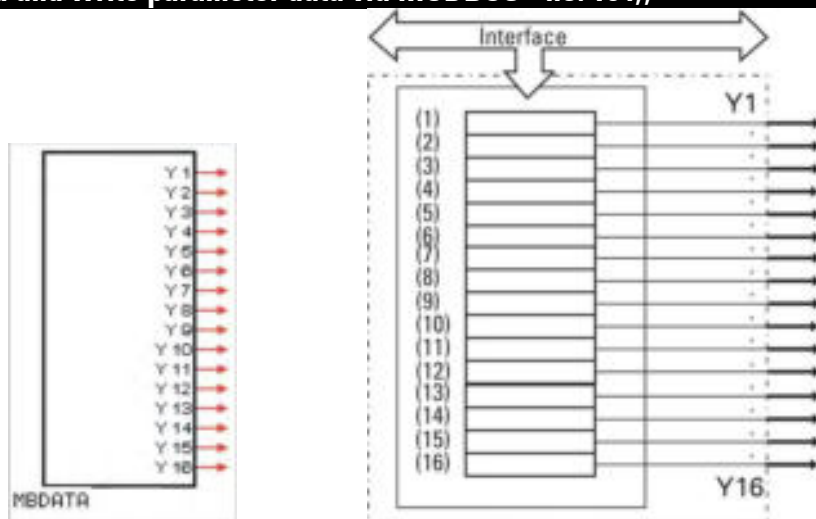
##### Analog outputs:

y1 ... y6	Analog process values, which can be written via the Profibus.
-----------	---

## MODBUS

In total, a maximum of 5 function blocks can be configured. Any combination of functions is possible. In the functions, any data may be used.

### 3.10.5. MBDATA (read and write parameter data via MODBUS - no. 104))



#### General

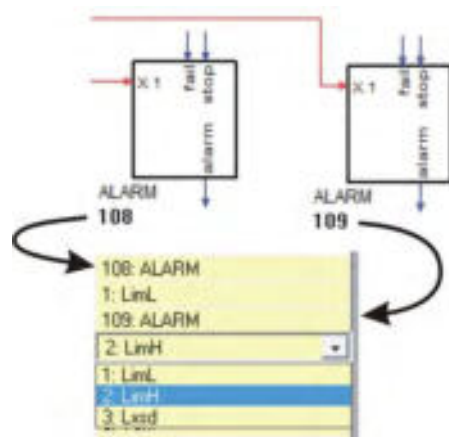
The new MBDATA function block behaves analogously to the already known function block VPARA and provides access via MODBUS. Thus up to 16 parameters of other function blocks available in the engineering can be read or changed via MODBUS.

Each parameter is declared to the MODBUS function including block and parameter number by means of two configuration data.

The engineering tool supports parameter setting using a special operating sequence in which the parameter numbers of the selected block are selected by means of the parameter descriptions ( → refer to the drawing opposite).

For additional information:

see KS98-1 Modbus interface description "sb\_ks98-1\_mod\_e\_9499-040-88711.pdf".



#### Inputs/outputs

Analog outputs:

Y1...Y16

Analog process values that can be read or written via interface (default = value of the assigned parameter or "0"). The values of the 16 parameters are output. Unused parameters provide a value of '0'.

#### Configuration data

Configuration	Description	Values	Default
Block1...Block16	Block number of the parameter	--*	--*
Num 1...Num 16	Number of the parameter	--*	--*

- \* To avoid confusion and thus operator errors, we recommend setting the block numbers and parameters exclusively using the engineering tool where the parameters must be specified with their short-form descriptions.

### 3.11. I/O extensions with CANopen

The additional CANopen interface completes the functionality of the multifunction unit basic version by:

- local I/O extensibility using the PMA RM 200 modular I/O system
- connection of the PMA multiple-channel temperature controllers with CANopen interface
- on-site data exchange with other KS98-1 (cross communication)



#### BUS terminating resistor

Both ends of the CANopen bus must be provided with a bus terminating resistor at (first and last node). For this, the bus terminating resistor provided in every KS98-2 can be used. With the SIL switch closed, the terminating resistor is activated (→ page 23).

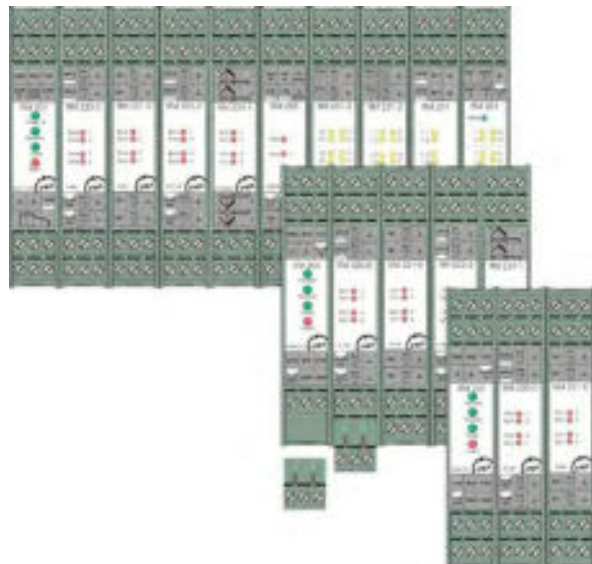
Status display: Status CAN bus → Chapter 1.10.7

#### 3.11.1. RM 211, RM212 and RM213 basic modules

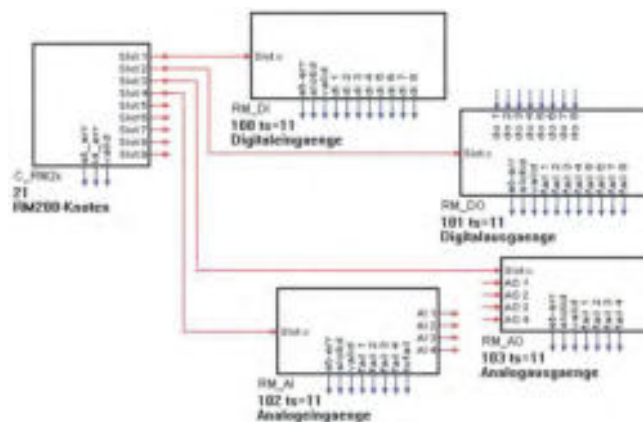
The RM 200 system is a basic module (housing) for snap-on rail mounting with 3, 5 or 10 slots. The left socket is always reserved for the RM 201 CANopen bus coupler module. Dependent of requirement, I/O modules or dummies can be plugged into the remaining sockets. The modules click in position in the basic module and can be released using simple tools for replacement (e.g. small screwdriver).

⚠ Wiring in the engineering-tool must be according to real wiring (Position = Slot = socket).

⚠ Don't insert or remove modules with the supply voltage switched on.

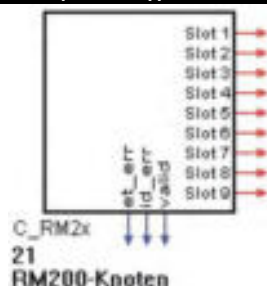


*Partial engineering for communication with an RM200 node.*





### 3.11.2. C\_RM2x (CANopen fieldbuscoupler RM 201 (No. 14))



Coupler module RM201 is fitted with an interface to the CAN bus and plugs into the first slot. The other slots are provided for various I/O modules, which are polled cyclically via an internal bus.

#### Outputs

##### Analog Outputs

Slot1... Slot9 Connection of RM modules RM\_DI, RM\_DO, RM\_AI and RM\_AO

##### Digital Outputs

et-err	0:	no engineering error detected
	1:	reply from min. 2 nodes with identical node ID; r Change the addresses of connected instruments accordingly (e.g. DIP switches on RM 201).
id-err	0:	correct node Id
	1:	wrong communication module ID no reply from any unit with the specified node ID; r Adjust the DIP switches on the connected RM 201 and on page "Parameter Dialog C_RM2x.
valid	0:	invalid data
	1:	data are valid

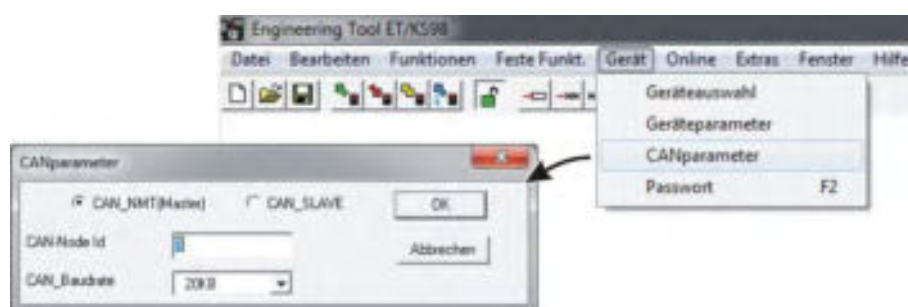
Unlike the other KS 98-1 functions, only one data function may be connected to the analog outputs.

#### Parameters and configuration data

Parameter	Description	Range	Default
NodeId	RM201node address	2...42	32

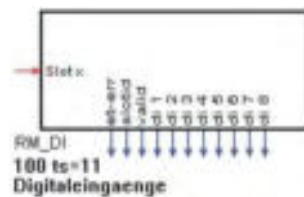
Prerequisite for communication between KS98-1 multifunction unit and CANopen field bus coupler RM 201 is that the CAN parameter setting is identical.

Adapt engineering tool settings and RM201 fieldbus coupler switch position.





### 3.11.3. RM\_DI (RM 200 - (digital input module (No. 15))



Function **RM\_DI** handles the data from the connected digital input modules.

#### Inputs and outputs

##### Analog input

**Slotx** Connection of one of the slot outputs of the RM200 node (C\_RM2x).

##### Digital outputs

**et-err** 0 = no engineering error detected  
1 = engineering error (several RM module functions at a slot)

**slotid** 0 = correct slot assignment  
1 = faulty slot assignment (wrong RM module inserted)

**valid** 0 = no data  
1 = Daten konnten empfangen werden

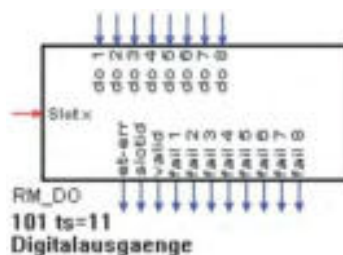
**di 1...di 8** 1st to 8th digital input signal

\* Slot = conn.-no., socket e.g. 2...10

#### Parameter and configuration data

Configuration	Description	Range	Default
<b>MTyp</b>	Module type	0: RM241 = 4 x 24 VDC 1: RM242 = 8 x 24 VDC 0 2: RM243 = 4 x 243 VAC	0
<b>Inv1</b>	Direct or inverse output of input signal 1	direct / invers	direct
...	...		
<b>Inv8</b>	Direct or inverse output of input signal 8		

### 3.11.4. RM\_DO (RM 200 - (digital output module (No. 16))



Function **RM\_DO** handles the data from connected digital output modules.

#### Input and output modules

##### Analog input

**Slotx** Connection of one of the slot outputs of the RM200 node (C\_RM2x)

##### Digital inputs

**do 1...do 8** setpoint for digital inputs 1 to 8

## Digital outputs

<b>et-err</b>	0 = no engineering error detected 1 = engineering error (several RM module functions at a slot)
<b>slotid</b>	0 = correct slot assignment 1 = faulty slot assignment (faulty RM module fitted)
<b>valid</b>	0 = no data 1 = data could be received
<b>fail 1...fail 8</b>	1st to 8th digital input signal

## Parameter and configuration data

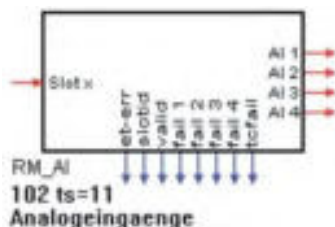
Configuration	Description	Range	Default
<b>MTyp</b>	Module type	0: RM251 = 8 x 24 VDC, 0,5 A 1: RM252 = 4 x Relais(230 VDC) 2 A	0
<b>Inv1</b>	Direct or inverse output of input signal 1?	direct / invers	direct
<b>Inv8</b>	Direct or inverse output of input signal 8?		
<b>FMode1</b>	Output last signal or <b>FState</b> in case of communication failure?	no → no particular reaction	no
<b>FMode8</b>		/	
<b>FState1</b>		FStat value output	0
<b>FState8</b>	Output status in case of error	0/1	



Note related to hardware type RM 251

The outputs are monitored pairwise. To avoid faulty displays, unused outputs should be short circuited.

## 3.11.5. RM\_AI (RM200 - analog input module (No. 17))



Function **RM\_AI** handles the data from connected analog input modules.

## Inputs and outputs

## Analog input

**Slotx** Connection of one of the slot outputs of the RM200 node (C\_RM2x).

## Digital outputs

<b>et-err</b>	0 = no engineering error detected 1 = engineering error (several RM module functions at a slot)
<b>slotid</b>	0 = correct slot assignment 1 = faulty slot assignment (faulty RM module fitted)
<b>valid</b>	0 = no data 1 = data could be received
<b>fail 1...fail 4</b>	Measurement error at channel 1 to 4 (e.g. sensor break)
<b>tcfile</b>	Temperature compensation error

## Analog outputs

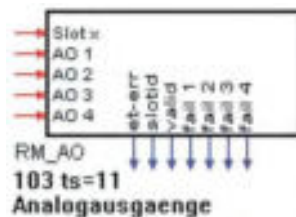
**Ai 1...Ai 4** 1st to 4th analog input signal

Configuration	Description	Range	Default
		0: RM221-0 = 4 x 0/4...20 mA 1: RM221-1 = 4 x -10/0...10 V 2: RM221-2 = 2 x 0/4...20 mA + 2 x -10/0...10 V 3: RM222-0 = 4 x 0/4...20 mA, TPS 4: RM222-1 = 4 x -10/0...10 V, Poti, TPS 5: RM222-2 = 2 x 0/4...20 mA + 2 x -10/0...10 V, Poti, TPS 6: RM224-1=4xTC/Pt100, 16 Bit 7: RM224-0 = 2 x TC, 16 Bit 8: RM224-2 = 1 x -3...3V, 1x TC, 16 Bit 1: Typ J = -120...1200°C 2: Typ K = -130...1370°C 3: Typ L = -120...900°C 4: Typ E = -130...1000°C 5: Typ T = -130...400°C 6: Typ S = 12...1760°C 7: Typ R = 13...1760°C 8: Typ B = 50...1820°C 9: Typ N = -109...1300°C 10: Typ W = 50...2300°C 30: Pt100 = -200...850°C 40: standard signal = 0 ... 10V 41: standard signal = -10...10V 50: standard signal = 4...20mA 51: standard signal = 0...20mA 0: unit = °C 1: unit = °F 0 2: unit = K	0
MType	Module type		
STyp 1...STyp 4	Input signal		51
Unit1...Unit4	Temperature unit input 1 to 4 (only relevant with thermo- couple and Pt100 inputs)		0
Tf 1...Tf 4	Filter time constant input 1 ... 4 in (s)	0 ... 999 999	0,5
x0 1...x0 4	Scaling start value input 1...input 4	-29 999 ... 999 999	0
x100 1...x100 4	Scaling end value input 1 ... input 4	-29 999 ... 999 999	100
Fai11...Fai14	Signal behaviour with sensor error at input 1...4	upscale downscale	←
X1in 1...4	Measured value correction input value Segment point 1 → input 1...4	-29 999 ... 999 999	0
X1out 1...4	Measured value correction output value Segment 1 → input 1...4	-29 999 ... 999 999	0
X2in 1...4	Measured value correction input value Segment point 2 → input 1...4	-29 999 ... 999 999	100
X2out 1...4	Measured value correction output value Segment point 2 → input 1...4	-29 999 ... 999 999	100

## Potentiometer connection and calibration

See chapter "calibration" → page 36

### 3.11.6. RM\_AO (RM200 - analog output module (No. 18))



Function RM\_AO handles the data from connected analog output modules.

#### Input and outputs

##### Analog inputs

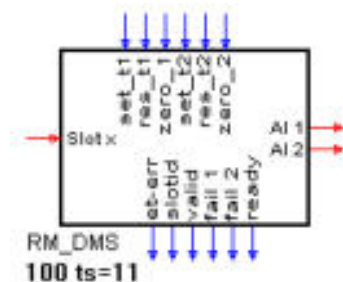
Slotx	Connection of one of the Slot outputs of the RM 200 node (C_RM2x).
AO 1...AO 4	1st to 4th analog output signal

##### Digitale Ausgänge

et-err	0 = no engineering error detected 1 = engineering error (several RM module functions on a slot)
slotid	0 = correct slot assignment 1 = faulty slot assignment (wrong RM module fitted)
valid	0 = no data 1 = data could be received
fail 1...fail 4	No calibration data / communication disturbed

#### Parameter and configuration data

Configuration	Description	Range	Default
MTyp	Module type	0: RM231-0 = 4 x 0/4...20 mA / 4 x 0...10 V 1: RM231-1 = 4 x 0/4...20 mA / 2 x 0...10 V / 2 x -10...10 V 2: RM231-2 = 4 x 0/4...20 mA / 4 x -10...10 V 10: standard signal = 0 ... 10V 11: standard signal = -10...10V 20: standard signal = 0 ... 20 mA 21: standard signal = 4 ... 20 mA	0
OTyp 1...OTyp 4	Output signal		
x0 1... x0 4	Scaling start value input 1...input 4	-29 999...999 999	0
x100 1...x100 4	Scaling end value input 1 ... input 4	-29 999...999 999	100

**3.11.7. RM\_DMS (strain gauge module (No. 22))**

Function RM\_DMS reads data from a special strain gauge module of KS98-1 I/O extension with CANopen. Max. 2 strain gauges can be connected to the module. The measured values are available at outputs AI 1 and AI 2.

The two measurements can be influenced via digital command inputs, e.g. zero setting. Monitoring a new command (positive flank at one of the digital inputs) is restarted only when the "ready" output is "1". The module position in the RM rack is determined by connection of analog input Slotx to the RM2xx node.

**! Important hint:**

A special coupler module (RM201-1) must be used for operation of the strain gauge module. This coupler module cannot be combined with thermocouple modules. Moreover, the limitations as for coupler module RM201 (e.g. max. 4 analog input modules) are applicable.

**Digital inputs:**

<b>set_t1</b>	Set tare strain gauge channel 1. The actual weight is not stored continuously as tare (packaging weight). The following measurements provide the net weight.
<b>res_t1</b>	Reset tare strain gauge channel 1. The tare value is set to 0. Gross weight= net weight.
<b>zero_1</b>	Zero setting of strain gauge channel 1 measured value. The actual measured value is stored as a zero value in a non-volatile memory.
<b>set_t2</b>	Set tare strain gauge channel 2. The actual weight is buffered as tare (packaging weight). The following measurements provide net weight.
<b>res_t2</b>	Reset tare strain gauge channel 2. The tare value is set to 0. Gross weight=net weight.
<b>zero_2</b>	Zero setting of the strain gauge channel 2 measured value. The actual measured value is stored as zero in the non-volatile memory.

**Digital outputs**

<b>et-err</b>	0 = no engineering error 1 = engineering error (several module blocks at a slot output). Slots not connected
<b>slotid</b>	0 = correct slot allocation 1 = faulty slot allocation (module type). Faulty coupler module
<b>valid</b>	0 = no data 1 = data could not be received
<b>fail 1</b>	faulty connection or measurement error on channel 1
<b>fail 2</b>	faulty connection or measurement error on channel 2
<b>ready</b>	ready message after command handling

**Analog inputs**

<b>Slotx</b>	connection of one of the slot outputs of the RM201-1-node block.
--------------	--

**Analog outputs:**

<b>AI 1</b>	1st measured value of strain gauge channel 1
<b>AI 2</b>	2nd measured value of strain gauge channel 2

**Parameter and configuration data**

Parameters:	Description	Range	Default
MTyp 1/2	module type	0: RM225 = strain gauge	
STyp 1/2	Input signal	0: -4 +4mV/V	
Unit 1/2	Unit of the input signal	mV/V	mV/V
Tf 1/2	filter time constant input 1 ... 2 (s)	0 ... 999 999 (0,5 )	0,5
x0 1/2	scaling start value input 1 ... 2	-29 999 ... 999 999	0
x100 1/2	scaling end value input 1 ... 2	-29 999 ... 999 999	(100)
Fail 1/2	signal action in case of sensor error	0:Upscale 1:Downscale	←
X1in 1/2	measured value correction input value segment point 1 > input 1...2	- 29 999...999 999	0
X1out 1/2	measured value correction output value segment point 1 > input 1...2	-29 999...999 999	0
X2in 1/2	measured value correction input value segment point 2 > input 1...2	-29 999...999 999	(100)
X2out 1/2	measured value correction output value segment point 2 > input 1...2	- 29 999...999 999	(100)

### 3.12. KS 98-1- KS 98-1 cross communication (CANopen)

While the data exchange between KS 98-1 and RM200, KS800 or KS816 must be done exclusively via the KS 98-1 to operating version 7 as master, the "cross communication" is directly possible.

#### KS 98-1 RM:

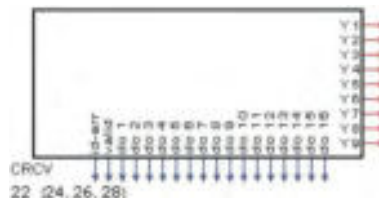
To every KS 98-1, even a slave, one or more RM nodes can be associated. But he can only access his own I/O. Data exchange between several KS 98-1 of a CAN network is via send modules (CSEND; block numbers 21, 23, 25, 27) and receive modules (CRCV; block numbers 22, 24, 26, 28).

Max. 9 analog values and 16 digital statuses from the relevant engineering can be transmitted persend/receive module. The sender sends the data together with its node address and block number. The receiver checks, if the messages correspond with the adjusted send address, and if the sender block number is by "1" lower than its own one

For BUS terminating resistor, see page: 159



#### 3.12.1. CRCV (receive mod. block no's 22,24,26,28 (No.56)



Function CRCV can receive data from a different KS98-1. The data of the other multifunction unit are made available by means of the CSEND function. Hereby, the CSEND block number is by 1 lower than the CRCV block number.

Der CRCV Nr. 22 reads the data of another KS98-1 from CSEND Nr. 21

Der CRCV Nr. 24 reads the data of another KS98-1 from CSEND Nr. 23

Der CRCV Nr. 26 reads the data of another KS98-1 from CSEND Nr. 25

Der CRCV Nr. 28 reads the data of another KS98-1 from CSEND Nr. 27

### Outputs

#### Analog outputs

Y1 . . . Y9 Analog output values 1 to 9

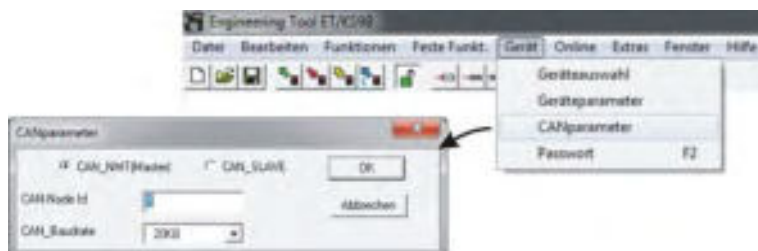
#### Digital outputs

id-err	0 = correct node Id 1 = faulty node Id
valid	0 = no data 1 = data could be received
do 1...do 16	Status values 1 to 16

## Parameter and configuration data

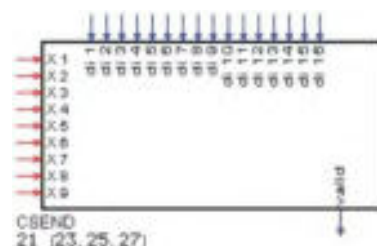
Parameter:	Description	Range	Default
<b>NodeId</b>	Node address of the sending KS98-1 (The sending KS98-1 is adjusted accordingly in engineering tool window "CANparameter ".) → see *1)		

\*1) The node address of the sending KS98-1 is adjustable in engineering tool window "CANparameter" or via the instrument parameters on the front panel (during off-line mode).



PSMA KS98-2	14:15
<b>Gerätedaten</b>	
Frequ. =	50 Hz
Sprach =	deutsch
CAN-Id =	(NMT) 1
CAN-Bd =	20kBit
Freeze =	aus
Delay =	0
Ethernet	

### 3.12.2. CSEND (Send mod. blockno.'s 21, 23, 25, 27 - (No. 57))



Function CSEND provides data for other KS98-1 units on the CANopen bus. The data can be read by the other multifunction units using the CRCVfunction.

## Inputs and outputs

### Analog inputs

X1...X9 Analog values 1 to 9, which are sent.

### Digital inputs

di1...di9 Digital values 1 to 16, which are sent.

### Digital output

valid 0 = invalid data (e.g. no KS98-1 but only KS 98-1)  
1 = data could be received

## Parameter and configuration data

Configuration	Description	Range	Default
delta	Change from which a new send operation is started.	0,000...999 999	0,1

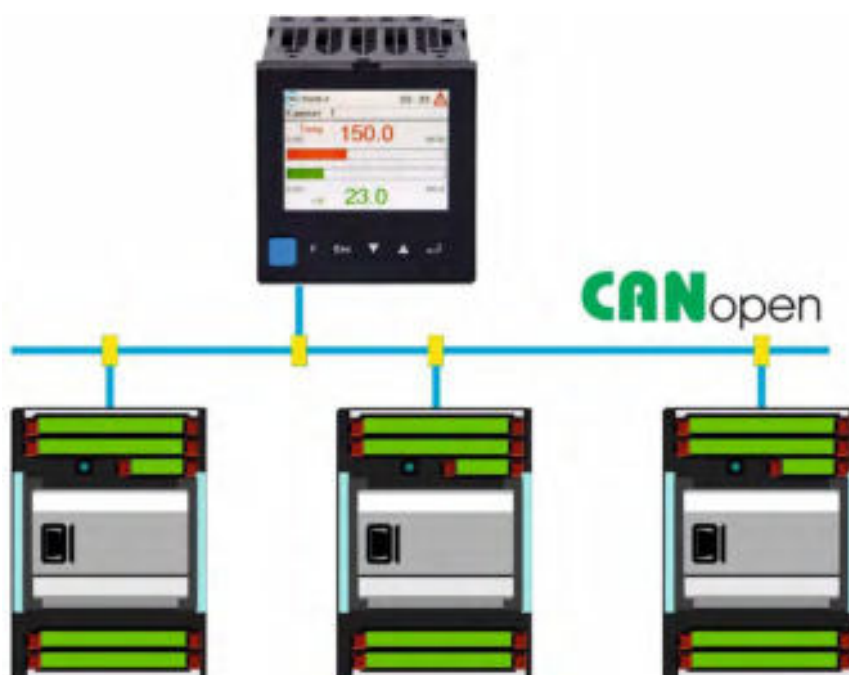


Transmission is at intervals of 200ms.

Note that there is a risk of data loss for values which are available only during 100 ms.



### 3.13. Connection of KS 800 and KS 816



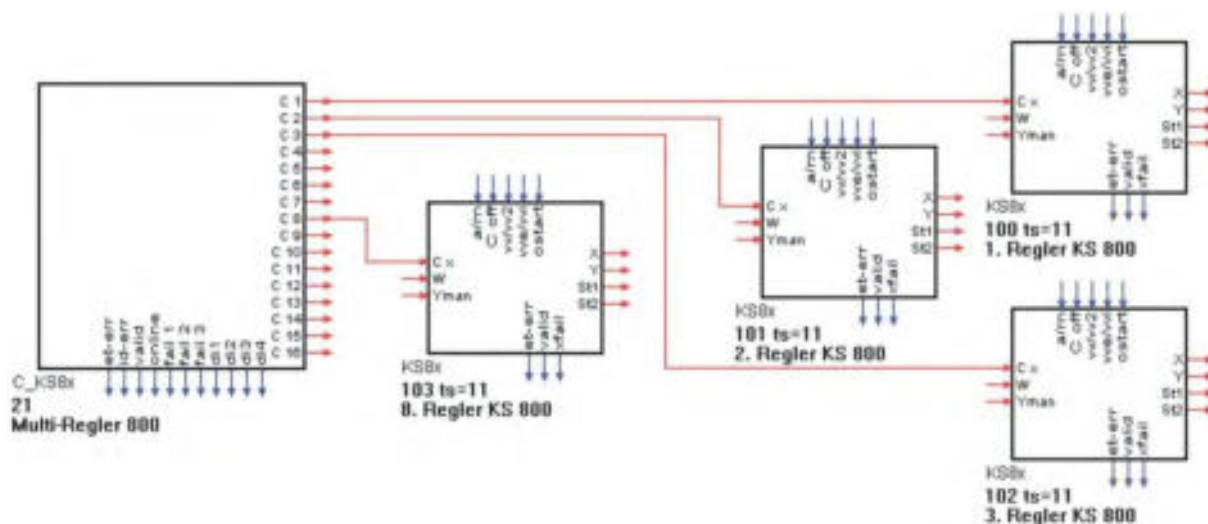
Function blocks C\_KS8x and KS8x can be used for communication of multifunction unit KS98-1 and multi-channel temperature controllers KS 800 and KS 816.

A node function C\_KS8x is allocated to each KS 800 or KS 816.

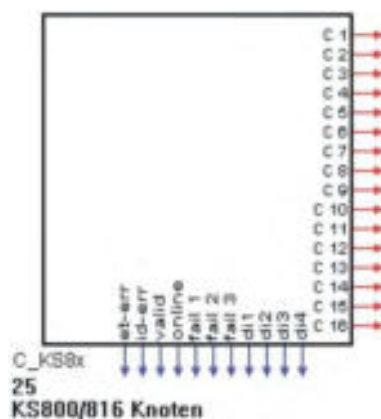
The KS8x functions are allocated to the various controllers of KS 800 (up to 8 controllers) or KS 816 (up to 16 controllers).

For BUS terminating resistor, see page: 159

*Partial engineering for communication with multiple channel temperature controllers KS800 and KS816*



### 3.13.1. C\_KS8x (KS 800 and KS 816 node function - (No. 58))



Node function C\_KS8x provides the interface to one of the multi-channel temperature controllers KS 800 or KS 816. Analog outputs C1 ... C16 can be used to connect the KS8x functions, which represent each a controller of KS 800 (max. 8 controllers) or of KS 816 (max. 16 controllers).

Unlike the other KS 98-1 functions, only one data function can be soft-wired to each analog output. Prerequisite for communication of KS98-1 multi-function unit and KS800 or KS816 is the complying adjustment of the CAN parameters. (→ see \*1).

#### Outputs

##### Analog inputs


C1...C16 Connection of the KS8x functions (single controllers in KS800 / KS816)

##### Digital outputs

et-err	0 = no engineering error 1 = engineering error (different node function at the same KS800)
Id-err	0 = correct node 1 = faulty node Id (no KS800 / KS816 replied under the configured node ID)
valid	0 = no data 1 = data were received
online	0 = KS800/816 is off-line 1 = KS800/816 is on-line
fail 1	0 = no fail at do1...do12 1 = fail at do1...do12
fail 2	0 = no fail at do13...do16 1 = fail at do13...do16
fail 3	0 = no heating current short circuit 1 = heating current short circuit
di1	di1 status
di2	di2 status
di3	di3 status
di4	di4 status

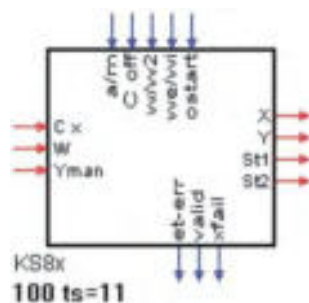
#### Parameter and configuration data

Configuration	Description	Range	Default
NodeId	KS800/KS816 node address	2...42	2

 The data from the various controllers are read cyclically. Maximally at intervals of 1.6 seconds (KS800) or 3.2 seconds (KS816), all data are updated.

\* 1) The parameters for the CANopen bus are adjustable in engineering tool window "CANparameter" or via the instrument parameters on the front panel (ET98 r Device r CANparameter).

### 3.13.2. KS8x (KS 800/ KS 816 controller function - (No. 59))



Each KS8x function handles a controller of KS 800 or KS 816. The analog and digital inputs can be used to send the control signals to the controller in KS800/16. The analog outputs provide the process and controller values.

#### Inputs and outputs

##### Analog inputs

C x	Connection to one of the C1...C16 outputs of node function C_KS8x
W	Controller
Yman	Correcting variable in manual mode

##### Digital inputs

a/m	0 = controller is in automatic mode 1 = controller is in manual mode
C off	0 = controller is switched on 1 = controller is switched off
w/w2	0 = controller is in automatic mode 1 = 2nd is active (safety)
we/wi	0 = external is active 1 = internal is active
ostart	0 = don't start self-tuning 1 = start self-tuning

##### Digital outputs

et-err	0 = no engineering error 1 = engineering error (several KS8x controller functions on a controller channel)
valid	0 = no data 1 = data were received
xfail	0 = no sensor fail 1 = sensor fail

##### Analog outputs

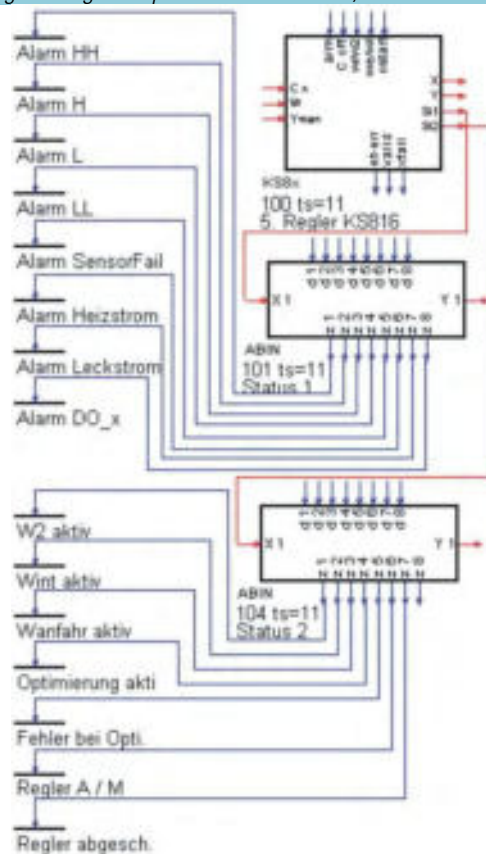
X	Controller process value
Y	Controller correcting variable
St1	Statusbyte 1
St2	Statusbyte 2

For an engineering example to evaluate St1 and St2, see the next page.

Engineering example for evaluationSt1/St2

St1 Statusbyte 1 (Bit)	Bit value	Description
0	1	HH Alarm
1	2	H Alarm
2	4	L Alarm
3	8	LL Alarm
4	16	sensor fail alarm
5	32	heating current alarm
6	64	leakage current alarm
7	128	alarm DOx

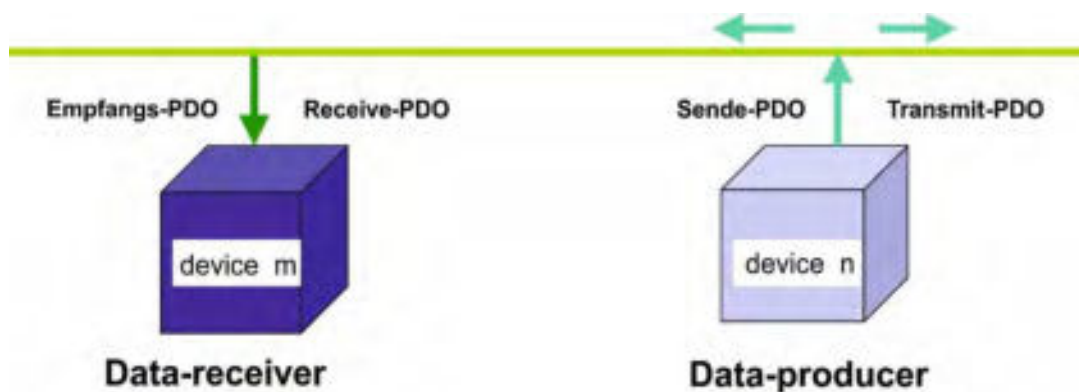
St2 Statusbyte 2 (Bit)	Bit value	Description
0	1	W2 aktiv
1	2	Wint aktiv
2	4	Wanfah aktiv
3	8	self-tuning active
4	16	self-tuning error
5	32	controller A / M
6	64	controller switched off
7	128	---



### 3.14. Description of KS 98-1 CAN bus extension

There are various modes for KS 98-1 communication via the CAN bus. The unit can be master for handling the NMT services (NMT = Network Management), or slave, can send or receive PDOs (PDO = process data object) cyclically or send SDO telegrams asynchronously (SDO = service data object). A KS98-1 can contact any bus parties simultaneously with other KS98-1, allocated remote IOs, KS800 multi-controllers and up to 40 sensors or actuators, and via asynchronous telegrams. Max. 42 CAN nodes can be addressed.

KS 98-1 handles guarding tasks as a master or a slave with an own local RM node. Display is in the CAN status window. However, there are limits to the performance of the bus parties and the bus itself. The dynamic operations of the bus can be evaluated only by statistics. The resulting bus and interface load of an instrument is dependent on the details of the communication structure and can be estimated only, if the behaviour of the individual parties is known exactly. In the following, properties and effects of various bus parties are explained and figures and facts are presented. Information on the COB-IDs consumed internally at PMA is given in the annex. This information should be taken into account when adding instruments from other manufacturers.

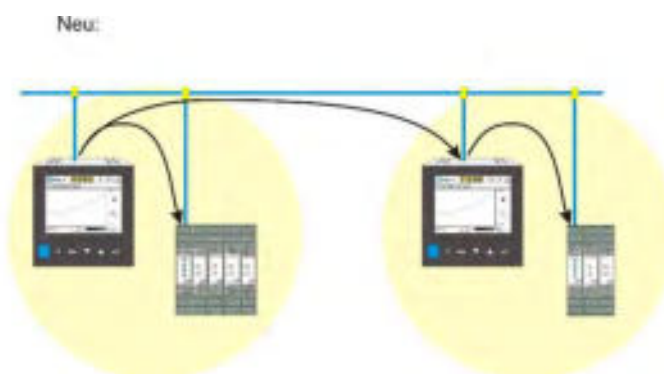
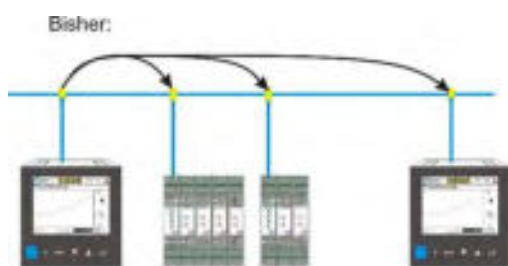


#### Process Data Objects (PDO)

- Process data for fast exchange
- one party sending = all parties can read (Producer / consumer concept)
- max. 8 bytes for data / message
- non-confirmed message
- synchronous or event-triggered
- Priority control via address
- Instrument-specific data contents

#### Service Data Objects (SDO)

- for data without real-time requirement
- asynchronous, confirmed messages
- Repartition over several telegrams is possible
- Addressing the data via indexes in the object directory (index, subindex).



The participants in the bus and also the bus itself have performance limits. About the dynamic processes on the bus can only make statistical statements. The resulting bus and interface load on a device will depend on the details of the communication structures and can only be estimated with accurate knowledge of the behaviors of the individual participants. In the following, characteristics and influences of different bus participants are explained and facts and figures compiled. The appendix provides information about the COD ID consumed internally by PMA. These must be taken into account when adding third-party devices.

### **KS98-1 CAN communication features**

Every message on the bus activates the KS 98-1 interrupt handler and loads the processor. The message is analyzed and queued, if the destination of the message is the own address. This queue is handled in the idle task and during the cyclical system processing phase (at intervals of 100ms).

70% of the CPU capacity is reserved for the engineering. This time is considered as 100 % in the KS98 ET timing dialogue. I.e. Min. 30ms are available for general tasks and communication. Included are front and rear instrument interface processing and Profibus handling. However, these loads are insignificant, because, for example, front and rear interface can only receive one telegram per 100 ms. This means that the CAN communication causes the largest part of the CPU load.

The PDO handling program is activated, as soon as the processing phase for the engineering within a cycle is finished (idle-task). Hence, more than 30 % of the processor capacity may be available for CAN communication with small engineering. The user can decide freely and at his own responsibility how to use these reserves can.

### **Receive PDO's**

The interrupt handler requires approx. 0,16ms for each PDO.

The event queues comprise 4 x 80 items. There is a queue for all send messages, another one for all PDO receive messages, still another one for the network receive messages and still another one for the SDO receive messages.

The queues are handled at intervals of 100 ms and during the idle task.

This means that no more than 80 PDOs per 100ms may be received.

The PDO handling is a processor load of approx. 1,2 ms for each individual PDO.

Blockwise handling of 50 receive PDOs takes KS 98-1 18 ms (19 ms, if the same number of PDOs for other receivers are rejected)

Although the load of the basic communication blocks (C\_RM2X, CPREAD, ...) cannot be allocated to a time slot, it is assigned as a fixed value to the engineering portion automatically.

### **Send-PDO's**

The load for transmitted PDOs is nearly the same as for receive PDOs (18ms / 50 PDOs), however, sending is not cyclical.

PDOs are sent only, when a value has changed (threshold adjustable with CSEND, otherwise, there will be a change of accuracy of the transmitted data format). At the latest after 2 seconds, the values are sent again also if unchanged. This reduces the output load by an unpredictable percentage. A filter can be used to reduce the transmission frequency of instable data.

### **Estimation of CAN bus activities of various instruments**

For reducing the data traffic between PMA instruments, PDOs are transmitted only in case of data changes. The changes are read with the accuracy of the used data format (LSB).

### **KS800-communication**

Both synchronous and asynchronous communication are used for KS800 communication. By configuration, one PDO is defined as synchronous and one PDO is defined as asynchronous.

### **A Sync message is sent at intervals of 200ms.**

This is followed by reception of a PDO containing the data of one controller channel by each KS800/816 . I.e. refreshing of 8 channels takes 1,6 seconds.

The internal KS800/816 cycle for handling a controller channel is 63,5 ms. If a channel status or correcting variable change occurs during this cycle time, KS800/81 sends 1 PDO asynchronously.

### **RM 200**

Data transmission in both directions is asynchronous. Data are transmitted only if changed (only the related PDOs). Checking, if changes were made is dependent on the accuracy of the data format (LSB). In both directions, the min. refresh rate is 100 ms

Max. 5 PDOs + 1status PDO are sent by the RM node dependent on the number of modules in the nodes.

Max. 5 PDOs are sent to the RM node by KS 98-1

### KS98-1 cross communication

Data transmission is asynchronous. Data are transmitted only when changes occurred (only the related PDOs). The min. refresh rate is 200 ms.

Max. 5 PDOs are sent dependent on the quantity of data connected to CSEND.

Max. 5 PDOs are received by KS 98-1

### Instruments from other manufacturers

Instruments from other manufacturers - sensors / actuators – can be addressed via synchronous data communication (send and receive PDOs), or using asynchronous data communication via SDOs. For reduction of the bus activities, checking for data changes is done by the sending side.


PDO reception can be influenced only by increasing the "Inhibit time" on the sensor side, in order to prevent information from being sent more frequently than once per 100 ms (KS 98-1 calculation cycle). Received data bytes can be converted into the internal format flexibly using function block A2BYTE. The operating principle of the block for the sending side is equal.

The receive and send interfaces (CPREAD/CPWRIT) are handled at intervals of 100 ms.

Konfig. wählen >> <b>RM-PDO-Zuordnung</b> << Neue Konfiguration			
	PDO-Bezeichnung	Knoten-ID	COB-ID
	RM-TPDO1 DI 8*8Bit	1	385
	RM-TPDO3 Ainp2 Int16	1	427
	RM-TPDO5 Ainp4 Int16	1	469
	RM-RPDO1 DO 8*8Bit	1	513
	RM-RPDO3 Aout2 Int16	1	555
	RM-RPDO5 Aout4 Int16	1	598
	RM-TPDO2 Ainp1 Int16	1	641
	RM-TPDO4 Ainp3 Int16	1	683
	RM-TPDO6 Fehler16DO8TK16AI16AO	1	725
	RM-RPDO2 Aout1 Int16	1	769
	RM-RPDO4 Aout3 Int16	1	811

In block number range 21-40, max. 40 PDO addresses (COB-ID=Communication Object Identifier: basic address + node address) can be addressed.

The data definition according to DS301 V4.0 complies with the Intel notation. The heartbeat protocol, which is offered by some manufacturers, is not supported.

 Recommendation for safe operation:

Bus load limitation

≤ 100 telegrams / 100 ms

Baudrate ≤ 250 kBit/s = 250m distance

Limitation of PDOs handled in the unit ≤ 50 telegrams / 100 ms (send/receive)

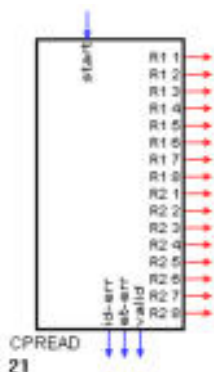
Send frequency for sensors > 100ms (inhibit time)

COB-ID allocation example for internal PMA CAN communication for node address 1:

Konfig. wählen >> - Querkommunikation << Neue Konfiguration		
	PDO-Bezeichnung	Knoten-ID COB-ID
	Quer-TPDO1 16Bits Zähler Analog1	1 385
	Quer-TPDO3 Analog 2/3	1 427
	Quer-TPDO5 Analog 4/5	1 469
	Quer-TPDO2 Analog 6/7	1 641
	Quer-TPDO4 Analog 8/9	1 683
	Quer-TPDO6 16Bits Zähler Analog1	1 725

Konfig. wählen >> - KS800-Zuordnung << Neue Konfiguration		
	PDO-Bezeichnung	Knoten-ID COB-ID
	KS800-TPDO1 synchron(chh,st,stat,Y)	1 385
	KS800-RPDO1 asynchron(chh,Soll,Y,Upd)	1 513
	KS800-TPDO2 asynchron(chh,st,stat,Y)	1 641
	KS800-RPDO2 asynchron(chh,Soll,Y,Upd)	1 769



**3.14.1. CPREAD (CAN-PDO read function (No. 88))**

Function CPREAD is used for read access to instrument PDOs. Due to the normal quantity of min. 2 PDOs per instrument, the data quantity of 2 PDOs 2 with 2 COB-IDs was grouped in one block.

Node address and COB-ID (CAN-Object Identifier) parameter setting is in the block. Moreover, node guarding for monitoring the CAN communication to the specified node can be switched on.

Data provided by the instrument must be interpreted according to the instrument specification. Groups of 4 transmitted bytes can be converted into different data types.

For this purpose, a conversion function for converting and inverting 1 to 4 bytes into a parameterizable data type (see function A2BYTE) is available.

Examples:  $R1+R2 > \text{Int16}$  /  $R1+R2+R3+R4 > \text{Long}$

**⚠ Important note:** The heart beat protocol is not supported. If an instrument can be operated only via "heart beat", the guarding function must be switched off.

**Digital inputs:**

start      The function is active with the input not connected, or if start=1 is connected.

**Digital outputs:**

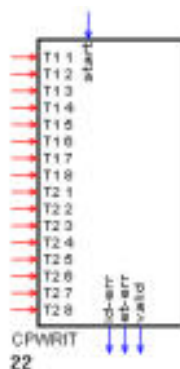
slotid	0 =	module correct
	1 =	module wrong
et-err	0 =	no engineering error
	1 =	no CAN-HW (KS 98-1 type) multiple node monitoring
id-err	0 =	correct node id
	1 =	faulty node id or instrument does not reply specify own node ID as "Nodeid"
		no free receive PDOs (RPDO)
valid	Bit follows node status with the node guarding active (0="preoperational", 1="operational") always 1 with node guarding switched off	

**Analog outputs:**

R1 1 ...R1 8	1st to 8th analog input value in byte format (8-bit) for COB-ID 1
R2 1 ...R2 8	1st to 8th analog input value in byte format (8-bit) for COB-ID 2

**Configuration parameters (can be changed only during OFFLINE):**

Nodeid	CAN node address
Guard	node guarding off/on
COBID1	decimal ID of the first CAN object identifier
COBID2	decimal ID of the second CAN object identifier

**3.14.2. CPWRIT (CAN-PDO write function (No. 89))**

Function CPWRITE is used for write access to instrument PDOs. Because of the normal quantity of min. 2 PDOs per instruments, the data quantity of 2 PDOs 2 with 2 COB-IDs was grouped in a block.

Node address and COB-ID (CAN-OBject IDentifier) parameter setting is in the block. Moreover, node guarding for monitoring the CAN communication to the specified node can be switched on.

Data sent to the instrument must be interpreted according to instrument specification. Groups of 4 transmitted bytes represent different data types.

To provide the bytes according to the required data type, a conversion function for transforming the value in the engineering into 1 to 4 bytes is available (see function A2BYTE).

Examples:  $R1+R2 > \text{Int16}$  /  $R1+R2+R3+R4 > \text{Long}$

**⚠ Important note:** The heart beat protocol is not supported. If an instrument can be operated only via "heart beat", the guarding function must be switched off.

**Digital inputs:**

**start** The function is active, unless the input is connected, or if start=1 is connected.

**Digital outputs:**

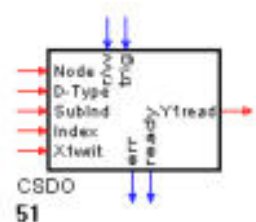
<b>slotid</b>	0 =	module correct
	1 =	module wrong
<b>et-err</b>	0 =	no engineering error
	1 =	no CAN-HW (KS 98-1 type) multiple node monitoring
	0 =	correct node id
<b>id-err</b>	1 =	faulty node id or the instrument does not reply own node ID was specified as "Nodeid" no free send PDOs (TPDO)
<b>valid</b>	bit follows the node status with the node guarding active (0="preoperational", 1="operational") always 1 with the node guarding switched off	

**Analog outputs:**

<b>R1</b>	1...R1	8	1st to 8th output value in byte format (8-bit) for COB-ID 1
<b>R2</b>	1...R2	8	1st to 8th analog output value in byte format (8-bit) for COB-ID 2

**Configuration parameters (can be changed only during OFFLINE):**

<b>NodeId</b>	CAN node address
<b>Guard</b>	node guarding off/on
<b>COBID1</b>	decimal ID of the first CAN object identifier
<b>COBID2</b>	decimal ID of the second CAN object identifier

**3.14.3. CSDO (CAN-SDO function (No. 92))**

Function CSDO permits access to the CAN bus by means of SDOs (Service Data Objects). SDOs are used for asynchronous data exchange without real-time inquiry.

Transmission started by the trigger input is always confirmed by the receiver, possibly during data inquiry along with value transmission. Reception of the confirmation is indicated by a logic 1 at the "ready" output. A new command can be generated via the positive flank at trig only with "1" indicated by the "ready" output.

Data required for command generation can be adjusted as parameters or connected as values to the inputs. As soon as a connection at an input was made, the relevant parameter loses its function. In this case, the value applied to the input is valid. Data (command) addressing in the connected instrument is done via indexes (index / sub-index), which is described in the CAN instrument documentation.

A value to be transmitted is connected to X1writ (or parameter "value"). A received value is output at Y1read. Y1read is set to 0 after power-on, after an error ( "err" = 1 ) and after a data output.

With RM modules provided in the KS 98-1 engineering, and for addressing the same nodes also via a CSDO block , the trigger should be interlocked with the valid bit of the RM-200 block. During access to RM nodes which are handled already by KS 98-1 in the background, there may be start-up collisions the consequences of which are removed only by restarting KS 98-1.

**⚠ Important note:** The heart beat protocol is not supported. If an instrument can be operated only via "heart beat", the guarding function must be switched off.

**Digital inputs:**

r / w      access mode : 0 = read, 1 = write

**Analog inputs:**

<b>Node</b>	decimal CAN-nodeaddress, 1..42 (KS 98-1 is the CAN Object Identifier according to CiA DS301, node ID + 600H) datatype of the connected value, 0..6. Following datatypes are available
	0: UInt8
	1: Int8
<b>D-Type</b>	2: UInt16
	3: Int16
	4: UInt32
	5: Int32
	6: Float
<b>SubInd</b>	address in object directory 1..255
<b>Index</b>	address in object directory 1..65535
<b>X1writ</b>	data value (-29999 ... 999999)

**Digital outputs:**

<b>err</b>	0 = no error 1 = error detected
<b>ready</b>	0 = transmission is being handled. So far, no confirmation was received. 1 = transmission completed. Ready for the next command.

### Analog outputs:

R1	1...R1	8	1. bis 8. analoger Eingangswert im Byteformat(8Bit) zur COB-ID 1
R2	1...R2	8	1. bis 8. analoger Eingangswert im Byteformat(8Bit) zur COB-ID 2

### Parameters (can be changed during operation):

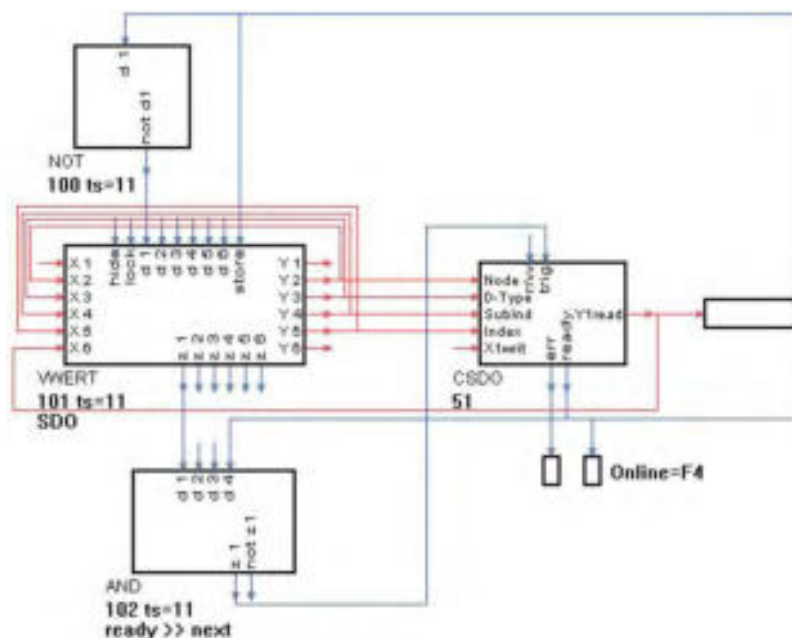
Access	Zugriffsart: 0 = lesen, 1 = schreiben
NodeID	dezimale CAN-Knotenadresse, 1..42 (KS 98-1+ bildet den CAN Object Identifier gemäß CiA DS301, Knoten ID + 600H) datatype of the connected value, 0..6. Following datatypes are available
D-Type	0: UInt8 1: Int8 2: UInt16 3: Int16 4: UInt32 5: Int32 6: Float
SubInd	address in object directory 1..255
Index	address in object directory 1..65535
Wert	data value (-29999 ... 999999)

### Possible errors (on output err):

- Faulty KS 98-1 hardware. KS98-1 CAN expected.
- The trigger input is not connected.
- No reply or faulty reply from the instrument.
- Instrument replies an inquiry with an error message.
- Min. one parameter or connected value is out of limits.

## SDO for data reading

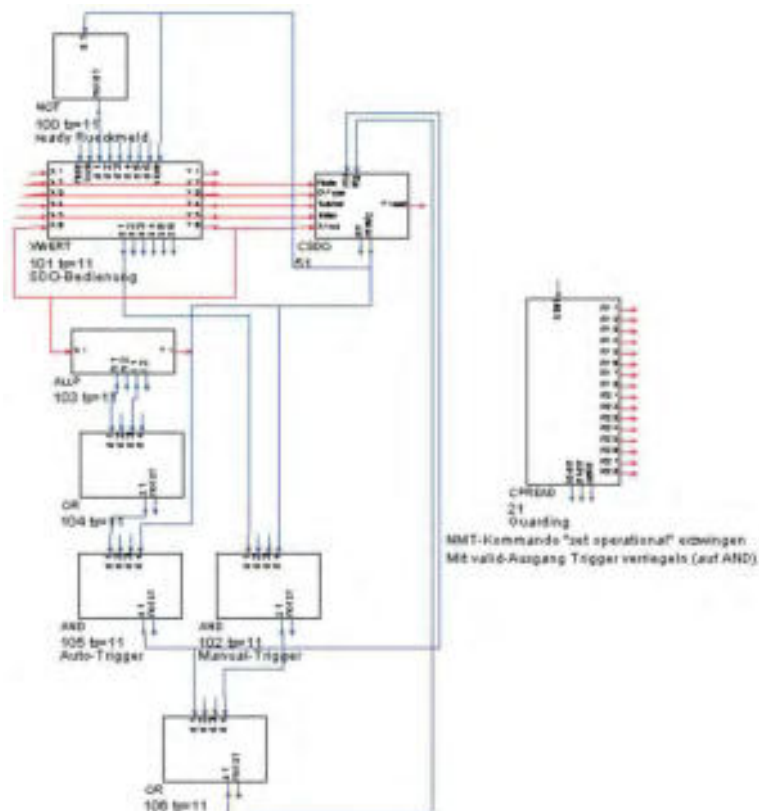
### Engineering examples



This example shows a possibility for data reading via an SDO access. Node address, data type, index and sub-index can be adjusted on an operating page. On the first line, a trigger bit which is reset by the following "ready" signal of the SDO block can be set. The engineering cannot be used to put a connected instrument into "operational" condition for PDO accesses. For this purpose, NMT commands must be used ( see the example given below ).

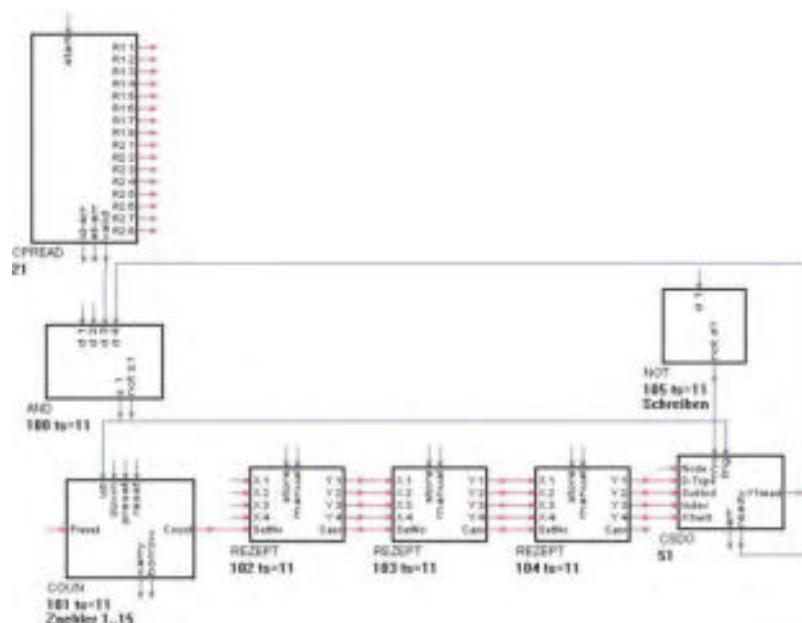
### SDO for data read/write with node guarding and set operational

In this engineering example for data write and read via SDOs, a trigger can be set automatically when changing a value to be transmitted, or manually via the first line of the operating page. Function block CPREAD, which is used normally for reading PDOs can be used to realize node guarding for an adjustable node. Moreover, this block ensures that the selected node is set "operational". In this case, connecting the "valid" output on the AND gates may be purposeful to prevent triggering as long as the connected instrument is not ready for addressing.

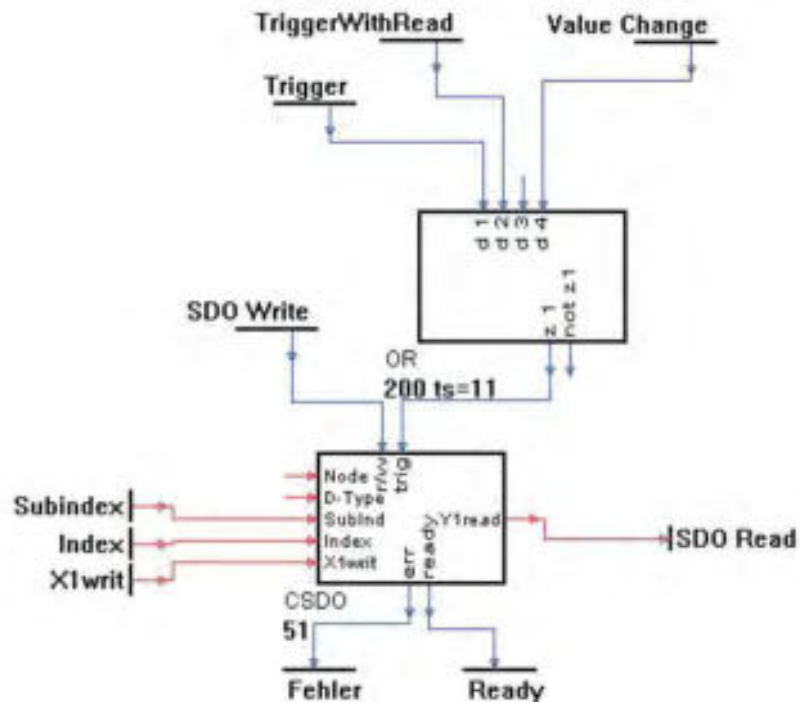


### Generating an SDO command sequence

Engineering example SDO-SEQ.EDG shows the generation of an endless SDO command sequence. The values for D-type, sub-index, index and value are stored in the recipe blocks. The counter ( COUN ) counts from 1 to 15 continuously.



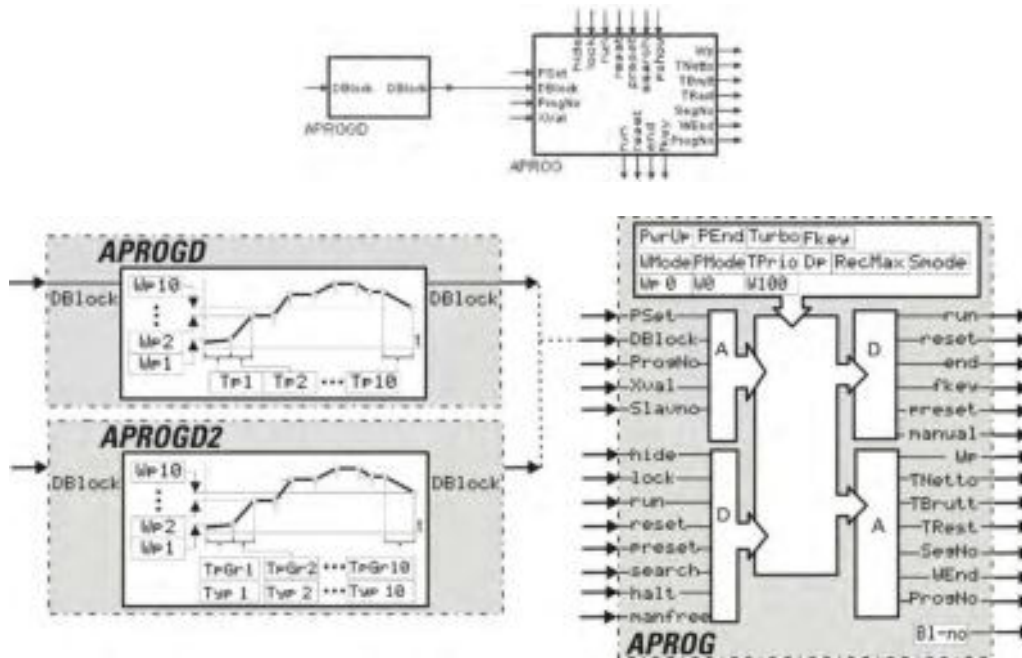




Command triggering is subject to various conditions: when reading, after changing during manual mode and cyclically in automatic mode.

### 3.15. Programmer

#### 3.15.1. APROG ( analog programmer (No. 24)) / APROGD ( APROG data (No. 25))



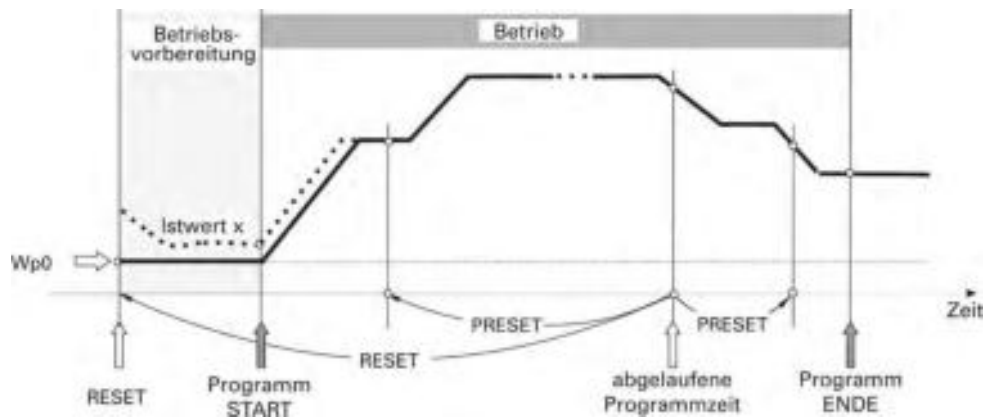
#### General

An analog programmer comprises a programmer (APROG) and min. one data block (APROD or APROGD2), whereby output DBlock of the APROGD/APROGD2 is connected with input DBlock of the APROG.

By connection of several of these cascable functions (each with 10 segments), a programmer with any number of recipes with any number of segments can be realized. APROGD and APROGD2 may not be mixed within a recipe. Limiting is only in the number of available block numbers and in the calculation time.

The data block (APROGD or APROGD2) has an analog output, at which the own block number is made available. This information is read-in by the programmer and used for segment data addressing. If an error with segment data addressing is detected, the reset value is output (status display on operating page: 'Error'). After an engineering download, Seg 0 is output (reset). If run is not connected, stop is used.







## APROG

### Digital inputs (APROG):

<b>hide</b>	Display suppression (with <b>hide</b> = 1 the page is not displayed in the operation).	
<b>lock</b>	Adjustment blocking (with <b>lock</b> = 1 the values are not adjustable by means of keys ID).	
<b>run</b>	Program Stop/Run ( 0 = Stop, 1 = Run )	
<b>reset</b>	Program Continue/Reset (0 = Continue, 1 = Reset )	reset has highest priority
<b>preset</b>	Program Preset ( 1 = Preset )	
<b>search</b>	Start programm search run (1 = search )	
<b>p-show</b>	Program editing enabled	
<b>halt</b>	Program interruption (e.g. due to an exceeded bandwidth detected outside the programmer).	
	0 = program not halted	
	1 = program halted	
	Manual mode disabled	
<b>manfree</b>	0 = switchover to manual mode not permitted	
	1 = switchover to manual mode is permitted	

### Digital outputs (APROG):

<b>run</b>	Status program stop/run (0 = program stop ; 1 = program run)
<b>reset</b>	Status program reset (1 = program reset)
<b>end</b>	Status program end (1 = program end reached)
<b>fkey</b>	Status  key / interface function 'fkey' (:pressing key  causes switch-over (0 or 1))
	This output indicates a programmer preset operation. With a single preset command, a pulse is output for the duration of one cycle (dependent on the time slot in which the programmer is classified). If the programmer is held in preset continuously, this output is always active.
<b>preset</b>	0 = no preset status
	1 = APROG stands in preset status
	This output indicates the programmer manual mode.
<b>manual</b>	0 = APROG works in automatic mode
	1 = APROG works in manual mode

### Analog inputs (APROG):

<b>PSet</b>	Preset value for program
<b>DBlock</b>	Block number of 1st data function 'APROGD'
<b>ProgNo</b>	Required program number (recipe)
<b>XVa1</b>	Value for search run
<b>SlavNo</b>	SlavNo: Block number of a connected slave block (for coupling master and slave blocks (APROG or DPROG))

## Programmer

### Analog outputs (APROG):

<b>Wp</b>	Programmer setpoint
<b>TNetto</b>	Net program time ( $\Sigma$ Trun)
<b>TBrutt</b>	Gross program time ( $\Sigma$ Trun + $\Sigma$ Tstop)
<b>TRest</b>	Programmer rest time
<b>SegNo</b>	Actual segment number
<b>WEnd</b>	End value of actual segment
<b>ProgNo</b>	Actual program number (recipe)
<b>SegRest</b>	Remaining segment time
<b>B1 - no</b>	Own block-number (e.g. for coupling master and slave blocks)

Parameter APROG	Description	Range	Default
<b>WMode</b>	Change mode:	Ramp Step	<b>Ramp</b> <b>Step</b> ←
<b>PMode</b>	Preset Mode:	Preset to segment Preset to time	<b>Pres.Seg.</b> <b>Pres.Zeit</b> ←
<b>TPrio</b>	Start mode in search run	Gradient has priority Segment/time has priority	<b>Grad.Prio</b> <b>Zeit Prio</b> ←
<b>Dp</b>	Decimals for setpoint	0..3	3
<b>RecMax</b>	Max.recipees	1..99	99
<b>Smode</b>	Smode (search mode):	0 = search run in segment 1 = search run in program/section 2 = no search run	
<b>Wp0</b>	Program at reset	W0..W100	W0
<b>W0</b>	Lower setpoint limit	-29 999 ... 999 999	-29 999
<b>W100</b>	Upper setpoint limit	-29 999 ... 999 999	999 999

## APROGD

### Analog inputs (APROGD):

<b>DBlock</b>	Block number of cascaded data function 'APROGD'
---------------	---

### Analog outputs (APROGD):

<b>DBlock</b>	Own block number
---------------	------------------

Parameter APROGD	Description	Range ET	Gerät	Default ET	Gerät
<b>Tp 1</b>	Time for segment 1 1	0 ... 95 999	<b>0:00. . . 999:59</b>	AUS	--:--
<b>Wp 1</b>	Segment end in segment 1	-29 999 ... 999 999		0	0
<b>Tp 2</b>	Time for segment 2	0 ... 95 999	<b>0:00. . . 999:59</b>	AUS	--:--
<b>Wp 2</b>	Segment end in segment 2	-29 999 ... 999 999		0	0
<b>...</b>					
<b>Tp 10</b>	Time for segment 10	0 ... 95 999	<b>0:00. . . 999:59</b>	AUS	--:--
<b>Wp 10</b>	Segment end in segment 10	-29 999 ... 999 999		0	0

Enter the time for a segment in seconds or minutes into the engineering tool dependent of configuration (**Turbo**). Entry into the unit is in Hrs:Min or Min:Sec. In addition to the range, a switch-off value can be entered (ET: OFF/-32000; unit: --:--). When reaching a segment with a switch-off value, 'End' is output

**APROGD2**

## Analog inputs (APROGD2):

**DBlock** Block number of cascaded data function `APROGD`

## Analog outputs (APROGD2):

**DBlock** Own block number

Parameter	Description	Value Range	Unit	Default	
APROGD2		ET		ET	Unit
		0	Time segment	←	←
		1	Gradient segment		
		2	Dwell segment		
		3	Step segment		
<b>Type 1</b>	(Type for segment 1)	4	Time segment and wait at the end		
		5	Gradient segment and wait at the end		
		6	Hold segment and wait at the end		
		7	Step change segment and wait at the end		
...					
<b>Typ10</b>	Type for segment 10	like type 1 0...7	...		
<b>TpGr1</b>	Time or gradient for segment 1	0 ... 59,999	0 ... 999:59	OFF	—:—
<b>Wp1</b>	Final value for segment 1	-29999 ... 999999	-29999 ... 999999	0	0
...					
<b>TpGr10</b>	Time or gradient for segment 10	0...59,999	0 ... 999:59	OFF	—:—
<b>Wp10</b>	Final value for segment 10	-29999 ... 999999	-29999 ... 999999	0	0

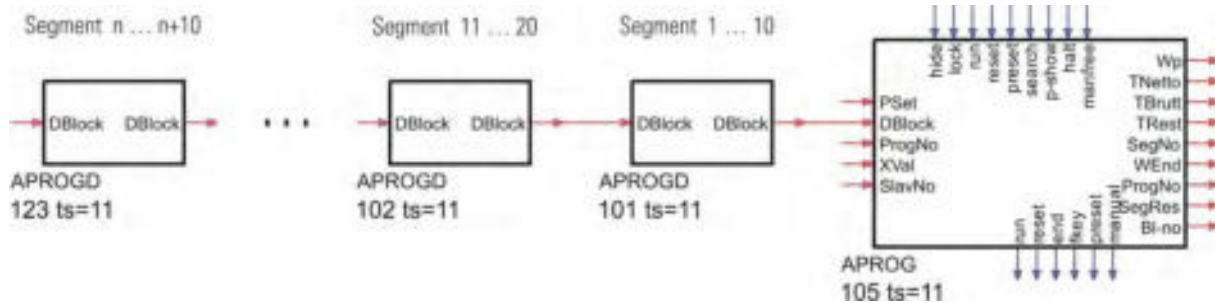
Configuration	Description	Value
APROG		
<b>PwrUp</b>	Behaviour after mains recovery	Continue program (default) Search run in the actual segment Continue at actual time
	Behaviour at program end PEnd:	stop after program end (default) Reset after program end
<b>PEnd</b>	0 = Stop 1 = Reset 2 = Reset + Stop (End-condition is reset with stop)	<b>Prog.Fort</b> <b>Fort.Seg.</b> <b>Fort.Zeit</b> <b>Stop</b>
<b>Turbo</b>	Time unit	Time = hours : minutes (default) Time = minutes : seconds
<b>FKey</b>		FKey (key function): 0 = fkey output status switchover by key (previous function) 1 = key used to generate a pulse at the fkey output (pulse length = 1 cycle) 2 = key controls the programmer (fkey output generates a pulse when actuating the key, pulse length = 1 cycle)

## Cascading

Cascading APROGD/APROD2 function blocks permits realization of a programmer with any number of segments. The segment sequence is dependent of the APROGD/APROD2 function block wiring (→ see below). The block numbers are without signification related to the order.

Segment parameters from right to left in the data blocks

Example of an analog programmer with n segments



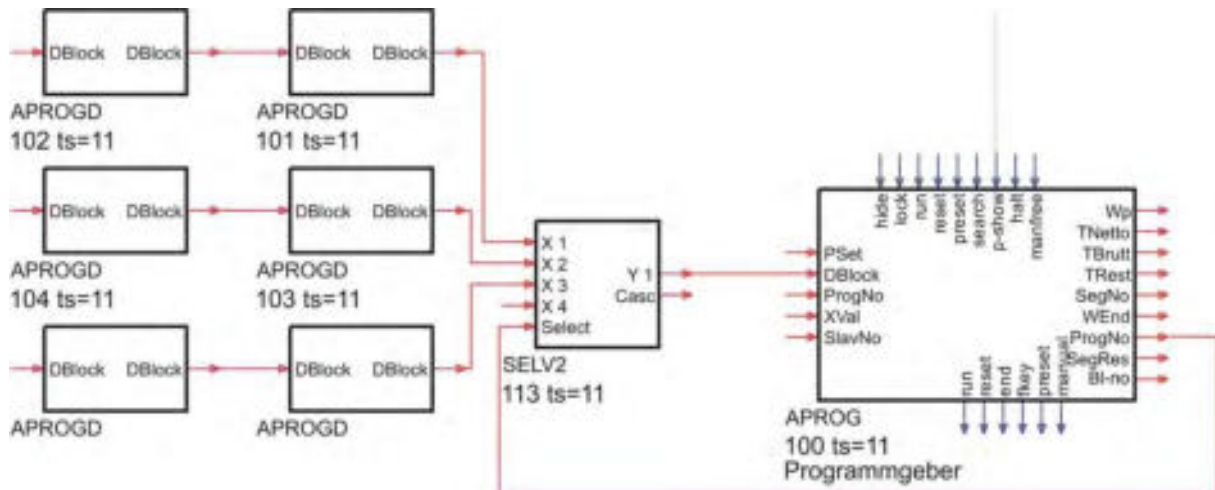
## Recipes


Analog output '**ProgNo**', at which the actual recipe number is output, and one or several SELV2 function blocks can be used to select a recipe the block number of which is switched to the APROG input (→ see below). Selection of the required recipe is possible via analog input '**ProgNo**' or recipe number, which can be entered via operation/interface

Recipe switch-over (new **ProgNo**) is effective only after programmer reset.

Entry of the recipe number via operation/interface is possible only, if analog input **ProgNo** is not connected.

Example of an analog programmer with 3 recipes à 20 segments



 Max. 800 ms after switching over, the block number of the first parameter block of a new recipe must be applied to the Dblock input. In a cascade, the SELV2 blocks must be arranged in ascending order.

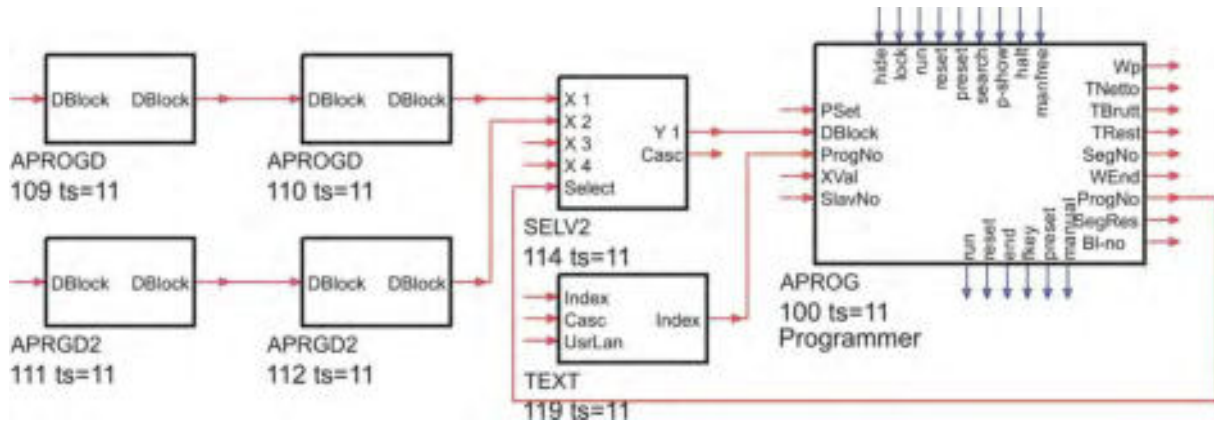
## Recipe changing - program selection

Whilst a program is active, switching over to a different recipe on the programmer operating page is not possible. Recipe changing is possible only in reset status!

## Recipe names

By linking TEXT blocks to the ProgNo input, display of recipe names rather than of recipe numbers is possible.

Recipe names



Each program starts at an initial position  $Wp_0$ , which is used and maintained with reset or first programmer set-up. With program start from rest position, the first programmer segment runs from the instantaneous process value at the time of start command („ramp“ with gradient  $(Wp_1 - Wp_0) / T_{p1}$ ). With step change mode, the of the first segment is activated immediately.

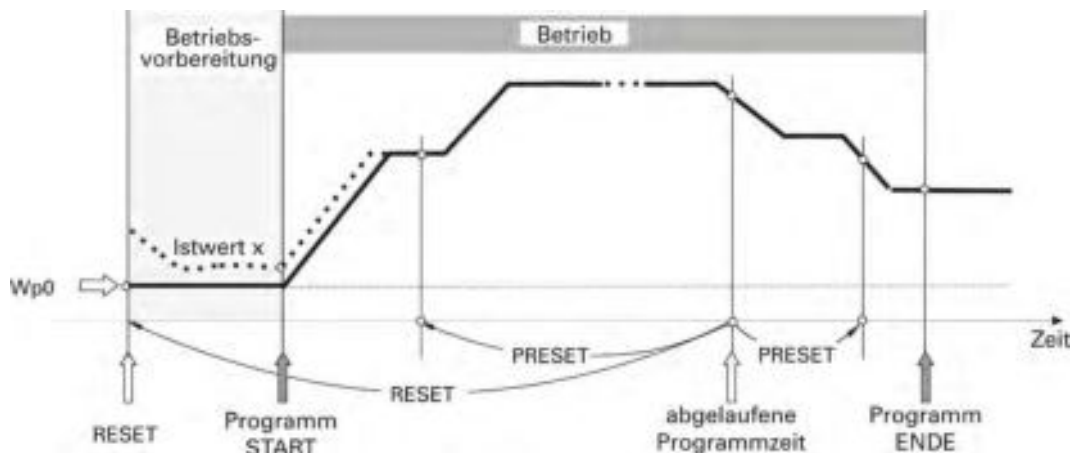
## Operation preparation and end position

Each program starts at an initial position  $Wp_0$ , which is used and maintained with reset or first programmer set-up. With program start from rest position, the first programmer segment runs from the instantaneous process value at the time of start command („ramp“ with gradient  $(Wp_1 - Wp_0) / T_{p1}$ ). With step change mode, the of the first segment is activated immediately.

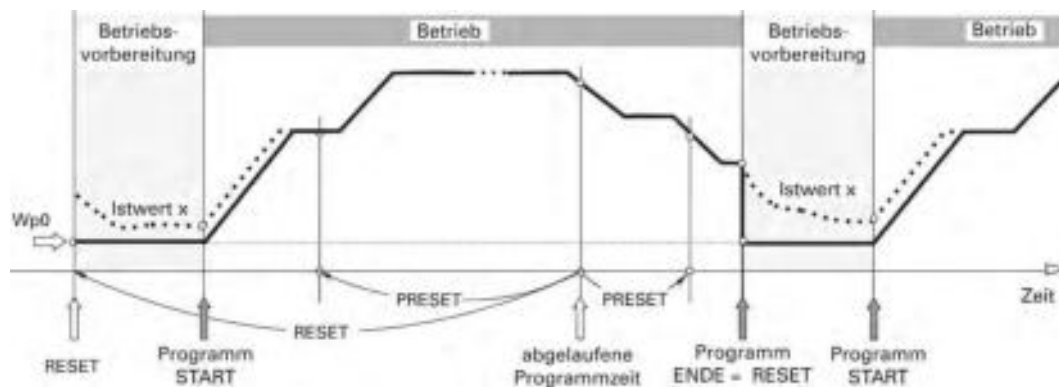
At program end, either

- 0 = Stop: the of the last segment is maintained (→ see below ),

[Profile with stop at end position](#)



- 1= reset: or the programmer goes to rest position Wp0 ( r see below ) and restarts automatically, if run is still valid.



- 2 = reset + stop: Programmer goes to rest condition Wp0.

At program end, the number of the last segment increased by 1 is output as active segment number (SegNo output of operating page and interface). This is required to bring the slave block to the end status safely with a segment preset.

## Start

The programmer uses a common start Wp0 for all programs. However, using an individual start for a recipe is possible as follows:

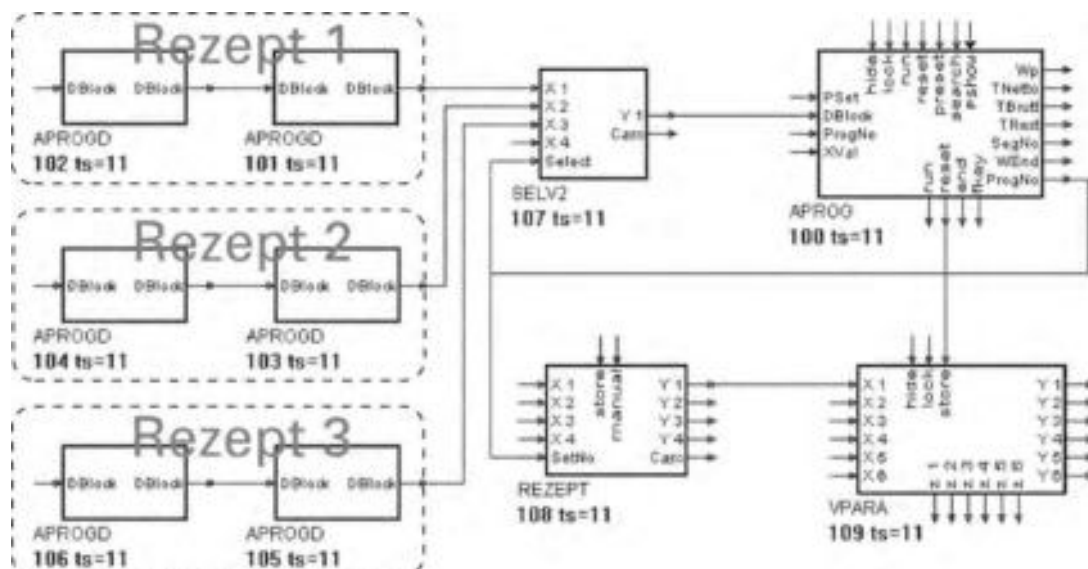
Use the setpoint of the 1st segment of each program as start value

Set the relevant segment time (Tp1) to 0.

Set the search run parameter Smode to 'Search run in program sections'. Now, the search run is not limited to the 1st segment and program start at the process value in the 2nd segment is possible (see search run, page → marking(SS)).

If each recipe shall have a separate reset (Wp0), function blocks REZEPT and VPARA can be used as shown up40. Note the calculation order (APROG → REZEPT → VPARA).

### Recipees with separate starting setpoints



### Halt status

Used e.g. for bandwidth monitoring

The halt status can be switched on and off only via the halt control input. Unlike the stop status, the run status remains unchanged (run output remains active) in the halt status.

Status display is "halt".

### Automatic/manual operation



The programmer can operate both in automatic or in manual mode:

automatic: The effective is determined by the programmer.

manual: The effective can be altered via programmer operating page or via interface. However, the program continues running and can be influenced via control inputs and operation/interface as during automatic mode (run/stop/reset/preset/search).

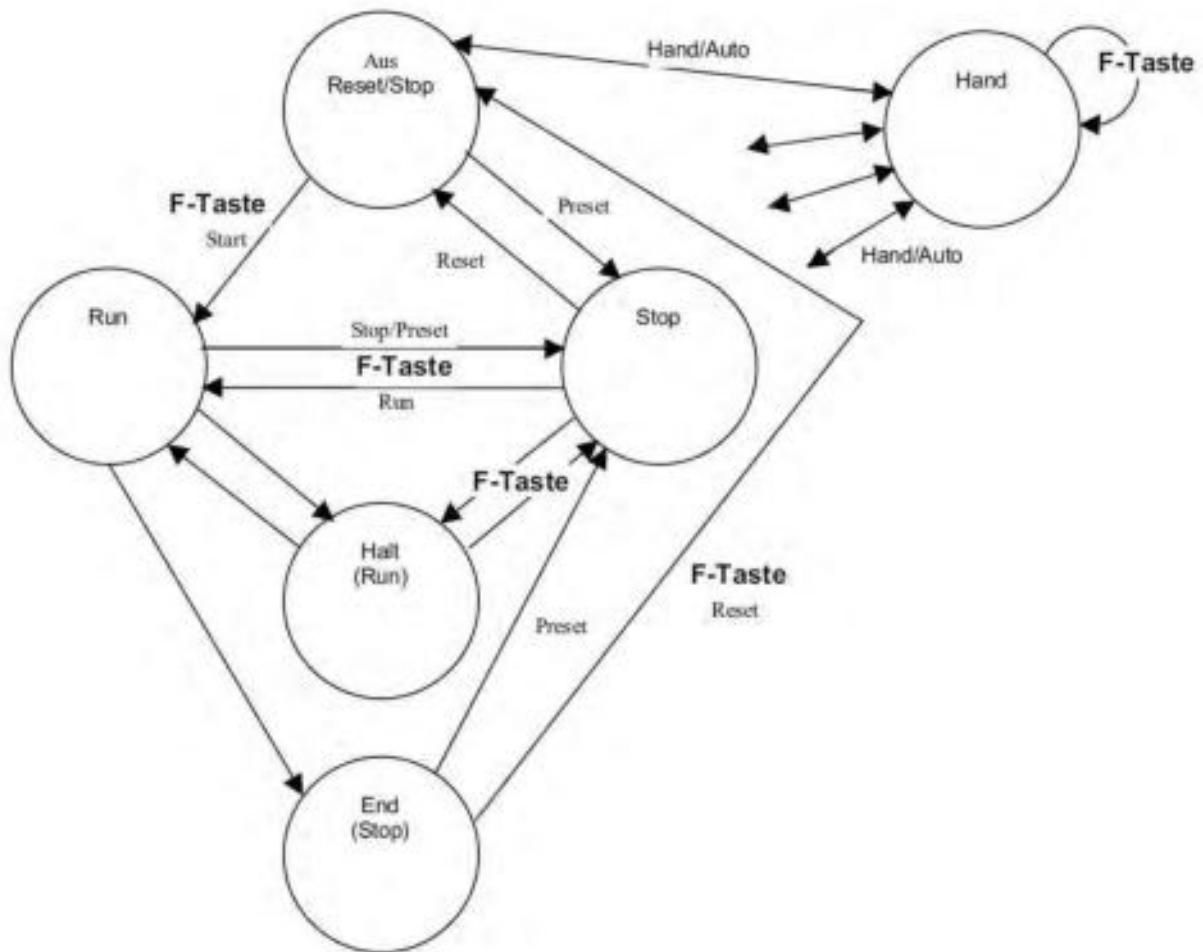
- When switching over from automatic & manual, the effective remains at the last value which was effective before switching over.
- Manual & automatic switchover: The effective changes from the manual to the actual programmer.
- Switching over can be done via the programmer operating page ("automatic" < > "manual") or via interface.
- The automatic/manual mode displayed only via the digital manual output.
  - 0 = auto,atic
  - 1 = manual
- The "manfree" control input can be used to enable switchover.
  - 0 = switchover to manual is disabled
  - 1 = switchover to manual is enabled

### Programmer control via key

Programmer control is possible by means of digital function block inputs, status changing on the operating page, interface and also using key . For selection of the  key functionality, a configuration parameter is offered

FKey: 0 = toggle bit changes at each key pressure at output fkey  
1 = F key function with pulse at output fkey  
2 = F key controls the programmer (fkey output generates a pulse when a key is actuated)

Hereby, the rule that the statuses at connected control inputs have priority over the operation is applicable. The following diagram describes the status sequence dependent on the actions:



### Change mode (ramp/step)

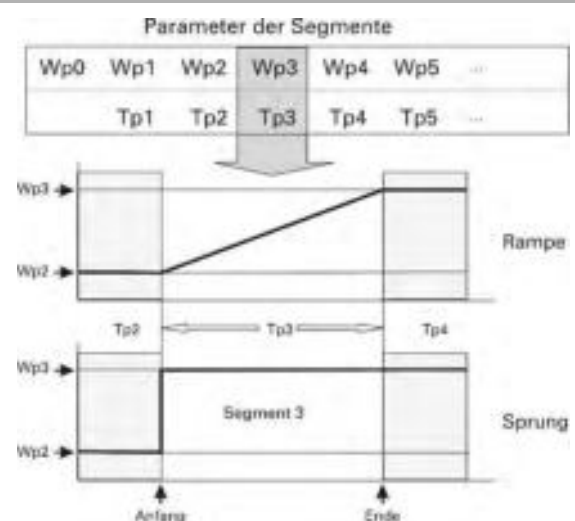
If the shall change in a step or ramp is determined by a parameter (**Wmode**) valid for all segments of a recipe (default: ramp).

Ramp: The changes linearly from the start (end value of previous segment) to the end value of the relevant segment in time **Tp**.

For the first segment, the following gradient is applicable:  

$$\frac{Wp1 - Wp0}{Tp1}$$

Step: The goes to value **Wp** immediately at segment start and maintains it during segment time **Tp**.





### Segment types

Different segment types for each individual segment can be determined separately in data block APRGD2. Like the APROGD, the APRGD2 block contains the parameter for 10 segments. Apart from parameters and time, APRGD2 has the segment type as a third parameter. I.e. the following basic segment types are possible within a program:

- Time segment with target and segment time
- Gradient segment with target and gradient
- Hold segment with hold time
- Step change segment with and segment time

Two variants of all segment types are available: with and without wait status at the segment end.

Segments with wait status feature particularities which must be taken into account. :

- A segment of this type does not limit the search run over several segments (see search run on page → Page 194).
- Behaviour after a short power failure ( $\leq 0.5$  Std.) with configuration PwrUp = 2 (continue at actual time): If the program time from power failure to power recovery includes min. one segment with wait status at the end, the search run in the segment in which the program would be without power failure is omitted and the program stops at the first wait status without search run.
- As with the APROGD block, the recipe end is determined by switching off a time parameter (TP=off) or by a DBLOCK input which is not connected any more.
- When connecting the APRGD2 block to the DBLOCK input of APROG, the new segment types are used automatically. The setting of parameter WMode is ignored when using the APRGD2 block.
- Mixing APROGD and APRGD2 blocks in a recipe is not permitted. However, a programmer can be operated with both data block types as long as only one parameter block type per recipe is used.

### Program sequence changes

During the running program, s and times (online) can be changed. Moreover, further segments, which did not exist so far, can be added. The actual segment number remains unchanged. Unless the actual segment is changed, the relative elapsed time also remains unchanged.

- ☐ Past changes  
A change of values and times of the past (already elapsed segments) are only effective after re-start (after previous reset).
- ☐ Future changes  
Changes of the future (segments which are not reached so far) are immediately effective. With changes of segment lines, the „rest time“ is re-calculated automatically.
- ☐ Present changes  
Changes of the actual segment time, which mean a return into the past (e.g. segment time Tp reduction to lower values than the relative time which has already elapsed in this segment) cause a branch to the start value of the next segment. Changes of the destination value of the actual segment cause unique re-calculation of the segment gradient for this program run, in order to reach the new destination value in the remaining time. Final re-calculation of the segment gradient is when starting a new batch (reset and start) or with preset to an earlier time.

## Search run

In the following cases, a search run is carried out:

- Start via operation
- Start via interface
- Start with **search** = 1
- Program start after **Reset**
- After short power failure with **PowerUp** = **Cont.seg.** or **Cont.time**

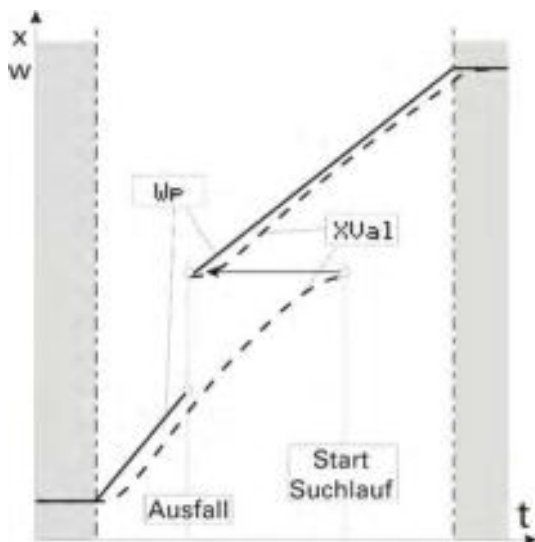
### Search run in program segment

When starting the search run, **Wp** is set to the **XVal** value, from where it runs towards the segment end value with actual gradient (**TPrio** = **Grad.prio**) or in the actual segment rest time (**TPrio** = **Time prio**)

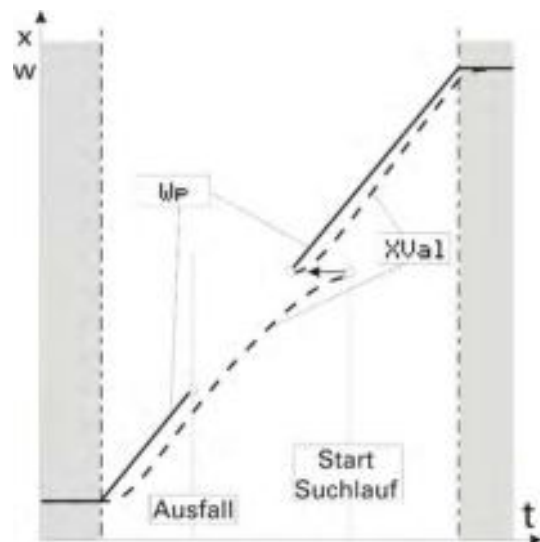
If the search value is out of the actual segment with **TPrio** = **Grad.prio**, the program is continued at the segment point next to the search value (actual segment start / end). With segment start value = segment end value (segment without gradient; holding time), the program is continued at the segment start.

With a step segment (including the APROGD segments with **WMode** = 1 (step)), the relevant target setpoint is always used from the segment start.

Search run for **TPrio** = **Zeit.Prio**



Search run for **TPrio** = **Grad.Prio**



### Search run in program section

Apart from the search run in the actual segment as described above, searching over several segments or search run switch-off are possible. The various search run functions can be selected via parameter **SMode**:


- 0 = search run in segment
- 1 = search run in program section
- 2 = no search run


Searching is limited to a section of several segments which have the same polarity sign as their gradient. Hereby, a hold segment is neutral  $\Rightarrow$  no change of polarity sign.


As the throughput times with a search run dependent on the number of segments may be very long, searching is limited to several time slots so that searching is done only in one segment per time slot.

### A search run is done in the following cases:

- Search run at program start: search over several segments up to the next gradient change
- search run started via control input, interface or via operation: forward and backward search from the actual program point to the next gradient change
- Search run after power failure at **PwrUp** = 1: forward and backward search from the point of failure to the next gradient change
- Search run after power failure at PwrUp = 2: forward and backward search from the program time in which the program would be without power failure until the next gradient change
- Search run in the hold segment (Gradient = 0): A search run is done only if there is min. one further segment (except hold segment) in this section. With a further hold segment directly before or behind this segment, the search run takes place only in the actual segment.

 Search run at TPrio = 1 (time priority): The search run is limited to the actual segment, i.e. the runs from the actual process value to the segment end value within the actual remaining segment time.

 Segments with wait status at the end do not limit the search range, except it's the search run after voltage failure!

 A search run may lead to program termination.

### Analog programmer operating page

Analog programmer APROG has an operating page, which can be selected in the operating page menu with input 'hide' not connected. If the FB inputs (function block inputs) allocated to the input fields in the following table are assigned to the engineering, operation (change) of this input field is not possible!

- |   |  |
|---|--|
| ① | Name of programmer block                                   |
| ② | Recipe name  |
| ③ | Process value  |
| ④ | Segment number   |
| ⑤ | Status (r/w) automatic/manual                              |
| ⑥ | Block change   |
| ⑦ | Setpoint   |
| ⑧ | Segment start and end value                                |
| ⑨ | Remaining segment time                                     |
| ⑩ | Program net time   |
| ⑪ | Remaining program time                                     |
| ⑫ | Status (r/w) stop, run reset, search, program, quit, error |
| ⑬ | Status (→) halt, end                                       |



Run, reset, preset and search are concerned, see following table:

Input field	Operation	Display	FB- input
Header	Selection of slave block	Display of slave data	- : -
<b>auto/manual</b>	Operating mode selection	<b>auto</b> or <b>manual</b>	- : -
<b>setpoint</b>	Automatic: programmer , Manual mode: operator setting in input field	Active setpoint	- : -
<b>Rec</b>	If input <b>ProgNo</b> is wired, entering of the desired recipe number is not possible over the frontside!	indicates the actual recipe number.	<b>ProgNo</b>
<b>Seg</b>	If control input preset is wired, entering the desired segmentnumber is not possible over the frontside!	indicates the actual segment number	<b>preset</b>
<b>tNetto</b>	Entry of required programmer time (preset to time)	indicates the <b>run</b> time total (without pause)	<b>preset</b>
<b>tRest</b>	No operation possible	indicates the time until program end	
<b>stop</b>	Stop the programmer	programmer stopped	<b>run</b>
<b>run</b>	Start the programmer	the programmer was started	
<b>Status</b>		the programmer is switched to segment 0 and ' <b>stop</b> '	<b>reset</b>
<b>reset</b>	The programmer is switched to segment 0 and ' <b>stop</b> '		
<b>quit</b>	Leave the field without change		
<b>program</b>	direct adjustment of segment parameters	segmentparameter	

### Notes related to the properties of the operating page

The display fields shown in bold and underlined letters on the picture above mark the elements which are switched over when changing to a slave page (see sections Master/slave operation on page 215).

The remaining fields continue indicating statuses and values of the master page.

### **Recipe name:**

- Recipes can be selected in the reset status. Unless a user text (TEXT block at ProgNo input) is provided, 'Rec n' is displayed (n stands for the current recipe number).
- The process value is visible only with the process value input connected.
- The segment number is adjustable only with preset to segment.
- The setpoint can be adjusted only during manual mode.
- Display of the remaining segment time is suppressed with preset to segment (e.g. with digital slave programmers).
- The program net time is adjustable with preset to time.
- 3 statuses are displayed (partly adjustable, dependent on operating status):
  - Status left: automatic / manual (adjustable)
  - Status middle: halt / end (unless one of the two statuses is active, this display is omitted)
  - Status right: stop / run / reset / search / program / quit / error

## Direct programmer adjustment

Program s and segment times can be adjusted at the operating page directly via the instrument front panel, without calling up the parameter level. Direct access to parameter setting is enabled with control input **p-show** = „1" set at the function blocks of programmer APROG and DPROG.

Menu item **Program** can be selected in the status line. After confirmation, all segment parameters **Tp** and **Wp** pertaining to an effective recipe **Rec** can be displayed and adjusted in a scroll window. Return to the normal operation is with **End**.

Scrolling is done over several data blocks (APROGD, APROGD2, DPROGD). „n" segment parameter (Wpn, Tn) indexing is with 3 digits. The segment parameters are distributed to the concerned data blocks automatically from left to right with ascending index.

If the last segment time Tn is adjusted to a valid value, the next parameter Tn+1 is displayed automatically. = --:-- etc.

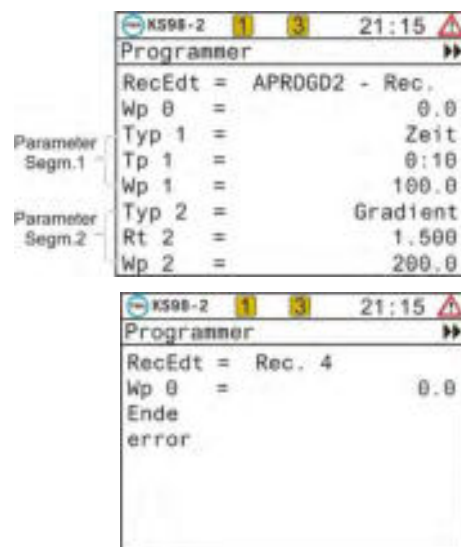
Thus an actual program can also be reduced by setting Tn+1. = --:-- in the required position. The following segments are suppressed in the program sequence. However, the relevant segment parameters remain unchanged and are made effective again by entry of a valid value at the relevant point.



The programmer adjustment via the main menu parameter level remains possible. In this case, however, each data block APROGD, APROGD2 or DPROGD must be selected separately. But parameters W0, W100 (adjustment limits) and Dp (decimal points) which pertain to the APROG are not effective when entering.

Recipe names used via text blocks are displayed also on the editing page. By changing the recipe name, switching over to the display of another recipe is possible. This can be done at any time and does not cause changing of the active recipe.

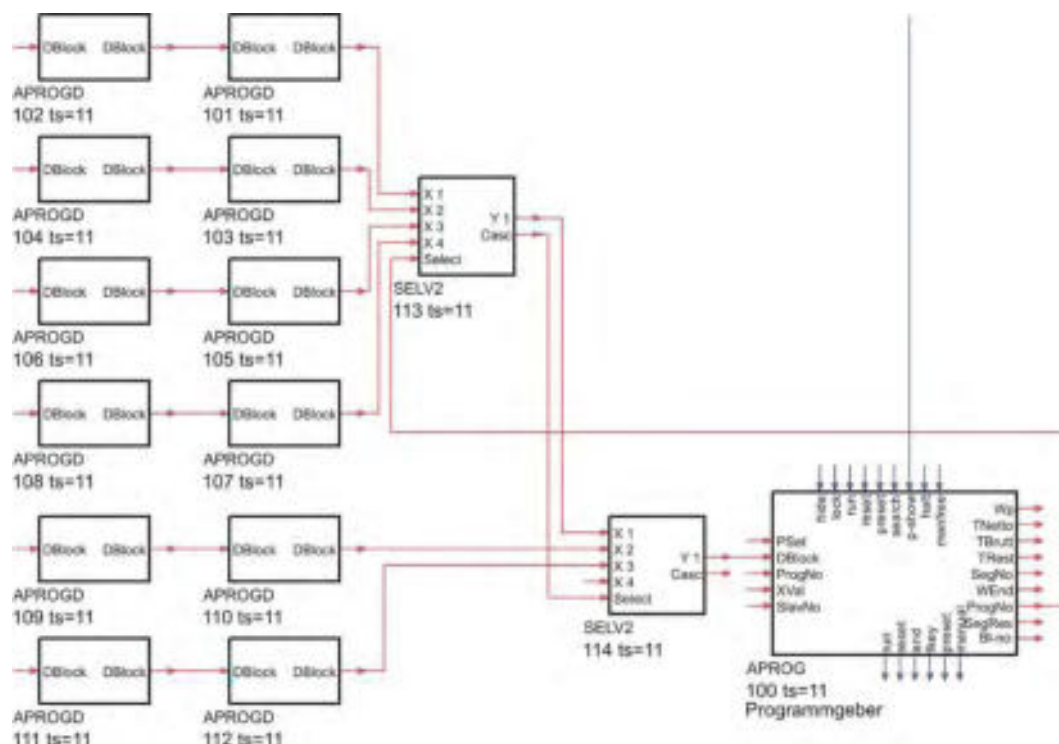
When using APROGD2 blocks, the following editing page is displayed.



If an X input of recipe switchover block SELV2 is not connected and the relevant recipe is selected nevertheless (should be disabled via the adjustment range of the recipe number), the following error display is output:


## Access to parameters of inactive recipes

To enable the access to all recipes relevant for this programmer block from the programmer program edit page (including the inactive ones), the following wiring principle is compulsory:



The SELV2 block switches the parameter block number onto the programmer Dblock input. Via the SELV2 block structure information, the programmer has access to all recipes.

Unless wiring is via SELV2, switchover to a different recipe on the parameter setting page is not possible, i.e. the recipe cannot be displayed.

 For active recipe switchover during reset status, another wiring type can be selected as well; however, at the latest 800 ms after switching over, the block number of the first parameter block of a new recipe must be applied safely at the Dblock input. The order of SELV2 block numbers plays an important part, in particular, if these blocks were assigned to the 800ms time slot. Unless the order is ascending, there will be an additional 800 ms delay at each cascade step

## Master/slave operation

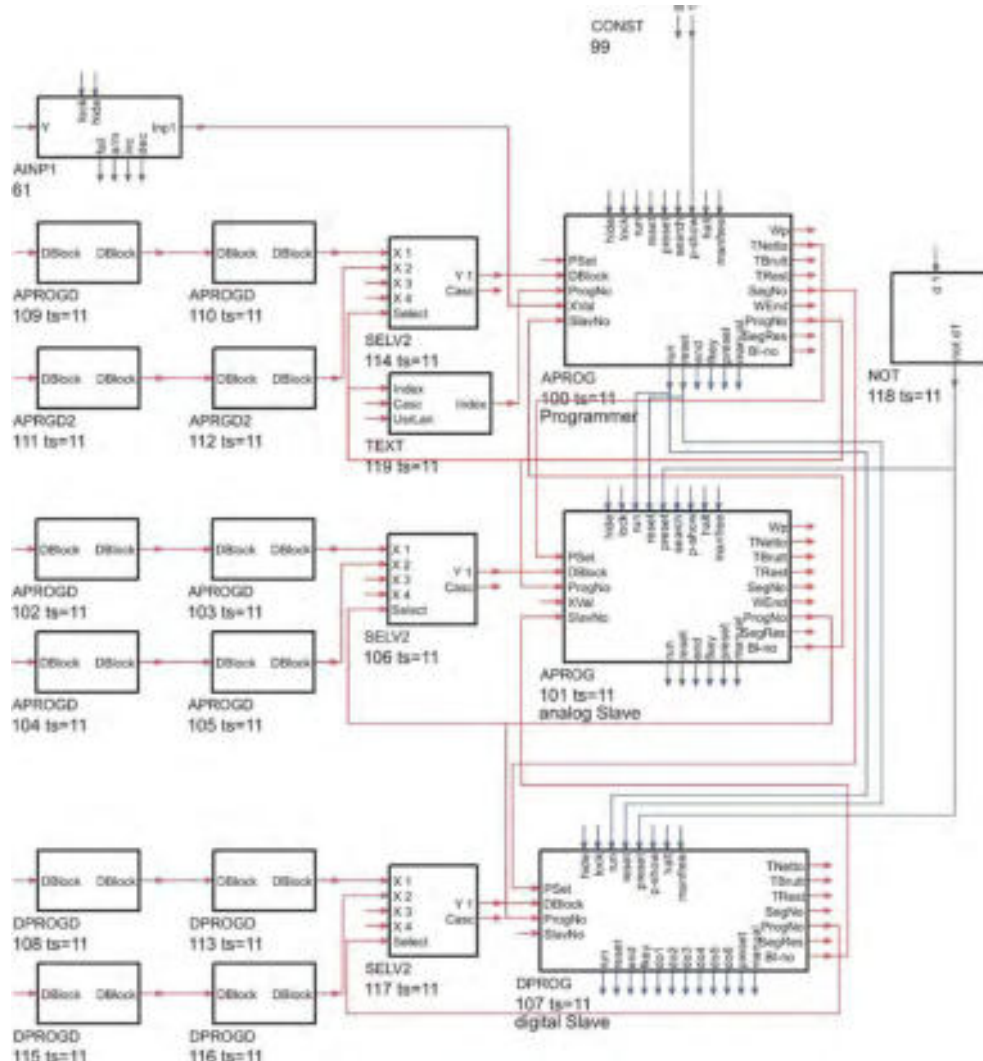
Programmers frequently consist of several coupled blocks which have a common time or segment structure (e.g. master block: oven temperature, 1st slave block atmosphere/C level, 2nd slave block 1..6 digital control signals). Such a programmer is provided with an operating page extending to all blocks in KS98-1. On the master operating page, the slave block data can be displayed via symbol



## Wiring

Synchronization of several programmers is by slave block preset coupling. The slave blocks are forced to the same time or to the same segment number via time or segment preset by the master.

Programmer with two analog and one digital blocks



To facilitate operation of the coupled blocks, the programmer has a SlavNo input and a BI-no output. These are used by the slave block for transfer of the block number to the following block. The block the BI-no output of which is not connected (block 100 in this example) should operate as master. Its Tnetto or SegNo output is wired to the PSet input of further blocks.

By linking the blocks (**B1 -no** → **SlavNo**), a programmer with coupled blocks is created. From the master block operating page, access to the data (incl. parameters) relevant for the slave for display or adjustment is possible easily.

## Operation of a programmer with several blocks

### Calling up a master block operating page via the operating page menu (page survey):

When selecting the operating page of a master programmer via the operating page menu with master/slave wiring via BI-no > SlavNo as described above, changing between the relevant programmer blocks can be done easily via the -symbol ►►. The order is determined by the wiring order (in the above example: 100 ⇒ 101 ⇒ 102 ⇒ 100 ⇒ ...)

However, changing to the next programmer block is not complete. Only some of the values and texts (e.g. title) relevant for the next block are displayed. The remaining elements continue indicating only the master information (see analog programmer operating page).

In case a change to the operating page menu (page survey) is made in this condition of the operating page, the block selection remains unchanged. I.e. when calling up the master block again later, the data of the slave block indicated last are displayed.

### ***Display information which is firmly assigned to the master block:***

- Recipe name (can be switched over in reset state)
- Program net time (adjustable for Preset to time)
- Remaining program time
- Status display for halt/end
- Status display for stop/run/reset/search/program/quit/error (adjustable)

### ***Display information of the actual programmer (master or slave):***

- Programmer block name
- Process value
- Segment number (adjustable for preset to segment only with master)
- Actual or actual control bits (both adjustable in manual mode)
- Segment start and end value
- Remaining segment time
- Status display for automatic/manual (adjustable, if permitted via manfree input)

As only the master shall decide upon the change of an active recipe, the wiring structure must be of such kind that changing is effective also for all relevant slave blocks (master ProgNo output  $\Rightarrow$  slave ProgNo input, see above diagram). This kind of master/slave coupling permits only a central recipe change for all blocks coupled accordingly.

### ***Calling up a slave block operating page via the operating page menu (page survey):***

When calling up the operating page of a slave programmer via the operating page menu, display of symbol ►► is suppressed. Easy changing to other blocks connected via the **B1 -no > S1 avNo** coupling as described above is not possible. Moreover, no data of the connected master are displayed.

To prevent inadmissible adjustments (recipe selection, run/stop/reset) from being offered on a called up operating page, outputs ProgNo, run and reset of the master programmer should be wired to the relevant slave programmer inputs. For plant operation, slave programmer display is suppressed purposefully with hide=1 when the page survey is active (PageNo at status block = 0).

### ***Subordinated parameter page (program editing page):***

On the subordinated parameter page, recipe switchover is possible at any time. But switchover acts only on the recipe parameter display on this page rather than changing to the effective recipe.

Changing directly to the parameters of the next programmer is not possible. For this, the superordinate operating page of the corresponding programmer must be used.

### **Programmers without coupling:**

On an operating page with a function block which is not connected with other programmer blocks via **B1 -no > S1 avNo** coupling, display of symbol ►► is suppressed.



### **Remaining segment time**

The remaining time of the actual segment is displayed on the operating page  
It is:

- readable via interface
- available as an additional analog output signal
- always 0 with reset
- suppressed with "preset to segment"

### **Incompatibility to earlier KS 98 functionality**

#### ***Recipe switchover:***

KS 98: The recipe number can be switched over at any time on the programmer operating page. However, the new selected recipe will be effective only after the next reset. On the subordinated parameter page, switching over has the same effect.

KS 98-1: On the programmer operating page, the recipe number can be switched over only during reset status. Switchover is effective immediately. On the subordinated operating page, switching over continues being possible at any time. Nevertheless, only the recipe to be displayed with its parameters is switched over. The instantaneously active recipe remains unaffected.

#### ***End behaviour with PEnd = 'stop':***

KS 98: Program is at the end, status is 'run', reset command leads to an immediate re-start

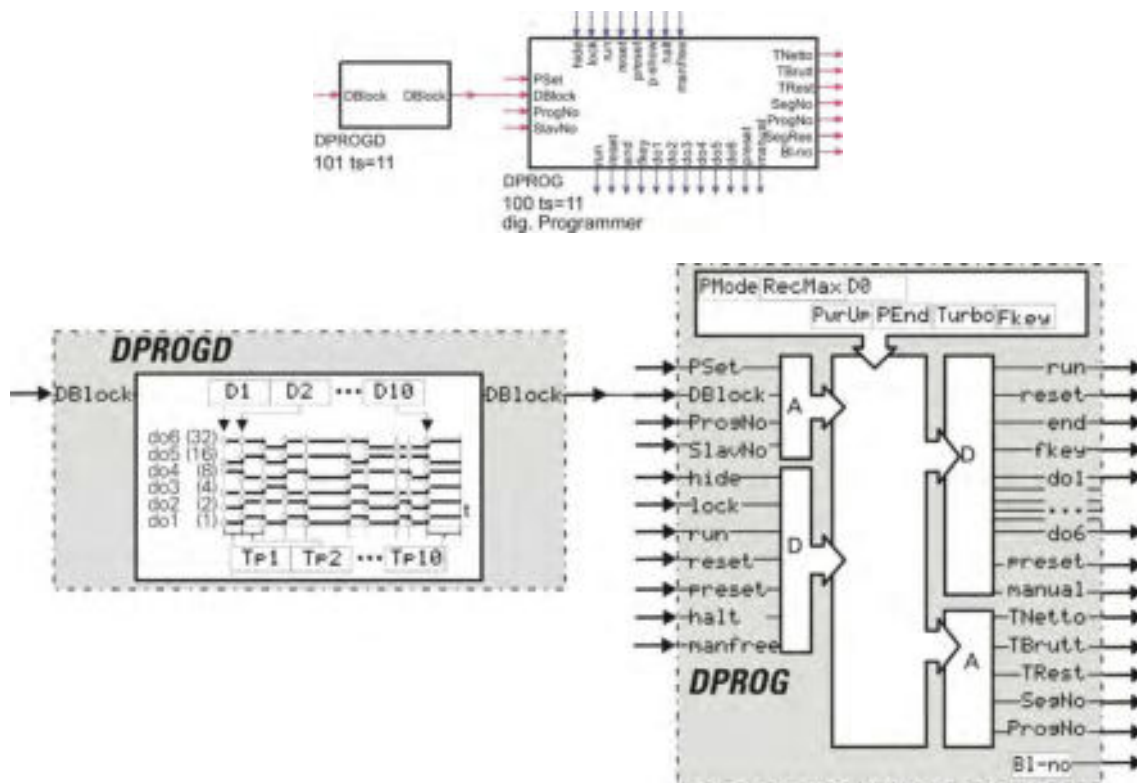
KS 98-1: Program stands at the end, status is 'stop', the program remains in reset status after a reset command

#### ***Segment number at program end:***

KS 98: At program end, the number of the last segment is displayed as segment number (SegNo output, operating page, interface).

KS 98-1: At program end, the number of the last segment + 1 is displayed as segment number (SegNo output, operating page, interface), in order to bring also any slave programmer to the end status.

### 3.15.2. DPROG ( digital programmer (No. 27)) / DPROGD ( DPROG data(No. 28))

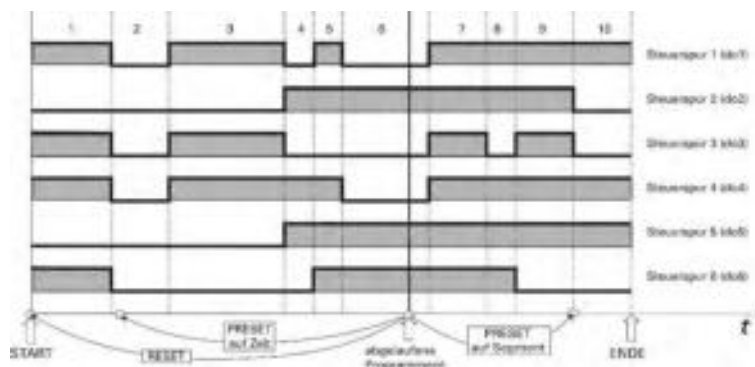


#### General

A digital programmer comprises a programmer (DPROG) and at least one data block (DPROD), whereby output **DBlock** of DPROGD is connected with input **DBlock** of the DPROG. Connection of several of these cascable functions (each with 10 segments) permits realization of a programmer with any number of recipes and any number of segments. Limiting is only in the number of available block numbers and calculation time.



The data block has an analog output, at which it provides its own block number. This information is read-in by the programmer and used for segment data addressing. When an error in the segment data addresses is found, the reset value is output (status display on operating page: **Error**).

After an engineering download, **Seg 0** is output (reset). If **run** is not connected, **stop** is used





## Inputs/outputs

### Digital inputs (DPROG):

<b>hide</b>	Display suppression (with <b>hide</b> = 1 the page is not displayed in the operation).	
<b>lock</b>	Adjustment blocking (with <b>lock</b> = 1 the values are not adjustable by means of keys   )	
<b>run</b>	Programm Stop/Run (0 = stop, 1 = run)	
<b>reset</b>	Programm Continue/Reset (0 = continue, 1 = reset)	reset has highest priority
<b>preset</b>	Programm Preset (1 = preset)	
<b>p-show</b>	Program editing enabled	
<b>halt</b>	halt: 0 = program not halted 1 = program halted	
<b>manfree</b>	manfree: 0 = switchover to manual mode not permitted 1 = switchover to manual mode is permitted	

### Digital outputs (DPROG):

<b>run</b>	Status program stop/run (0 = program stop ; 1 = program run)
<b>reset</b>	Status program reset (1 = program reset)
<b>end</b>	Status program end (1 = program end reached)
<b>fkey</b>	Status  key / interface function 'fkey' (:pressing key  causes switch-over (0 or 1))
<b>do1...do6</b>	status of control outputs in the actual segment
<b>preset</b>	preset: 0 = no preset status 1 = DPROG stands in preset status
<b>Manual</b>	manual: 0 = DPROG works in automatic mode 1 = DPROG works in manual mode

### Analog inputs (DPROG):

<b>PSet</b>	Preset value for program
<b>DBlock</b>	Block number of 1st data function 'DPROGD'
<b>ProgNo</b>	Required program number (recipe)
<b>SlavNo</b>	SlavNo: Block number of a connected slave block (for coupling master and slave blocks (APROG or DPROG))

### Analog inputs (DPROGD):

<b>DBlock</b>	Block number of cascable data function 'DPROGD'
---------------	---

### Analog outputs (DPROG):

<b>TNetto</b>	Net program time ( $\sum T_{run}$ )
<b>TBrutt</b>	Gross program time ( $\sum T_{run} + \sum T_{stop}$ )
<b>TRest</b>	Programmer rest time
<b>SegNo</b>	Actual segment number
<b>ProgNo</b>	Actual program number (recipe)
<b>SegRest</b>	Remaining segment time
<b>B1-no</b>	Own block-number (e.g. for coupling master and slave blocks)

### Analog outputs (DPROGD):

<b>DBlock</b>	Own block number
---------------	------------------

## Parameter and configuration data

Parameter DPROG	Description	Range	Default
<b>PMode</b>	Preset Mode: Preset to segment Preset to time	<b>Pres. Seg.</b> <b>Pres. Zeit</b>	←
<b>RecMax</b>	Max.recipees	1..99	99
<b>D0</b>	Status of control outputs 6...1 with reset	0 / 1 je Spur	000000

Parameter	Description	Range	Unit	Default	Unit
DPROGD		ET		ET	
<b>TP 1</b>	Time for segment 1 (①)	0 ... 59 999	<b>0:00...999:59</b>	OFF	--:--
<b>D 1</b>	Status of control output values in segment 1 (②)		0 / 1 per output	000000	<b>000000</b>
<b>TP 2</b>	Time for segment 2 (①)	0 ... 59 999	<b>0:00...999:59</b>	OFF	--:--
<b>D 2</b>	Status of control output values in segment 2 (②)		0 / 1 per output	000000	<b>000000</b>
...					
<b>TP 10</b>	Time for segment 10 (①)	0 ... 59 999	<b>0:00...999:59</b>	OFF	--:--
<b>D 10</b>	Status of control output values in segment 10 (②)		0 / 1 per output	000000	<b>000000</b>

The time for a segment is entered in the engineering tool in seconds or minutes dependent of configuration (**Turbo**), whereas entry into the unit is in **Hrs:Min** or **Min:Sec**. In addition to the range, a switch-off value can be entered (ET: OFF/-32000; unit: --:--). When reaching a segment with a switch-off value, 'End' is output.

With entry of control values in the engineering tool, the first digit before the decimal point corresponds to control output 1 (do1), the second digit before the decimal point corresponds to control output 2 (do2) etc. Entries behind the decimal point are interpreted as 0. Leading zeros are deleted.



Configuration	Description	Value
DPROG		
<b>PwrUp</b>	Behaviour after mains recovery	Continue program (default) Continue at actual time stop after program end (default)
<b>PEnd</b>	Behaviour at program end PEnd: 0 = Stop 1 = Reset 2 = Reset + Stop (End-condition is reset with stop)	Reset after program end <b>Reset</b>
<b>Turbo</b>	Time unit	Time = hours : minutes (default) Time = minutes : seconds <b>Std:Min</b> <b>Min:Sek</b>
<b>FKey</b>	FKey (F-key function): 0 – fkey output status switchover by F key (previous function) - 1 – F key used to generate a pulse at the fkey output (pulse length = 1 cycle) 2 – F key controls the programmer (fkey output generates a pulse when actuating the key, pulse length = 1 cycle)	

## DPROG- functions

In der folgenden Liste sind alle beim digitalen Programmgeber wirksamen Funktionen aufgeführt. Da fast alle Punkte beim analogen Programmgeber genauso verwendet werden, wird für die Beschreibung auf das entsprechende APROG-Kapitel verwiesen.

- Data blocks are cascable (like APROG, →S. 188)
- Program selection (like APROG, →S. 188)
- Program changes during an active recipe (like APROG, →S. 193)
- Access to parameters of inactive recipes (like APROG, →S. 198)
- Programmer control via key F (like APROG, →S. 191)
- Halt status (like APROG, →S. 191)
- Automatic/manual mode with adjustability of control bits (like APROG, →S. 191)
- Recipe changing during reset status (like APROG →S. 188)
- Recipe names via TEXT block coupling (like APROG, →S.189)
- Program end behaviour (like APROG, →S. 189)
- Master/slave operation (like APROG, →S. 198)
- Remaining segment time (like APROG, →S. 201)
- Operating page elements like for the APROG (with display of control blocks and relevant block number, →S. 195)

## Digital programmer operating page

Digital programmer DPROG has an operating page, which can be selected in the operating page menu with input 'hide' not connected. If the FB inputs assigned to the input fields (function block inputs) in the following table are allocated to the engineering, operation (change) of this input field is not possible.

- ① \* Program block name
- ② Recipe name: recipe can be switched over in reset status.
- ③ Segment number: adjustable with segment preset
- ④ \* automatic / manual (adjustable))
- ⑤ \* Page changing::  
permits switching to an analog or digital page connected by block coupling. This switchover type applies only to the values marked with \*. The remaining display elements invariably show the master programmer values
- ⑦ Control bit
- ⑧ \* Control bits: can be altered individually in manual mode
- ⑨ Remaining segment time: display is suppressed with preset to segment (e.g. with digital slave programmers). Otherwise the invariable overall segment time would be displayed.
- ⑩ \* Program net time: adjustable with preset to time.
- ⑪ Remaining program time
- ⑫ Programmer status stop / run / reset / program / quit / error
- ⑬ halt / end (if none of the two conditions is active, this display remains empty)

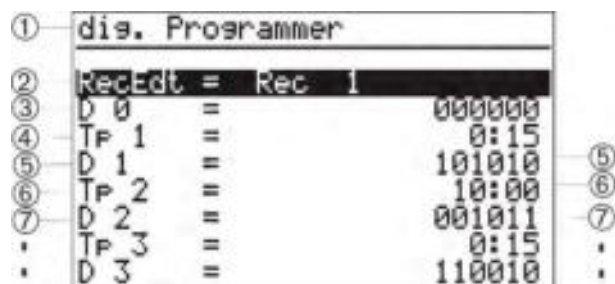


## DPROG program edit page

Like the analog programmer, changing from the operating page to the subordinated parameter page is done by setting the status text shown at the bottom right to 'program' (possible only with p-show = 1). Display includes recipe name, the 6 start control bits and the segment parameters of the instantaneously active recipe.

Switchover to display of a different recipe is by changing the recipe name. This is possible at any time and does not cause switchover of the active recipe.

- ① Name of program block
- ② Recipe name: Recipe changing is possible at any time.
- ③ Control bit status in reset mode
- ④ Time for segment 1
- ⑤ Control bit status in segment 1
- ⑥ Time for segment 2
- ⑦ Control bit status in segment 2



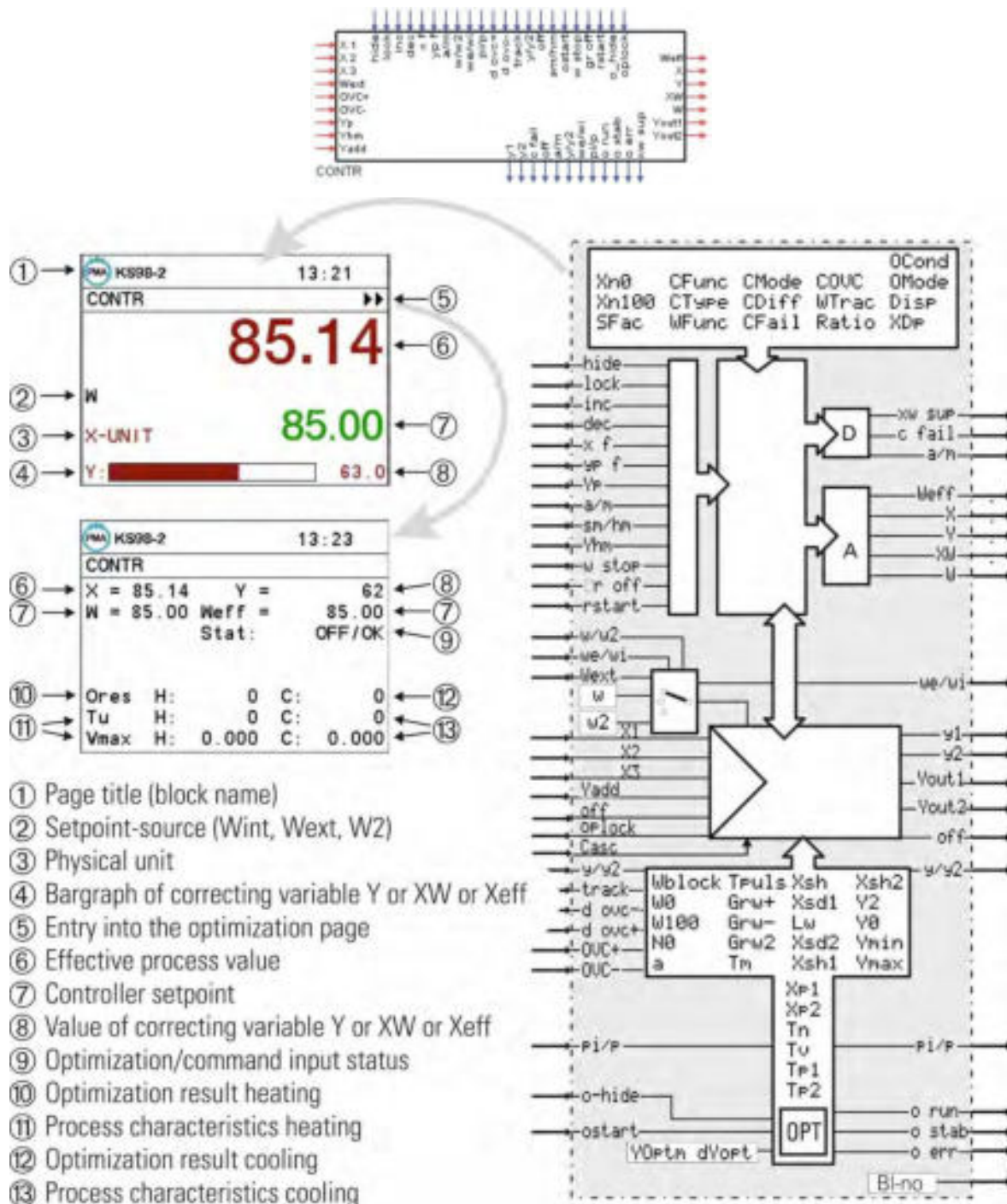
### 3.16. Controller

General: Function blocks CONTR and CONTR+ and PIDMA provide a complex control function. Unlike CONTR, CONTR+ includes six selectable control parameter sets. However, PIDMA provides a special control algorithm and different self-tuning.

In the following sections, the main features of the functionblocks CONTR, CONTR+ together and PIDMA separately are described. Then the common controltechnical application ranges are explained.

### 3.16.1. CONTR (Controller with one parameterset (No. 90))

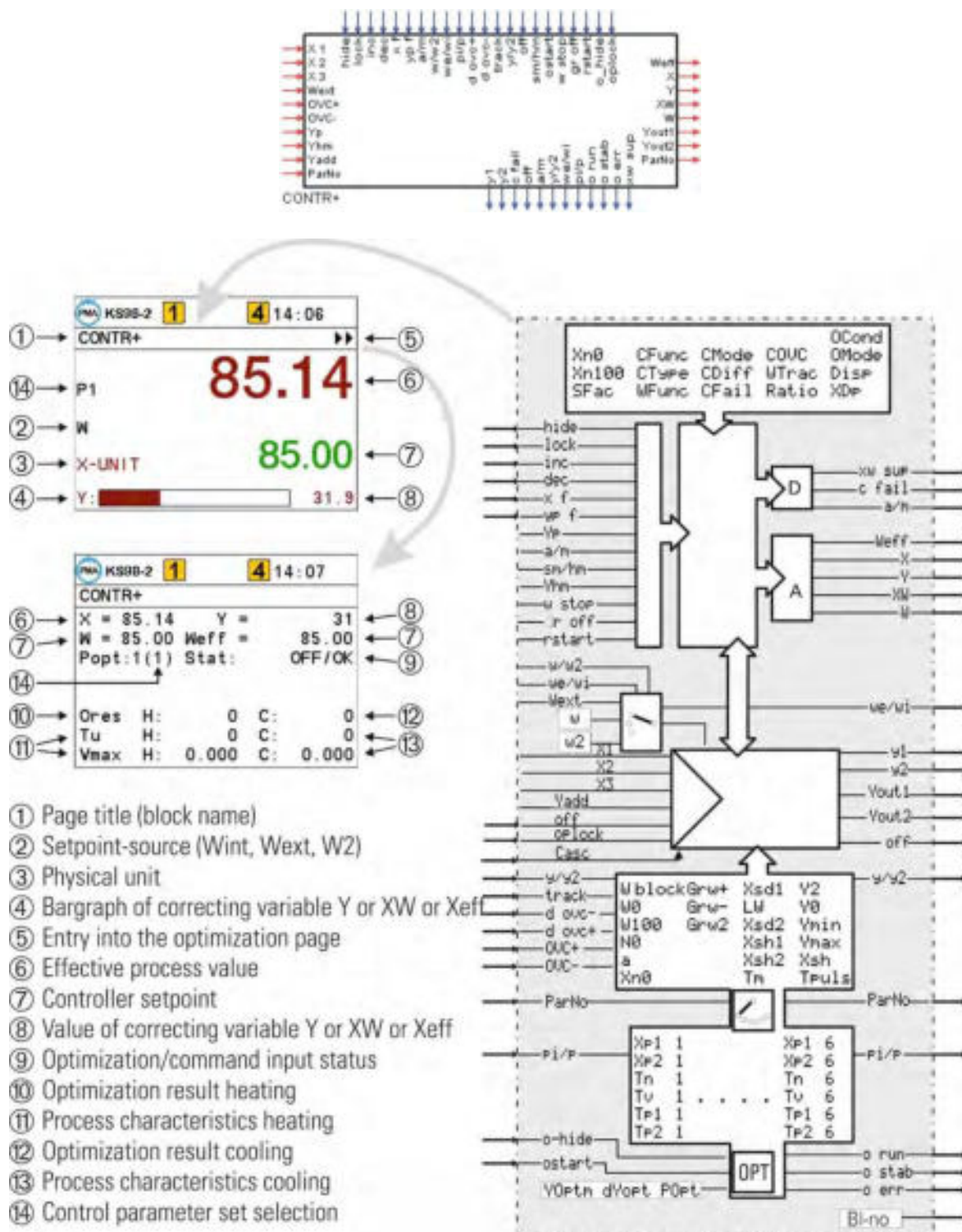
The CONTR block provides a PID controller with numerous functions such as ramp, internal/external/W2 switchover, /process value tracking, self-tuning, override control, feed-forward control, positioning signal forcing, ratio and three-element control in 12 different controller types (continuous/2-point/3-point/3-point stepping/ ...).









### 3.16.2. CONTR+ (Controller with six parametersets (Nr. 91))

Function block CONTR+ provides the same functionality as the CONTR block. Additionally, it permits adaptation by adjustment. Six parameter sets can be activated dependent on process criteria (process value, correcting variable, control deviation), plant or batch characteristics. The parameter sets can be determined independently by self-tuning.



**Inputs/outputs for CONTR and CONTR+****Digital inputs:**

<b>hide</b>	Display suppression (with hide = 1 the operating page is not displayed).
<b>lock</b>	Adjustment locking (with lock = 1 the values are not adjustable by means of keys   ).
<b>inc</b>	Increment for manual adjustment
<b>dec</b>	Decrement for manual adjustment
<b>x f</b>	Sensor error x1...x3
<b>yp f</b>	Sensor error Yp
<b>a/m</b>	0 = automatic      1 = manual
<b>w/w2</b>	0 = int./ext.      1 = W2
<b>we/wi</b>	0 = external      1 = internal
<b>pi/p</b>	0 = PI action;      1 = P action 1) (→ page PI/P switch-over) (not applicable to PIDMA)
<b>d ovc+</b>	1 = override control+ with 3-point stepping controller (→ page 250 ff) (not applicable to PIDMA)
<b>d ovc</b>	1 = override control - with 3-point stepping controller (→ page 250 ff) (not applicable to PIDMA)
<b>track</b>	0 = tracking function off; 1 = tracking function on (→ page 244; 245)
<b>y/y2</b>	0 = output value Y1 = output value Y2
<b>off</b>	0 = controller switched on 1 = controller switched off
<b>sm/hm</b>	0 = Soft manual 1 = Hard manual
<b>ostart</b>	1 = self-tuning start → page 46 ff)
<b>w stop</b>	1 = effective freeze (can e.g. be used for bandwidth monitoring)
<b>gr off</b>	1 = Setpoint gradient suppression
<b>rstart</b>	1 = Start the ramp → the makes a step change towards the process value and goes to the adjusted process value according to GRW+ (GRW-). The rising flank (0 → 1) is evaluated.
<b>o-hide</b>	1 = self-tuning page display suppression
<b>oplock</b>	Blockage of key  (with oplock = 1, switchover to manual by means of key  is not possible).

**Digital outputs:**

<b>y1</b>	Status of switching output Y1; 0 = off 1 = on
<b>y2</b>	Status of switching output Y2; 0 = off 1 = on
<b>c fail</b>	1 = controller in error handling
<b>off</b>	0 = controller switched off 1 = controller switched on
<b>a/m</b>	0 = automatic 1 = manual
<b>y/y2</b>	0 = output value Y 1 = output value Y2
<b>we/wi</b>	0 = external 1 = internal
<b>pi/p</b>	feedback/integrator 0 = with 1 = without (not applicable to PIDMA)
<b>o run</b>	Self-tuning running
<b>o stab</b>	Process at rest (for self-tuning) (not applicable to PIDMA)
<b>o err</b>	Error during self-tuning
<b>xw sup</b>	Alarm suppression with change

**Analog inputs:**

<b>x1</b>	Main variable x1
<b>x2</b>	Auxiliary variable x2      e.g. for ratio control
<b>x3</b>	Auxiliary variable x3      e.g. for 3-element control
<b>Wext</b>	External
<b>OVC+</b>	Override Control+ → page 250 ff)
<b>OVC-</b>	Override Control- → page 250 ff)
<b>Yp</b>	Position feedback
<b>Yhm</b>	Output with hard manual
<b>Yadd</b>	Feed-forward control
<b>ParNo</b>	Only with CONTR+; required parameter set
<b>Casc</b>	Cascade input for controller cascade



## Analog outputs:

<b>Weff</b>	Effective setpoint
<b>X</b>	Effective process value
<b>Y</b>	Effective output value
<b>XW</b>	Control deviation
<b>W</b>	Internal setpoint
<b>Yout1</b>	Output value yout1 (heating)
<b>Yout2</b>	Correcting variable yout2 (cooling; only with continuous controller with split-range behaviour r CFunc= splitRange)
<b>ParNo</b>	Only with CONTR+; effective parameter set
<b>B1 -no</b>	Own block number

**3.16.3. Parameter and configuration for CONTR, CONTR+**

## Parameter for CONTR and CONTR+

Parameter	Description	Range	Default	Unit
<b>W_Block</b>	Setpoint	Switchover is disabled via front-panel operation.	←	
	switchover	Wext ↔ Wint switchover is disabled t		
	disable	W ↔ W2 switchover is disabled		
	function	All switchover functions are enabled		
<b>W0</b>	Min. setpoint limit (Weff)	-29999...999999	0	<b>0</b>
<b>W100</b>	Max. setpoint limit (Weff)	-29999...999999	100	<b>100</b>
<b>W2</b>	Additional setpoint	-29999...999999	100	<b>100</b>
<b>Grw+<sup>3)</sup></b>	Setpoint gradient plus	unit/min	0,001...999999	Aus ----
<b>Grw-<sup>3)</sup></b>	Setpoint gradient minus	unit/min	0,001...999999	Aus ----
<b>Grw2<sup>3)</sup></b>	Setpoint gradient for W2	unit/min	0,001...999999	Aus ----
<b>N0</b>	Zero offset for ratio control	-29999...999999	0	<b>0</b>
<b>a</b>	Factor a for 3-element control + setpoint ramps	-9,99...99,99	1	<b>1</b>
<b>Xsh<sup>2)</sup></b>	Trigger point separation (stepping controller)	0,2...20,0%	0,2	<b>0,2</b>
<b>Tpu1s</b>	Min. positioning step time (stepping controller)	0,1...2,0[s]	0,3	<b>0,3</b>
<b>Tm</b>	Actuator response time (stepping controller)	5...999999 [s]	30	<b>30</b>
<b>Xsd1</b>	Switching difference (signaller control)	0,10...999999	1	<b>1</b>
<b>LW</b>	Distance additional contact (signaller control)	-29999...999999	Aus	----
<b>Xsd2</b>	Switching difference additional contact (signaller control)	0,10...999999	1	<b>1</b>
<b>Xsh1<sup>1)</sup></b>	Trigger point separation (PD) (3-point controller)	0,0...1000,0[%]	0	<b>0</b>
<b>Xsh2<sup>1)</sup></b>	Trigger point separation (PD) (3-point controller)	0,0...1000,0[%]	0	<b>0</b>
<b>Y2</b>	Additional correcting variable (not for stepping controllers)	-105,0...105,0[%]	0	<b>0</b>
<b>Ymin</b>	Min. correcting variable limiting (not for stepping controller)	-105,0...105,0[%]	0	<b>0</b>
<b>Ymax</b>	Max. correcting variable limiting (not for stepping controller)	-105,0...105,0[%]	100	<b>100</b>
<b>Y0</b>	Working point of the controller (not for stepping controller)	-105,0...105,0[%]	0	<b>0</b>
<b>Y0ptm<sup>4)</sup></b>	Positioning value during process at rest	-105,0...105,0[%]	0	<b>0</b>
<b>dYopt<sup>4)</sup></b>	Self tuning step height	5...100[%]	100	<b>100</b>
<b>P0pt<sup>4)</sup></b>	Only for CONTR+; parameter set to be optimized	1...6	1	<b>1</b>
<b>Xp1 1...6<sup>1)</sup></b>	Proportional band 1	0,1...999,9[%]	100	<b>100</b>
<b>Xp2 1...6<sup>1)</sup></b>	Proportional band 2 (three-point and split range)	0,1...999,9[%]	100	<b>100</b>
<b>Tn 1...6</b>	Integral action time (Tn = 0 → I-part is not effective)	0,0...999999[s]	10	<b>10</b>
<b>Tv 1...6</b>	Derivative time (Tv = 0 → D-part is not effective)	0,0...999999[s]	10	<b>10</b>
<b>Tp1 1...6</b>	Cycle time heating (2-and 3-point controller)	0,4...999,9[s]	5	<b>5</b>
<b>Tp2 1...6</b>	Cycle time cooling (3-point controller)	0,4...999,9[s]	5	<b>5</b>

<sup>1)</sup> %-specifications are related to measuring range  $x_{n0} \dots x_{n100}$ <sup>2)</sup> The neutral zone  $x_{sn}$  for 3-point stepping controllers depends on  $T_{puls}$ ,  $T_m$  and  $x_{p1}$  (→ V. Hints for self-tuning).<sup>3)</sup> Gradient control → Seite 243<sup>4)</sup> Self optimization → Seite Fehler! Textmarke nicht definiert. ff

**Configuration data CONTR, CONTR+**

Configuration	Description	Values	Default
<b>CFunc</b>	Control behaviour:	Signaller 1 output	<b>Signal 1</b>
		Signaller 2 outputs	<b>Signal 2</b>
		2-point controller	<b>2-point</b>
		3-point controller (heating/cooling switching)	<b>3-point</b>
		3-point controller (heat.continuous/cool.switching)	<b>Cont/swi</b>
		3-point controller (heat.switching/cool.continuous)	<b>Swi/Cont</b>
		triangle-star-off ( $\Delta/Y$ -off)	<b>2P-DS0</b>
		3-point stepping controller	<b>Stepping</b>
		3-point-stepping controller with pos. feedback Yp	<b>Step+Yp</b>
		Continuous controller	<b>Cont</b> ←
<b>CType</b>	Controller type	Continuous controller with split-range operation	<b>splitRang</b>
		Continuous controller with position feedback Yp	<b>Cont Yp</b>
		Standard controller	<b>Standard</b> ←
		Ratio controller	<b>Ratio</b>
<b>WFunc</b>	Setpoint function	3-element controller	<b>3-elem.</b>
		Setpoint control	<b>Set-point</b> ←
<b>CMode</b>	Output action	cascade control	<b>Sp/casc</b>
		Inverse output action	<b>Invers</b> ←
<b>CDiff</b>	Differentiation	Direct output action	<b>Direct</b>
		Xw differentiation	<b>Xw</b> ←
		X differentiation	<b>X</b>
<b>CFail</b>	Behaviour with sensor error	Neutral	<b>Neutral</b>
		Ypid = Ymin (0%)	<b>Ymin</b> ←
		Ypid = Ymax (100%)	<b>Ymax</b>
		Ypid = Y2 (no adjustment via front panel)	<b>Y2</b>
		Ypid = Y2 ((automatic) or Yman (manual operation))	<b>Y2/Yman</b>
<b>COVC</b>	Output limiting	No override control	<b>Off</b> ←
		Override-Control +	<b>OVC+</b>
		Override-Control -	<b>OVC-</b>
		Override-Control + / -	<b>OVC+ / OVC-</b>
<b>WTrac</b>	Tracking of int. setpoint	No tracking of Wint	<b>Off</b> ←
		Setpoint-tracking	<b>SP-track</b>
		Process value tracking	<b>PV-track</b>
<b>Ratio</b>	Ratio controller function:	(x1 + N0) / x2	<b>Typ 1</b> ←
		(x1 + N0) / (x1 + x2)	<b>Typ 2</b>
		(x2 - x1 + N0) / x2	<b>Typ 3</b>
<b>XDp</b>	Digits behind the decimal point (process value)	0...3	0
<b>Disp</b>	Contents of bargraph line:	Output variable	<b>Y</b> ←
		Control deviation	<b>XW</b>
<b>OMode</b>	Self-tuning mode:	Xeff	<b>Xeff</b>
			<b>Standard</b> ←
<b>OCond</b>	Condition for process at rest:	grad = 0	<b>grad=0</b> ←
		grad < 0 (controller inverse) grad > 0 (controller direct)	<b>grad&lt;0 / &gt;0</b>
		grad <= 0	<b>grad&lt;=0</b>
<b>Xn0</b>	Span start	-29999 ... 999999	0
<b>Xn100</b>	Span end	-29999 ... 999999	100
<b>SFac</b>	Factor stoichiom. ratio	0,01 ... 99,99	1,00

### 3.16.4. Control behaviour

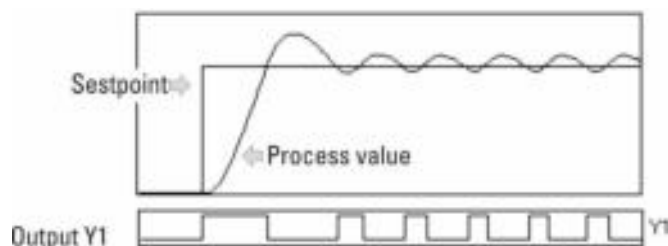
The following chapter describes the different control behaviours adjustable with the configuration parameter CFUNC and determines the parameters effective then.

All available parameters can be adjusted in the engineering tool. However it is not recognizable, which of the adjusted values are really affecting the process.

The following compilation shall help to enlighten, which parameters are really used, dependent from the adjusted controller type. Therefore the relevant parameters for control behaviour are accented grey in the tables.

#### Signaller, 1 output:

The signaller is suitable for processes with small  $T_u$  and low  $v_{max}$ .



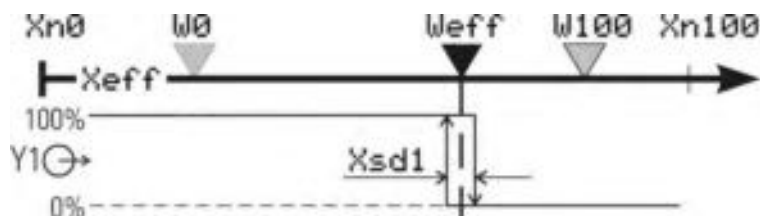
The advantage is in the low switching frequency. Switch-on is always at a fixed value below the , switch-off is always at a fixed value above the .

The control variable oscillation band is determined as a result of:

$$X_0 = X_{max} \cdot \frac{T_u}{T_g} + X_{sd} = v_{max} \cdot T_u + X_{sd}$$

The signal function corresponds to limit signalling, whereby the is the limit value. The trigger point is symmetrical to the; hysteresis  $X_{sd1}$  is adjustable.

Static operating principle of the signalling function of a signaller, 1 output



Configuration	Effective controller parameters of a signaller with one output		
CFunc = signaller, 1 output	<b>W0<sup>1)</sup></b>	Lower setpoint limit for Weff	-29 999 ...999 999
	<b>W100<sup>1)</sup></b>	Upper setpoint limit for Weff	-29 999 ...999 999
	<b>W2<sup>1)</sup></b>	Additional setpoint	-29 999 ...999 999
	<b>Grw+<sup>2)</sup></b>	Setpoint gradient plus	aus / 0,001 ... 999 999
	<b>Grw-<sup>2)</sup></b>	Setpoint gradient minus	aus / 0,001 ... 999 999
	<b>Grw2<sup>2)</sup></b>	Setpoint gradient for W2	aus / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with CType = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with CType = 3-element control)	-9,99 ... 99,99
	<b>Xsd1<sup>1)</sup></b>	Signaller switching difference	0,1 ... 999 999
	<b>Title</b>	Title of controller page (only display)	16 characters
	<b>X.unit</b>	Process value unit (only display)	6 characters
	<b>W.int</b>	Internal setpoint after transmission of the engineering to KS 98	-29 999 ...999 999

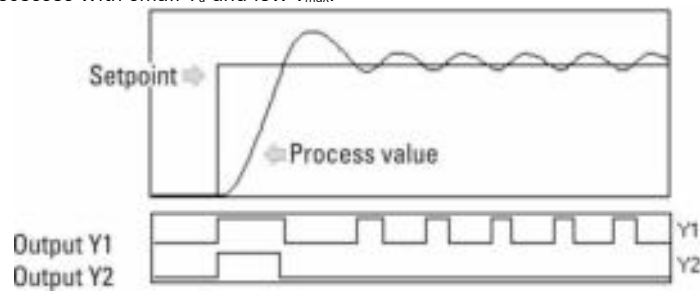
<sup>1)</sup> The values are specified in the process value unit - z.B. [°C, °F, bar, %, usw.]

<sup>2)</sup> The rate of change must be specified in units /minute (e.g. °C/min).

→ see gradient control page 243.

## Signaller, 2 outputs

The signaller is suitable for processes with small  $T_u$  and low  $v_{\max}$ .



The advantage is in the low switching frequency. Switch-on is always at a fixed value below the, whereas switch-off is always at a fixed point above the.

The control variable oscillation band is determined as a result of:

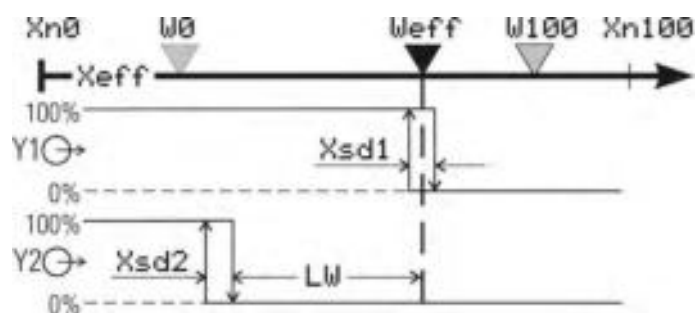
$$X_0 = X_{\max} \cdot \frac{T_u}{T_g} + X_{sd} = v_{\max} \cdot T_u + X_{sd}$$

The signalling function provides alarm signalling, whereby the is the limit value. The trigger point is symmetrical to the hysteresis  $X_{sd1}$  is adjustable.

The signaller with two outputs has an additional "limit contact". Its difference from the is adjustable in parameter LW (including polarity sign). The contact can be used to activate additional power levels, or to trigger an alarm when the setpoint distance is large, in a symmetrical position around the setpoint ( $LW$  negative and  $X_{sd2} = LW/2$  can also be used for bandwidth control or control deviation alarm).

*Static operating principle of the signalling function Signaller, 2 outputs*

*$LW$  is shown as a negative value in the example (e.g. -20)*



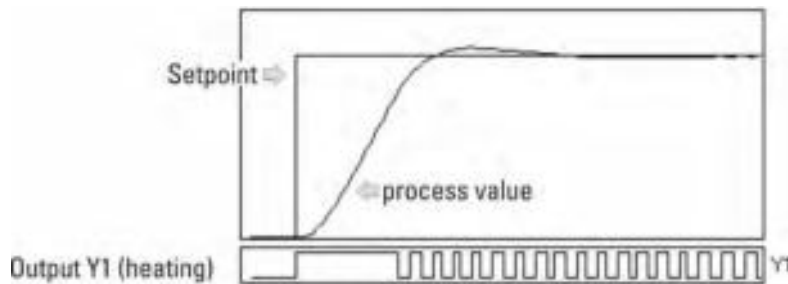
Configuration	Effective controller parameters of a signaller with two outputs		
CFunc = Signaller, 2 outputs	<b>W0<sup>1)</sup></b>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100<sup>1)</sup></b>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2<sup>1)</sup></b>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+<sup>2)</sup></b>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-<sup>2)</sup></b>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2<sup>2)</sup></b>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with CType = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with CType = 3-element control)	-9,99 ... 99,99
	<b>Xsd1<sup>1)</sup></b>	Signaller switching difference	0,1 ... 999 999
	<b>LW</b>	Trigger point separation of additional contact OFF $\triangleq$ the additional contact is switched off	-29 999 ... 999 999 -32 000 = Off
	<b>Xsd2<sup>1)</sup></b>	Signaller switching difference of additional contact	0,1 ... 999 999
	<b>Titel</b>	Signaller switching difference	16 characters
	<b>EinH.X</b>	Title of controller page (only display)	6 characters
	<b>W int</b>	Process value unit (only display)	-29 999 ... 999 999

<sup>1)</sup> The values are specified in the process value unit - e.g. [°C, °F, bar, %, etc.]

<sup>2)</sup> The rate of change must be specified in units / minute (e.g. °C/min) r see gradient control page 243.

## Two-point controller

Switching controller with two switching statuses:



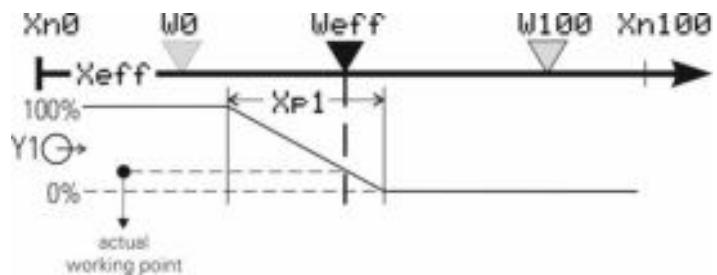
1. Heating switched on; → output Y1 = 1
2. Heating switched off; → output Y1 = 0

E.g. for temperature control with electrical heating (inverse operation) or cooling (direct operation).

Adjust cycle time  $T_{p1}$  as follows:  $T_{p1} = 0,25 \cdot T_u$

With higher  $T_{p1}$ , oscillations must be expected.  $T_{p1}$  corresponds to the minimum cycle time (time in seconds) at 50 % duty cycle.

*Static operating principle of a two-point controller*



PD action ( $T_n = 0 \triangleq$  abgeschaltet  $T_n = \infty$ )

The working point is in the middle of proportional band  $X_{p1}$  at 50 % duty cycle. In order to keep the control variable constant, a defined quantity of energy dependent of is required. This energy causes a permanent control deviation, which increases with growing  $X_{p1}$ .

DPID action

By means of the I action, line-out is without permanent control deviation.

The static characteristic of a two-point controller is identical to the one of a continuous controller, with the difference that a duty cycle instead of a linearly variable current signal is output (relay contact, logic signal 0/20mA or control output 0/24V).

Working point  $Y_0$  and cycle time  $T_{p1}$  at 50% are adjustable.

The shortest switch-on or switch-off time is 100ms.

Configuration	Effective controller parameters of a two-point controller		
<b>CFunc</b> = 2-Punkt	<b>Popt</b>	Parameter set for self-tuning (only with <b>CONTR+</b> )	1...6
	<b>W0</b> <sup>1)</sup>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100</b> <sup>1)</sup>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2</b> <sup>1)</sup>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+</b> <sup>2)</sup>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-</b> <sup>2)</sup>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2</b> <sup>2)</sup>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with <b>CType</b> = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with <b>CType</b> = 3-element control)	-9,99 ... 99,99
	<b>Y2</b>	Additional correcting variable	0 ... 100 [%]
	<b>Ymin</b>	Min. correcting variable limiting	0 ... 100 [%]
	<b>Ymax</b>	Max. correcting variable limiting	0 ... 100 [%]
	<b>Y0</b>	Correcting variable working point(start-up correcting variable)	0...100 [%]
	<b>YOptm</b>	Correcting variable during process at rest (not with PIDMA)	0...100 [%]
	<b>dYopt</b>	Self-tuning step change height	5...100 [%]
	<b>Xp1 (1...6)</b> <sup>3)5)</sup>	Proportional band 1	0,1 ... 999,9 [%]
	<b>Tn1 (1...6)</b> <sup>5)</sup>	Integral action time	0 ... 999 999 [s]
	<b>Tv1 (1...6)</b> <sup>5)</sup>	Derivative action time	0 ... 999 999 [s]
	<b>Tp1 (1...6)</b> <sup>5)</sup>	Cycle time heating	0,4 ... 999,9 [s]
	<b>Titel</b>	Title of controller page (only display)	16 characters
	<b>Einh.X</b>	Unit of the process value (only display)	6 characters
	<b>W int</b>	Internal after transmission of the engineering to KS98	-29 999 ... 999 999
	<b>A/H</b>	Controller status after transmission of the engineering to KS98	0 oder 1

<sup>1)</sup> The values are specified in the process value units - e.g. [°C, °F, bar, %, etc.]

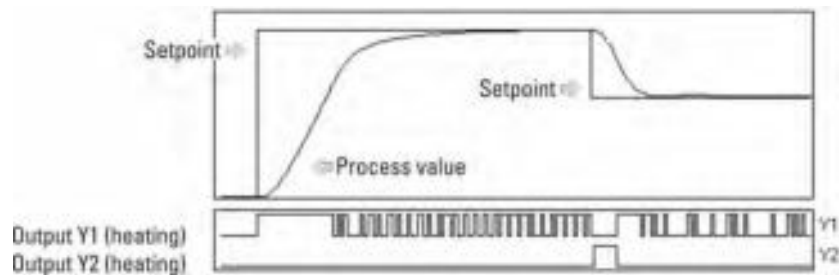
<sup>2)</sup> Specify the rate of change in units / minute (e.g. °C/min) → see gradient control page 260.

<sup>3)</sup> % specifications are related to measuring range **Xn100** - **Xn0**. They are not related to values **W0** and **W100**.

<sup>4)</sup> **(1...6)** refers to the six parameter sets of CONTR+ (e.g. Xp1, Xp2, Xp3...Xp6).).

### Three-point controller

Switching controller with three switching statuses:



1. Heating switched on; → output Y1 = 1, Y2 = 0
2. Heating and cooling switched off; → output Y1 = 0, Y2 = 0
3. Cooling switched on; → output Y1 = 0, Y2 = 1

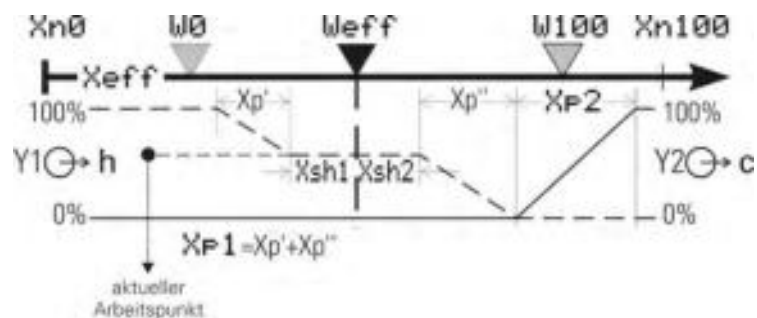
E.g. for temperature control with electrical heating (h) and cooling (c).

Adjust cycle time **Tp1** and **Tp2** as follows:

$$Tp1 \leq 0,25 \cdot Tu (h) \quad Tp2 \leq 0,25 \cdot Tu (c)$$

With higher **Tp1/Tp2**, oscillations have to be expected. Cycle times **Tp1** and **Tp2** are the minimum cycle times at 50% duty cycle.

*Static operating principle  
of a three-point controller*



PD/PD action ( $T_n = 0 \triangleq$  switched off  $T_n = \infty$ )

The positioning range reaches from 100 % heating (Y1) to 100 % cooling (Y2). The proportional bands must be adapted to the various heating and cooling power values. In order to keep the control variable constant, a defined amount of energy dependent of is required. This causes a permanent control deviation, which increases with growing  $X_{p(1,2)}$ .

DPID/DPID action

By means of the I action, line-out without permanent control deviation is possible. Transition from trigger point 1 (heating) to trigger point 2 (cooling) is without neutral zone. The proportional bands must be adapted to the various heating and cooling power values. The picture shows the static characteristic for inverse output action.

Direct/inverse switchover only causes exchanging of the outputs for "heating/cooling".

Expressions "heating" and "cooling" may also mean similar processes (dosing acid/lye, ...).

The neutral zone is adjustable separately for the trigger points ( $X_{sh1}$ ,  $X_{sh2}$ ) i.e. it need not be symmetrical to the.

The type of positioning signals is selectable:

<b>CFunc</b> = 3-point	heating switching,	cooling switching
<b>CFunc</b> = cont/switch	heating continuous,	cooling switching
<b>CFunc</b> = switch/cont	heating switching,	cooling continuous

Combination "heating continuous" and "cooling continuous" is covered by "splitRange - continuous controller with split-range behaviour". → see also "continuous controller" page: 221.

Configuration	Effective controller parameters with three-point controller		
CFunc = 3-Punkt	<b>Popt</b>	Parameter set for self-tuning (only with <b>CONTR+</b> )	1...6
	<b>W0</b> <sup>1)</sup>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100</b> <sup>1)</sup>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2</b> <sup>1)</sup>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+</b> <sup>2)</sup>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-</b> <sup>2)</sup>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2</b> <sup>2)</sup>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with <b>CType</b> = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with <b>CType</b> = 3-element control)	-9,99 ... 99,99
	<b>Xsh1</b> <sup>3)</sup>	Neutral zone (Xw > 0)	0,0 ... 1000 [%]
	<b>Xsh2</b> <sup>3)</sup>	Neutral zone (Xw < 0)	0,0 ... 1000 [%]
	<b>Y2</b>	Additional correcting variable	0 ... 100 [%]
	<b>Ymin</b> <sup>4)</sup>	Min. correcting variable limiting	0 ... 100 [%]
	<b>Ymax</b>	Max. correcting variable limiting	0 ... 100 [%]
	<b>Y0</b>	Correcting variable working point (start-up correcting variable)	0...100 [%]
	<b>YOptm</b>	Correcting variable during process at rest (not with PIDMA)	0...100 [%]
	<b>dYopt</b>	Self-tuning step change height	5...100 [%]
	<b>Xp1 (1...6)</b> <sup>3)5)</sup>	Proportional band 1	0,1 ... 999,9 [%]
	<b>Xp2 (1...6)</b> <sup>3)5)</sup>	Proportional band 2	0,1 ... 999,9 [%]
	<b>Tn1 (1...6)</b> <sup>5)</sup>	Integral action time	0 ... 999 999 [s]
	<b>Tv1 (1...6)</b> <sup>5)</sup>	Derivative action time	0 ... 999 999 [s]
	<b>Tp1 (1...6)</b> <sup>5)</sup>	Cycle time heating	0,4 ... 999,9 [s]
	<b>Tp2 (1...6)</b> <sup>5)</sup>	Cycle time cooling	0,4 ... 999,9 [s]
	<b>Title</b>	Title of controller page (only display)	16 characters
	<b>Einh.X</b>	Unit of the process value (only display)	6 characters
	<b>W int</b>	Internal after transmission of the engineering to KS98	-29 999 ... 999 999
	<b>A/H</b>	Status of controller after transmission of the engineering to KS98	0 oder 1

<sup>1)</sup> The values are specified in the process value unit - e.g. [°C, °F, bar, %, etc.]

<sup>2)</sup> The rate of change must be specified in units/minute (e.g. °C/min) → see gradient control page 243.

<sup>3)</sup> % specifications are related to measuring range **Xn100** - **Xn0**. There is no relation to values **W0** and **W100**.

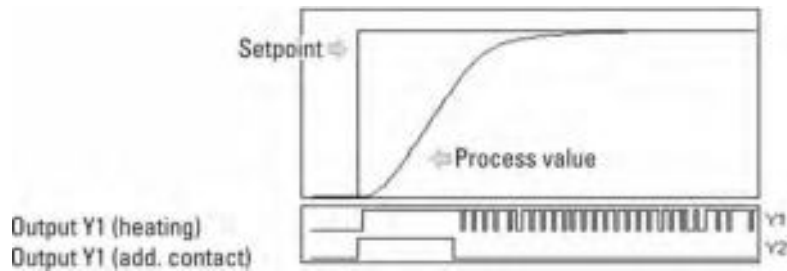
<sup>4)</sup> As default, value Ymin is set to 0. In this case, output Y1 cannot switch!

<sup>5)</sup> **(1...6)** refers to the six parameter sets of CONTR+ (e.g. Xp1, Xp2, Xp3...Xp6).



## $\Delta/Y/off$

The principle is identical to the control behaviour of a 2-point controller with additional contact.

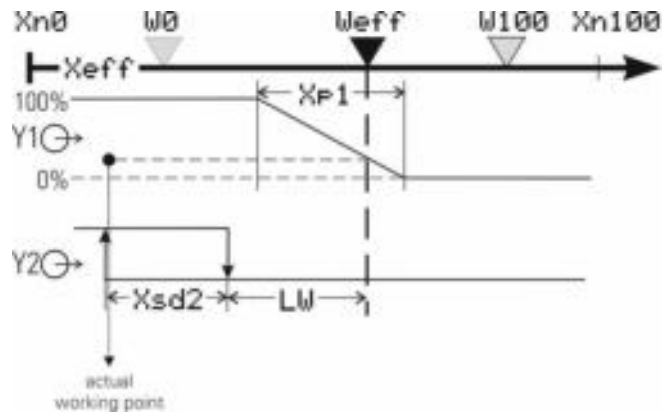


Output Y2 is used for switchover of the connected circuit between "  $\Delta$  " and "Y". Output Y1 switches the heating energy on and off.

E.g. for temperature control with electrical heating (inverse operation) or cooling (direct operation).

Cycle time  $T_{p1}$  must be adjusted as follows:  $T_{p1} \leq 0,25 \cdot T_u$ , With higher  $T_{p1}$ , oscillations must be expected.  $T_{p1}$  corresponds to the minimum cycle time (time in seconds) at 50 % duty cycle

*Static operating principle  
of the  $\Delta/Y/off$  function*



PD action ( $T_n = 0$   $\Delta$  switched off  $T_n = \infty$ )

The working point is in the middle of the proportional band  $X_{p1}$  at 50 % duty cycle.

For keeping the control variable constant, a defined amount of energy dependent of is required. This causes a permanent control deviation, which increases with higher  $X_{p1}$ .

DPID action

By means of the I action, line-out without permanent control deviation is possible.

The static characteristic of a two-point controller is identical to the one of a continuous controller. The difference is that a duty cycle instead of a linearly variable current signal is output (relay contact, logic signal 0/20mA or control output 0/24V).

Working point  $Y_0$  and cycle time  $T_{p1}$  of the cycle at 50% are adjustable.

The shortest switch-on or off time is 100ms.

Configuration	Effective controller parameters with $\Delta$ / Y / off controller		
CFunc = 2-P + addition	<b>Popt</b>	Parameter set for self-tuning (only with <b>CONTR+</b> )	1...6
	<b>W0</b> <sup>1)</sup>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100</b> <sup>1)</sup>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2</b> <sup>1)</sup>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+</b> <sup>2)</sup>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-</b> <sup>2)</sup>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2</b> <sup>2)</sup>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with <b>CType</b> = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with <b>CType</b> = 3-element control)	-9,99 ... 99,99
	<b>LW</b> <sup>1)</sup>	Trigger point separation of additional contact	-29 999 ... 999 999
		OFF = the additional contact is switched off	-32 000 = OFF
	<b>Xsd2</b> <sup>1)</sup>	Switching difference of additional contact	0,1 ... 999 999
	<b>Y2</b>	Additional correcting variable	0 ... 100 [%]
	<b>Ymin</b>	Min. correcting variable limiting	0 ... 100 [%]
	<b>Ymax</b>	Max. correcting variable limiting	0 ... 100 [%]
	<b>Y0</b>	Correcting variable working point (start-up correcting variable)	0...100 [%]
	<b>YOptm</b>	Correcting variable during process at rest (not with PIDMA)	0...100 [%]
	<b>dYopt</b>	Self-tuning step change height	5...100 [%]
	<b>Xp1 (1...6)</b> <sup>3)4)</sup>	Proportional band 1	0,1 ... 999,9 [%]
	<b>Tn1 (1...6)</b> <sup>4)</sup>	Integral action time	0 ... 999 999 [s]
	<b>Tv1 (1...6)</b> <sup>4)</sup>	Derivative action time	0 ... 999 999 [s]
	<b>Tp1 (1...6)</b> <sup>4)</sup>	Cycle time heating	0,4 ... 999,9 [s]
	<b>Title1</b>	Title of controller page (only display)	16 characters
	<b>Einh.X</b>	Unit of the process value (only display)	6 characters
	<b>Wint</b> <sup>1)</sup>	Internal after transmission of the engineering to KS98	-29 999 ... 999 999
	<b>↕</b>	Status of controller after transmission of the engineering to KS98	0 oder 1

<sup>1)</sup> The values are specified in the process value unit - e.g. [°C, °F, bar, %, etc.]

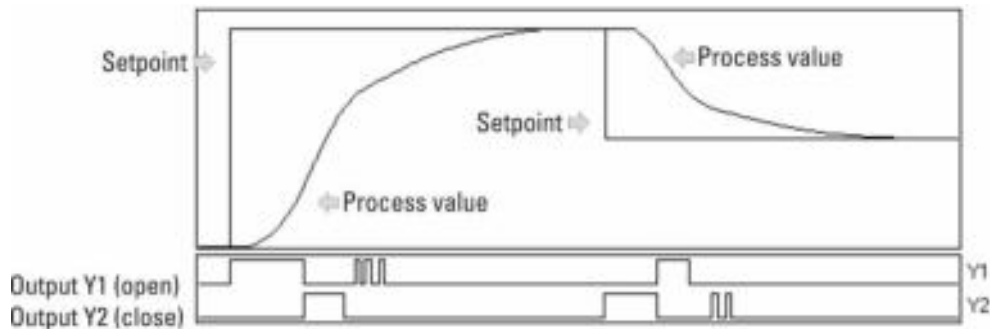
<sup>2)</sup> The rate of change must be specified in units/minute (e.g. °C/min) → see gradient control page 243.

<sup>3)</sup> % specifications are related to measuring range **Xn100** - **Xn0**. There is no relation to values **W0** and **W100**.

<sup>4)</sup> **(1...6)** refers to the six parameter sets of CONTR+ (e.g. Xp1, Xp2, Xp3...Xp6).

### Three-point stepping controller

Switching controller for control of a valve (e.g. temperature control by means of motorized valve and gas-air mixture)

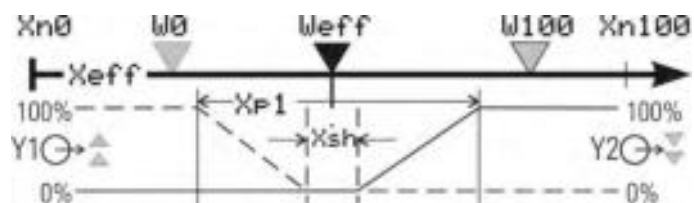


1. Open valve; → outputs Y1 = 1, Y2 = 0
2. Don't move valve; → outputs Y1 = 0, Y2 = 0
3. Close valve; → outputs Y1 = 0, Y2 = 1

To validate the adjusted  $X_p$  for the actuator response time, response time  $T_m$  must be adjusted. The smallest positioning step is 100ms.

With PMA controllers the position feedback has no influence on the PID-behaviour!

*Static operating principle of a three-point stepping controller*



Adjusting the neutral zone

Neutral zone  $X_{sh}$  can be increased in case of excessive output switching. However, note that an increase the neutral zone will reduce the control sensitivity.

For this reason, we recommend optimizing switching frequency (actuator wear) and control sensitivity.

Three-point stepping controllers can be operated with or without position feedback  $Y_p$ .

**Stepping** 3-point stepping controller

**Step+YP** 3-point stepping controller with position feedback

whereby  $Y_p$  is not used for control. The static characteristic of a three-point stepping controller is shown in the figure above.

The hysteresis shown in this diagram is practically unimportant, but can be calculated from the adjustable min. pulse length  $T_{puls} \geq 100ms$  ( $T_s$  = scanning sequence 100/200/400/800 ms).

$$X_{sh} = \left( \frac{T_{puls}}{2} - 0,5 \cdot T_s \right) \cdot \frac{X_p}{T_m}$$

With **Tpu1s** switched off, the shortest positioning step **Tpu1s** is dependent of **Tm**, **Xsh** and **Xp**. By varying **Xsh**, a required min. pulse length **Tpu1s** can be realized:

$$X_{sh} = 12,5 \cdot X_p \cdot \frac{T_{puls}}{T_m} - 0,75$$

Configuration	Effective controller parameters with three-point stepping controller		
CFunc = Step Step Yp	<b>Popt</b>	Parameter set for self-tuning (only with <b>CONTR+</b> )	1...6
	<b>W0</b> <sup>1)</sup>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100</b> <sup>1)</sup>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2</b> <sup>1)</sup>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+</b> <sup>2)</sup>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-</b> <sup>2)</sup>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2</b> <sup>2)</sup>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with <b>CType</b> = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with <b>CType</b> = 3-element control)	-9,99 ... 99,99
	<b>Xsh</b> <sup>3)</sup>	Trigger point separation	0,2 ... 20 [%]
	<b>Tpu1s</b>	Min. positioning step time	0,1 ... 2 [s]
	<b>Tm</b>	Actuator response time	5 ... 999 999 [s]
	<b>Y2</b>	Additional positioning value (only with step Yp → with position feedback)	0 ... 100 [%]
	<b>YOptm</b>	Positioning value during process at rest (not with PIDMA)	0...100 [%]
	<b>dYopt</b>	Self-tuning step height	5...100 [%]
	<b>Xp1 (1...6)</b> <sup>3)4)</sup>	Proportional band 1	0,1 ... 999,9 [%]
	<b>Tn1 (1...6)</b> <sup>4)</sup>	Integral action time	0 ... 999 999 [s]
	<b>Tv1 (1...6)</b> <sup>4)</sup>	Derivative action time	0 ... 999 999 [s]
	<b>Titel</b>	Title of controller page (only display)	16 characters
	<b>Einh.X</b>	Unit of the process value (only display)	6 characters
	<b>W int</b> <sup>1)</sup>	Internal after transmission of the engineering to KS98	-29 999 ... 999 999
	<b>A/H</b>	Status of controller after transmission of the engineering to KS98	0 oder 1

<sup>1)</sup> The values must be specified in the process value unit - e.g. [°C, °F, bar, %, etc.]

<sup>2)</sup> The rate of change must be specified in units/minute (e.g. °C/min) → see gradient control page 243.

<sup>3)</sup> % specifications are related to measuring range **Xn100** - **Xn0**. There is no relationship to values **W0** and **W100**.

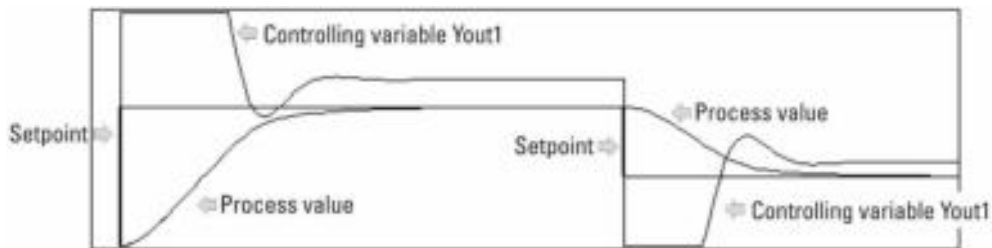
<sup>4)</sup> **(1...6)** refers to the six parameter sets of CONTR+ (e.g. Xp1, Xp2, Xp3...Xp6).

## Continuous controller / Split range

Continuous controller

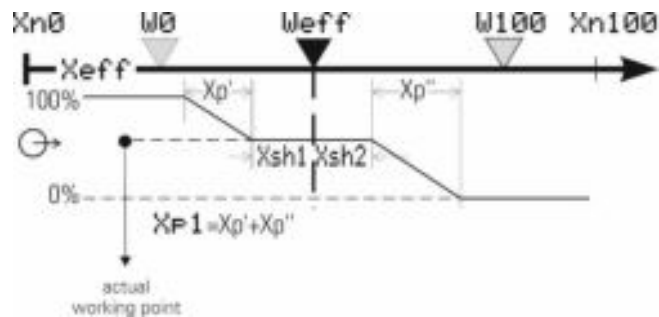
An analog value is provided as correcting variable by output **Yout1**, e.g. temperature control with electrical heating and thyristor power regulator.

A continuous controller in 'split-range' operation is comparable with a three-point controller. The neutral zone is also separately adjustable.



Within limits  $X_{sh1}$  and  $X_{sh2}$ , the control deviation for calculation of the controller reaction is set to zero. A pure P controller does not change the correcting variable within these limits. A PID controller has a dynamic behaviour which has not always decayed, also when reaching "control deviation = 0". Both D and I action can still have an effect according to the characteristic determined by  $T_v$  due to a preceding disturbance or a setpoint step change. This effect can be strong enough to cause the control deviation to leave range  $X_{sh1}/X_{sh2}$ , i.e. the P action is activated again for reaching the neutral zone.

*Operating principle of the proportional part of the continuous controller*



Selection from the following continuous controllers is possible:

- 1.) **CFunc** = continuous → continuous controller
- 2.) **CFunc** = splitRang → continuous controller with split-range operation  
The continuous output is split on outputs Yout1 and Yout2.
- 3.) **CFunc** = continuous Yp → continuous controller with position feedback.  
The actually flowing positioning current can be displayed via input Yp. Yp is not included in the control operation.

Configuration	Effective controller parameters of a continuous controller		
CFunc = continuous SplitRange	<b>Popt</b>	Parameter set for self-tuning (only with <b>CONTR+</b> )	1...6
	<b>W0</b> <sup>1)</sup>	Lower setpoint limit for Weff	-29 999 ... 999 999
	<b>W100</b> <sup>1)</sup>	Upper setpoint limit for Weff	-29 999 ... 999 999
	<b>W2</b> <sup>1)</sup>	Additional setpoint	-29 999 ... 999 999
	<b>Grw+</b> <sup>2)</sup>	Setpoint gradient plus	off / 0,001 ... 999 999
	<b>Grw-</b> <sup>2)</sup>	Setpoint gradient minus	off / 0,001 ... 999 999
	<b>Grw2</b> <sup>2)</sup>	Setpoint gradient for W2	off / 0,001 ... 999 999
	<b>N0</b>	Zero offset (only effective with <b>CType</b> = ratio controller)	-29 999 ... 999 999
	<b>a</b>	Factor a (only effective with <b>CType</b> = 3-element control)	-9,99 ... 99,99
	<b>Xsh1</b> <sup>3)</sup>	Neutral zone (Xw > 0)	0,0 ... 1000 [%]
	<b>Xsh2</b> <sup>3)</sup>	Neutral zone (Xw < 0)	0,0 ... 1000 [%]
	<b>Y2</b>	Additional correcting variable	0 ... 100 [%]
	<b>Ymin</b>	Min. correcting variable limiting	0 ... 100 [%]
	<b>Ymax</b>	Max. correcting variable limiting	0 ... 100 [%]
	<b>Y0</b>	Correcting variable working point (start-up correcting variable)	0...100 [%]
	<b>YOptm</b>	Correcting variable during process at rest (not with PIDMA)	0...100 [%]
	<b>dYopt</b>	Self-tuning step change height	5...100 [%]
	<b>Xp1 (1...6)</b> <sup>3)4)</sup>	Proportional band 1	0,1 ... 999,9 [%]
	<b>Xp2 (1...6)</b> <sup>4)</sup>	Proportional band 2 (only with continuous controller split range)	0 ... 999 999 [s]
	<b>Tn1 (1...6)</b> <sup>4)</sup>	Integral action time	0 ... 999 999 [s]
	<b>Tv1 (1...6)</b> <sup>4)</sup>	Derivative action time	0,4 ... 999,9 [s]
	<b>Title1</b>	Title of controller page (only display)	16 characters
	<b>Einh.X</b>	Unit of the process value (only display)	6 characters
	<b>Wint</b> <sup>1)</sup>	Internal after transmission of the engineering to KS98	-29 999 ... 999 999
	<b>A/H</b>	Status of controller after transmission of the engineering to KS98	0 or 1

<sup>1)</sup> The values must be specified in the process value unit, e.g. [°C, °F, bar, %, etc.]

<sup>2)</sup> The rate of change must be specified in units/minute e.g. °C/min). → see gradient control page 243.

<sup>3)</sup> % specifications are related to measuring range **Xn100** - **Xn0**. There is no relationship to values **W0** and **W100**.

<sup>4)</sup> **(1...6)** refers to the six parameter sets of CONTR+ (e.g. Xp1, Xp2, Xp3...Xp6).

### 3.16.5. Controller characteristics (CONTR und CONTR+)

#### Process characteristics

Characteristics are determined automatically by the controller during self-tuning and converted into control parameters. In exceptional cases, however, manual determination of these process characteristics may be necessary. For this, the response of process variable  $x$  after a step change of correcting variable  $y$  can be used (see Figure below).

Usually, it is not possible to plot the complete response curve (0 to 100 %), as the process must be kept within certain limits.

The maximum rate of increase  $v_{max}$  can be determined from the values  $T_g$  and  $x_{max}$  (complete step response) or  $\Delta t$  and  $\Delta x$  (partial step response).

$$K = \frac{V_{max}}{X_h} \cdot T_u \cdot 100\%$$

$y$  = correcting variable

$Y_h$  = control range

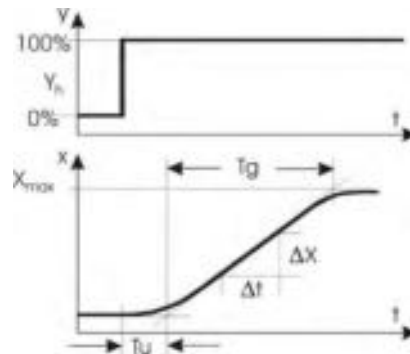
$T_u$  = delay time (s)

$T_g$  = recovery time (s)

$V_{max} = \frac{x_{max}}{T_g} = \frac{\Delta x}{\Delta t} \triangleq$  max. rate of increase of process value

$X_{max}$  = maximum process value

$X_h$  = control range  $\triangleq \times 100 - \times 0$



#### Characteristic values of the controllers

Generally, quick line-out to the without oscillation is required.

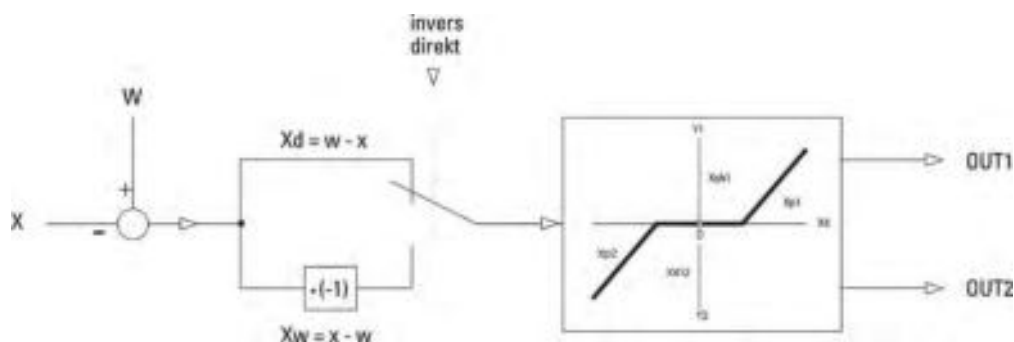
Dependent of process, different control actions should be used:

- Processes with good controllability ( $K < 10\%$ ) can be controlled by means of PD controllers,
- Processes with medium controllability ( $K = 10...22\%$ ) can be controlled with PID controllers and
- Processes with bad controllability ( $K > 22\%$ ) can be controlled with PI controllers.

The control parameters can be determined from the calculated values of delay time  $T_u$ , max. rate of increase  $v_{max}$ , control range  $X_h$  and characteristic value  $K$  according to the formulas. For more exact adjustment, see the hints given in the table of parameter adjustment effects. Increase  $X_p$  if line-out oscillates.

Formulas				Parameter adjustment effects				
Action	Xp[%]	Tv[s]	Tn[s]	Setting	Control	Line-out of disturbances	Start-up behaviour	
(D)PID	1,7 K	2 Tu	2 Tu	Xp	higher	Increased damping	Slower line-out	Slower reduction pf energy
PD	0,5 K	Tu	$\infty = 0000$		lower	Reduced damping	Faster line-out	Faster reduction of energy
PI	2,6 K	0	6 Tu	Tv	higher	Reduced damping	Faster line-out	Earlier reduction of energy
P	K	0	$\infty = 0000$		lower	Increased damping	Slower line-out	Later reduction of energy
3- point stepping controller PID				Tn	higher	Increased damping	Slower line-out	Slower reduction pf energy
	1,7 K	Tu	2 Tu		lower	Reduced damping	Faster line-out	Faster reduction of energy

Direct / inverse switchover is always possible in configuration parameter **CMode** (action). *Folgendes Bild zeigt das Prinzip.*



### 3.16.6. Empirical optimization CONTR / CONTR+

With missing distance data can empirically be optimized by means of the self-optimization or in manual attempts. With the attempts for empirical optimization the following is to be considered:

- It is to be guaranteed that correcting variable and controlled variable take never forbidden values!!!
- The conditions for the attempts should be always alike, in order to win comparable statements.
- The test sequence must be oriented at the goal of the optimization: Leadership- or interference behaviour.
- The operating point of the controller must be alike with the attempts.

The control parameters are to be set as follows with their first use:

Xp maximum: to the largest adjustable value,

Tv relatively large: time max., which the controlled system needs for distinct beginning of the reaction.

Tn large: time max., which the controlled system needs for the entire process.

The time requirement for an empirical optimization is large. In order to achieve an useful result in relatively short time, the following is recommended for appropriate procedure results:

- ① Adjust  $T_n = T_v = 0$  and Xp largest possible (p-controller). The Xp is reduced from attempt to attempt, as long as the control is sufficiently stable. If it becomes too unstable, then the Xp is to be increased and next step is ②.
- ② Measure lasting offset: If it is sufficiently small, then the optimization is successfully terminated (P). If it is too large, then the controlled system is better regulated with PD (adjust Tv relatively large and next step is ③).
- ③ Reduce Xp from attempt to attempt, as long as the control is sufficiently stable. If it becomes too unstable, then the next step is ④.
- ④ Tv is to be made smaller and determined whether the regulation can be sufficiently stabilized again. If, then it the next step is ③, if not, then Xp is to increase and the next step is ⑤.
- ⑤ Determine whether with the procedures ③ and ④ the Xp was substantially made smaller. If, then the next step is ⑥, if not, then the controlled system better is pi-regulated (Tv set to 0 and the next step is ⑦).
- ⑥ Measure lasting offset. If it is sufficiently small, then the optimization is successfully terminated (PD). If it is too large, then the controlled system is better PID-regulated (no longer change Xp and Tv and the next step is ⑦).
- ⑦ Tn is adjusted largely and reduced from attempt to attempt, as long as the control is sufficiently stable. If it becomes too unstable, then the Xp is to be increased, and the optimization is successfully terminated (PID or pi).

- ❗ For the controlled variable (process value X) the empirical optimization is substantially improved with a writer (or trend function of the engineering tool) in time requirement and quality, and evaluation of the test results is clearly simplified.
- ❗ The procedure mentioned can only with restrictions be generalized and does not lead to a clear improvement of the behavior with all controlled systems.
- ❗ Changes of the operating point (Y0), the switching point distance (Xsh) and the lasting switching period (Tp1 and Tp2) lead to results, which can be better or worse. With 3 - Point - step controller's TM must be adjusted to the real running time of the connected actuator.



### 3.16.7. Self-tuning → controller adaptation to the process

For determination of optimal parameters a self-optimization can be accomplished.

This is applicable for controlled systems with reconciliation and none dominating dead time and  $K \leq 30\%$ . After start by the operator the controller initiates an adaptation cycle in order to determine the line characteristic values  $T_u$  and  $V_{max}$ . It calculates by it the control parameters for fast, overshoot-free correction to the ( $X_{p1}$ ,  $X_{p2}$ ,  $T_n$ ,  $T_v$ ,  $T_{p1}$ ,  $T_{p2}$ , depending upon kind of controller).

#### Preparation

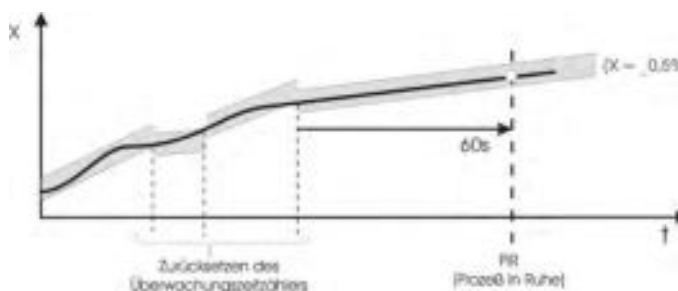
- Set the desired controller behaviour.
 

P- controller:	<b><math>T_n = 0.0</math></b>	<b><math>T_v = 0.0</math></b>
PD- controller:	<b><math>T_n = 0.0</math></b>	<b><math>T_v &gt; 0.0</math></b>
PI- controller:	<b><math>T_n &gt; 0.0</math></b>	<b><math>T_v = 0.0</math></b>
PID- controller:	<b><math>T_n &gt; 0.0</math></b>	<b><math>T_v &gt; 0.0</math></b>
- The parameters  **$T_n$**  and/or  **$T_v$**  can be switched off, by being adjusted to the value = **0,0**. Thus they do not participate in the self-optimization.
- With the automatic controller CONTR+ is to be selected, which parameter set is to be optimized ( **$P_{Opt}=1...6$** ).
- Configure conditions for process at rest ( **$O_{Cond}$** )  
The condition designates, when "Prozess at rest" is to be recognized ( **$PIR\_H$** ):  
 **$grad=0$ ,  $grad<0/>0$  or  $grad\neq 0$**  (→ also see process at rest, page 225)
- The correcting variable  **$Y_{Optm}$**  is to be specified. This is, in automatic running, the correcting variable, which is output with the start of self-optimization in order to generate "Prozess at rest".
- The step of the correcting variable  **$dY_{Opt}$**  is to be specified.  **$dY_{Opt}$**  is the amount the correcting variable jumps, from the initial value  **$Y_{Optm}$**  and/or in manual operation from the original correcting variable.
- Consider the reserve (→ also see reserve, page 226)

#### Process-at-rest' monitoring (PiR):

'Process-at-rest' monitoring is done at any time. The process is at rest, when the process value is within a tolerance band of  $\pm\Delta X = 0,5\%$  during more than 60 seconds.

When the process value is out of this band, the monitoring timeout counter is reset to zero. With detection of PiR e.g. during control operation and output of a widely deviating stable correcting variable  **$Y_{Optm}$**  at self-tuning start, waiting until the full PiR time has elapsed is required.



With extended monitoring, monitoring is for a constantly varying instead of a constant process variable!

Configuration word  **$O_{Cond}$**  can be used to determine 'Process at rest'detection. One of the following modes can be selected:

$grad(X) = 0$ :	Process at rest is detected, when x is constant.
$grad(X) <0/>0$ :	Process at rest is detected when x decreases constantly with a controller with inverse output action. Process at rest is detected, when x increases constantly with a controller with direct output action.
$grad(X) \neq 0$ :	Process at rest is detected with constantly changing x. In this case, continuation of this constant change over the duration of identification must be ensured.

### Setpointreserve:

In order to make self-tuning possible, the separation between and process value before the output step change must be higher than 10 % of  $W0...W100$ . The reserve is provided either automatically by reducing the correcting variable during the PiRphase, or by changing the or the process value manually (manual mode).

With inverse controllers, the must exceed the process value by at least the reserve. With direct controllers, the must be smaller than the process value by at least the reserve. This is necessary, as the is a limit which should not be exceeded during self-tuning.

### Self-tuning start

Self-tuning can be started or stopped from automatic or from manual mode on the self-tuning page ( → see "Start from automatic/manual mode" page 226, 226).

Select the self-tuning page by marking the two arrows **▶▶** followed by configuration.

Select function Stat: **OFF/OK** (inverse display) and confirm it by **■**

**Stat: OFF/OK** blinks and can be switched over to **Stat: Start** by pressing **▲**.

Press key **■** to start the self-tuning attempt. adjustment is always possible

### Self-tuning cancelling

A self-tuning attempt can always be cancelled.

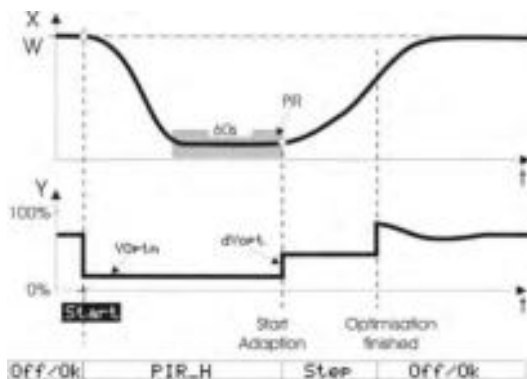
Self-tuning can always be stopped by pressing function key **■** on the controller front panel, provided that key **■** was not disabled (1-signal on input **oplook**).

Moreover, cancellation is possible from the self-tuning page of the required controller. For this, press key **▲** on the self-tuning page to select the **Stat:** line (inverse display), press **■**, **Stat:** line blinks. Press **▲** until **Stat:** Stop blinks. Press **■**, the self-tuning attempt was stopped and the controller continues operating in automatic mode.

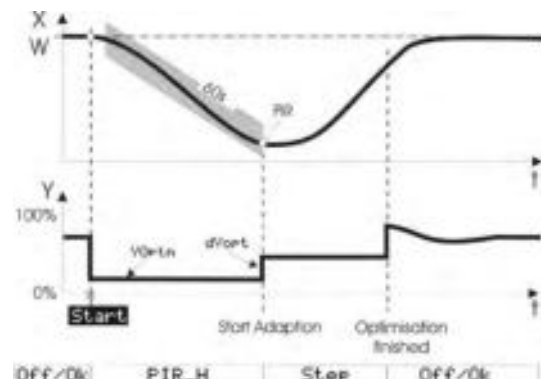
### Start from automatic mode:

After self-tuning start, stable correcting variable **YOptm** is output. When 'Process at rest' (PiR) is detected and a sufficient reserve (→ see page 226) is provided, the correcting variable is changed by output step **dYOpt** (increased with indirect controller, increased with direct controller). The self-tuning procedure is realized using the varying process value.

*self-tuning [grad(x)=0]*



*self-tuning [grad(x)<0]*



After successful self-tuning, the controller goes to the automatic mode and controls the using the new parameters. Parameter **Ores** indicates the self-tuning result (→ see page 47).

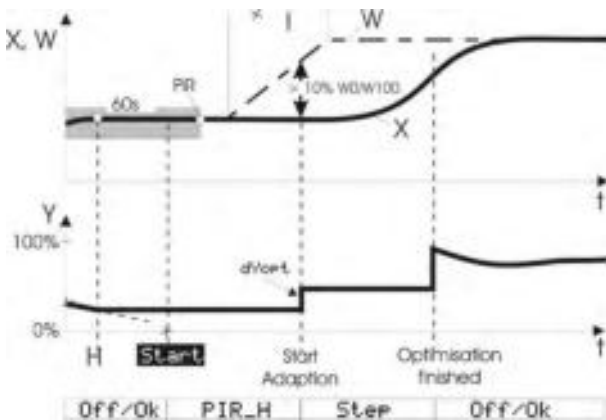
**!** When finishing self-tuning with an error (**Ada\_Err**), the stable correcting variable is output, until self-tuning is finished by the operator via system menu, front-panel key **■**, or via interface.

### Start from manual mode

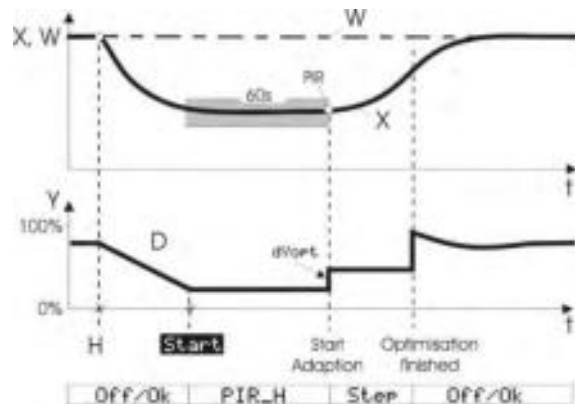
To start self-tuning from manual mode, switch the controller to manual. During transition to manual mode, the correcting variable output last is taken over as manual correcting variable. At self-tuning start, this correcting variable is taken over and output as temporary stable correcting variable. Like in automatic mode, the can be changed at any time.

With 'Process at rest' (PiR) detection and a sufficient reserve (→ see page 226), the correcting variable is changed by the correcting variable step **dYOpt** (increased with indirect controller, decreased with direct controller). 'Process at Rest' (PiR) can be reached at starting time, i.e. the normal 60 s waiting time can be avoided. The self-tuning procedure is realized using the varying process value.

*Start by rising the setpoint*



*Start by lowering the setpoint*



After successful self-tuning, the controller goes to the automatic mode and controls the using the new parameters. Parameter **Ores** indicates the self-tuning results (→ see page 47).

⚠ When finishing self-tuning with an error (**Ada\_Err**), the stable correcting variable is output, until self-tuning is finished by the operator via system menu, front-panel key **F**, or via interface.

### Self-tuning procedure with heating:

(2-point, 3-point stepping, continuous controller)

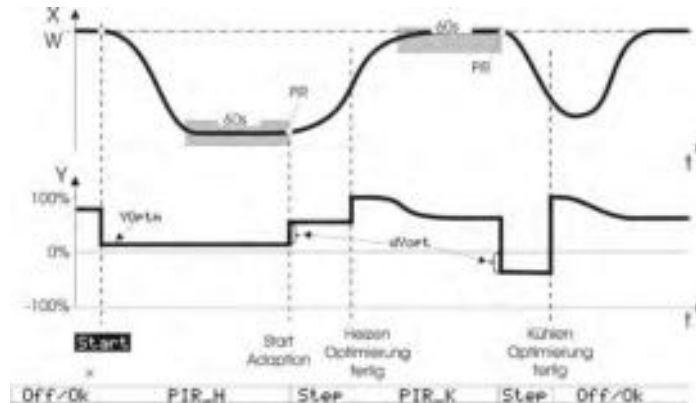
After reaching 'Process in rest', the process is stimulated by means of an output step change and the process response is used to determine  $Tu1$  and  $Vmax1$  at the step response reversal point, if possible.

### Self-tuning procedure with heating and cooling processes:

(3-point / split-range controller)

Self-tuning starts as with a "heating" process. After self-tuning end, the controller settings based on the calculated parameters are made. This is followed by line-out at the pre-defined , until PIR is reached again. Subsequently, a step to cooling is made to determine the "cooling" parameters, in order to determine  $Tu_2$  and  $V_{max2}$  using the step response. Based on these characteristics, the controller settings for the cooling process are made.

When cancelling the cooling attempt, the parameters for "heating" are also taken over for cooling. No error message (**Ada\_Err**) is indicated.



**i** With 3-point stepping controllers, the motor actuator is closed first after starting and opening to  $Y_{Optm}$  will occur only then. This calibration procedure (**Stat: Abg1 .**) is not shown in the figures.

**i** For maintaining a safe process condition, monitoring for an exceeded is done continuously.

**!** During self-tuning, the control' function is switched off! I.e.:  $Y_{pid}$  is within the limits of  $Y_{min}$  and  $Y_{max}$ .

**!** With **Δ/Δ/Aus** controllers, self-tuning is using the function **Δ**, i.e.  $Y_2 = 0$ .

### Controlled adaptation

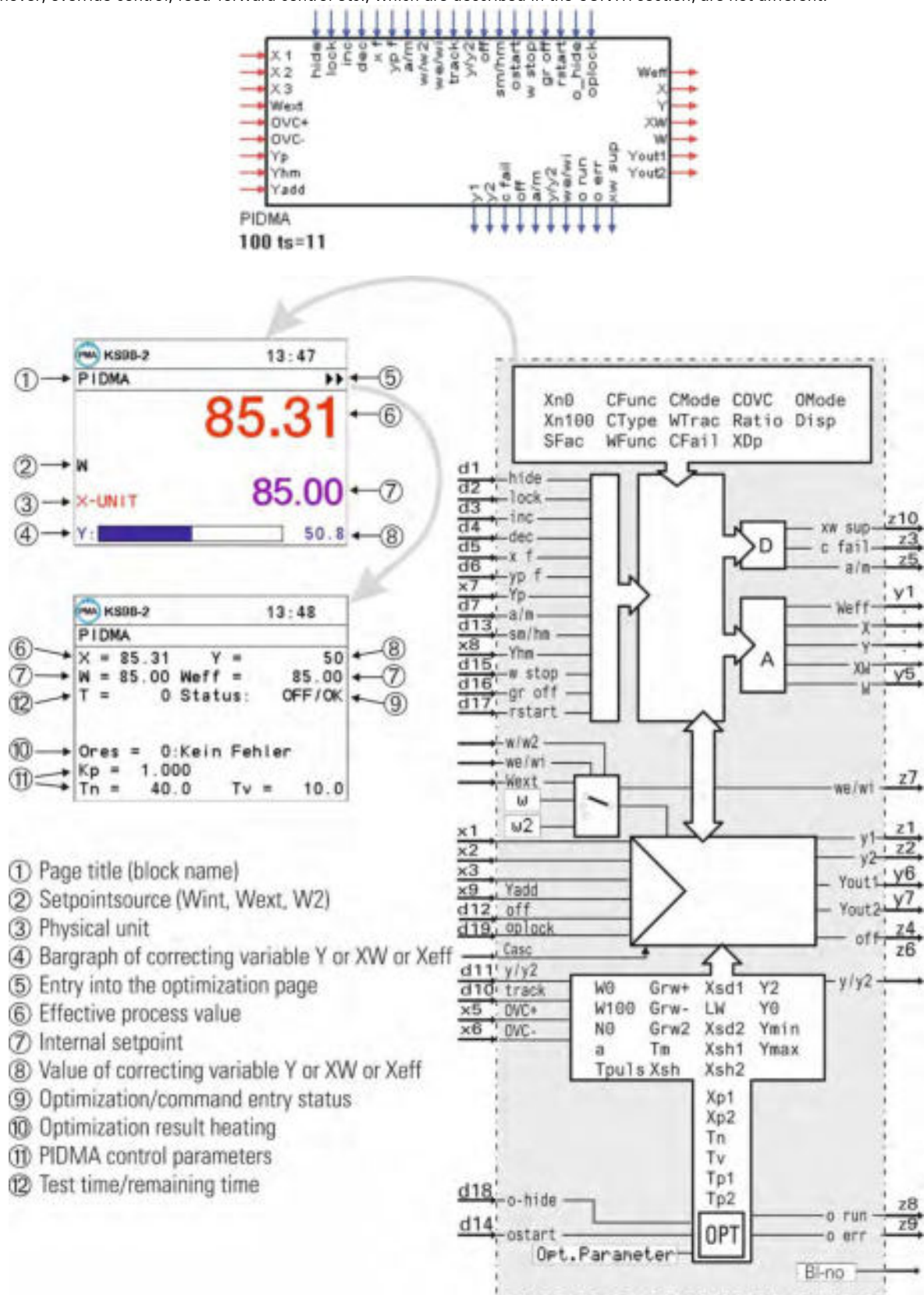
For defined applications, adaptation of the control parameter set to the current process condition is purposeful. For this, the Contr+ is provided with 6 control parameter sets, which can be selected via analog input **ParNo**.

### Signification of self-tuning messages ORes1/ORes2

**i** Unless control is as required despite self-tuning, proceed additionally as described in section "Empirical self-tuning" (→ page 224, Hints for self-tuning, Hints for adjustment), and follow the hints given on further parameters. Meaning of the messages see page 47.

### 3.16.8. PIDMA (Control function with particular self-tuning behaviour (Nr. 93))

The controller block PIDMA is particularly suitable for difficult processes (with delay time, or of higher order). The difference between PIDMA and the CONTR block is only in the PID controller kernel (self-tuning and control algorithm). The additional functions ramp, switchover, override control, feed-forward control etc., which are described in the CONTR section, are not different.



The most important differences compared to controller functions CONTR and CONTR+ are:

- Integrated, front-panel operated optimization method like PMA tune.  
This function permits optimization also for difficult processes with  $T_g/T_u < 3$  without engineering tool and laptop, which was not possible with previous PMA (and competitor) controllers.
- Parallel controller structure unlike all other PMA controllers with "serial structure".
- Distinction of "behaviour" and "disturbance behaviour" by adjustable factors, which can be used for individual attenuation of P (proportional action) and D (differential action) control effect on changes.
- The adjustable derivative gain VD of the D action, which is adjusted and matched to the process dynamics automatically by self-tuning. Purposeful values for VD are within 2...10, whereby all previous PMA controllers are fixed to VD=4 (empirical value for serial structure).

Using a PIDMA control block is purposeful where conventional PMA self-tuning methods are not satisfactory. PIDMA should not be used for processes where PMA self-tuning has always been and still is unrivalled:

- Processes with a ratio  $T_g/T_u > 10$
- (2nd order processes; with 2 [...3] energy storing elements!).


Roughly, these are applications in the plastics processing industry (extrusion, ...), where no improvements related to quick line-out without overshoot are possible (unless a "robust" controller design with stable results also with variable process dynamics and non-linearities is required)!

With classic thermal processes (ovens of all types, dryers, ...) air conditioning, level, flow, etc., however, a considerable number of difficult cases require the expenditure of many hours spent with hotline support or even at the site, in order to make a plant operate properly.

The various types of control behaviour are not explained in detail in this section, because there are no differences compared to controller blocks CONTR and CONTR+ (see from page 211).





Only the additional parameters explained at the beginning of chapter "PIDMA control characteristics" have to be described.

The difference between split-range and 3-point behaviour is that PIDMA does not provide parameter distinction of heating and cooling.

 PIDMA does not permit the adjustment of the signaller control behaviour.

## Inputs/outputs for PIDMA

## Digital inputs:

<b>hide</b>	Display suppression (with <b>hide</b> = 1 the operating page is not displayed).
<b>lock</b>	Adjustment locking (with <b>lock</b> = 1 the values are not adjustable. Function keys   are not active).
<b>inc</b>	Increment for manual adjustment
<b>dec</b>	Decrement for manual adjustment
<b>x f</b>	Sensor error x1...x3
<b>yp f</b>	Sensor error Yp
<b>a/m</b>	0 = automatic      1 = manual
<b>w/w2</b>	0 = int./ext. setpoint   1 = W2
<b>we/wi</b>	0 = external setpoint   1 = internal setpoint
<b>track</b>	0 = tracking function off; 1 = tracking function on → page 244; 245)
<b>y/y2</b>	0 = output value Y, 1 = output value Y2
<b>off</b>	0 = controller switched on 1 = controller switched off
<b>sm/hm</b>	0 = soft manual 1 = hard manual
<b>ostart</b>	1 = self-tuning start → page 46 ff)
<b>w stop</b>	1 = effective setpoint freeze (can e.g. be used for bandwidth monitoring)
<b>gr off</b>	1 = setpoint gradient suppression
<b>rstart</b>	1 = Start the setpoint ramp → the setpoint makes a step change towards the process value and goes to the adjusted process value according to GRW+ (GRW-). The rising flank (0→1) is evaluated.
<b>o-hide</b>	1 = self-tuning page display suppression
<b>oplock</b>	Blockage of key  (with oplock = 1, switchover to manual by means of key  is not possible).

## Digital outputs:

<b>y1</b>	Status of switching output Y1; 0 = off 1 = on
<b>y2</b>	Status of switching output Y2; 0 = off 1 = on
<b>c fail</b>	1 = controller in error handling
<b>off</b>	0 = controller switched off 1 = controller switched on
<b>a/m</b>	0 = automatic 1 = manual
<b>y/y2</b>	0 = output value Y 1 = output value Y2
<b>we/wi</b>	0 = external setpoint, 1 = internal setpoint
<b>o run</b>	1 = Self-tuning running
<b>o err</b>	1 = Error during self-tuning
<b>xw sup</b>	Alarm suppression with setpoint change → ALARM

## Analog inputs:

<b>x1</b>	Main variable x1
<b>x2</b>	Auxiliary variable x2      e.g. for ratio control
<b>x3</b>	Auxiliary variable x3      e.g. for 3-element control
<b>Wext</b>	External setpoint
<b>OVC+</b>	Override control + → page 250 ff)
<b>OVC-</b>	Override Control - → page 250 ff)
<b>Yp</b>	Position feedback
<b>Yhm</b>	Output with hard manual
<b>Yadd</b>	Cascade input for controller cascade
<b>Casc</b>	Kaskadier-Eingang für Reglerkaskade

## Analog outputs:

<b>Weff</b>	Effective setpoint
<b>X</b>	Effective process value
<b>Y</b>	Effective output value
<b>XW</b>	Control deviation
<b>W</b>	Internal setpoint
<b>Yout1</b>	Output value yout1 (heating)
<b>Yout2</b>	Correcting variable yout2 (cooling; only with continuous controller with split-range behaviour → <b>CFunc</b> = splitRange)
<b>B1-no</b>	Own block number

### 3.16.9. Parameter and configuration for PIDMA

#### Parameters for PIDMA

Parameter	Description	Range	Default	Unit
<b>PType</b>	Process type (with compensation or integral)	comp. integral	comp.	<b>comp.</b>
<b>Drift</b>	Drift compensation	switched off switched on	off	<b>off</b>
<b>CSpeed</b>	Control dynamics	slow normal fast	normal	<b>normal</b>
<b>W_Block</b>	switchover disable function	Switchover via front-panel operation disabled. Wext ↔ Wint switchover is disabled W ↔ W2 switchover is disabled All switchover functions are enabled	←	
<b>W0</b>	Min. setpoint limit (Weff)	0: Block All 1: Block We 2: Block W2 3: None	0	<b>0</b>
<b>W100</b>	Max. setpoint limit (Weff)	-29999...999999	100	<b>100</b>
<b>W2</b>	Additional setpoint	-29999...999999	100	<b>100</b>
<b>Grw+<sup>2)</sup></b>	Setpoint gradient plus	unit/min	0,001...999999	off
<b>Grw-<sup>2)</sup></b>	Setpoint gradient minus	unit/min	0,001...999999	off
<b>Grw2<sup>2)</sup></b>	Setpoint gradient for W2	unit/min	0,001...999999	off
<b>N0</b>	Zero offset with ratio control	-29999...999999	0	<b>0</b>
<b>A</b>	Factor a for 3-element control	-9,99...99,99	1	<b>1</b>
<b>Xsh<sup>1)</sup></b>	Trigger point separation (stepping controller)	0,2...20,0%	0,2	<b>0,2</b>
<b>Tpause</b>	Min. pause time (3-point stepping controller)	0,1...999999[s]	0,1	<b>0,1</b>
<b>Tpuls</b>	Min. positioning step time (3-point stepping controller)	0,1...2,0[s]	0,3	<b>0,3</b>
<b>Tm</b>	Actuator response time (3-point stepping controller)	5...999999 [s]	30	<b>30</b>
<b>thron</b>	Threshold for OPEN and CLOSE (stepping controller) currently not active	0,2...100%	0,2	<b>0,2</b>
<b>throff</b>	Threshold for OPEN and CLOSE (stepping controller) currently not active	0,2...100%	0,2	<b>0,2</b>
<b>Y2</b>	Additional correcting value (not with 3-point stepping controller)	-105,0...105,0[%]	0	<b>0</b>
<b>Ymin</b>	Min. output limiting (not with 3-point stepping controller)	-105,0...105,0[%]	0	<b>0</b>
<b>Ymax</b>	Max. output limiting (not with 3-point stepping controller)	-105,0...105,0[%]	100	<b>100</b>
<b>Y0</b>	Controller working point (not with 3-point stepping controller)	-105,0...105,0[%]	0	<b>0</b>
<b>dYopt<sup>3)</sup></b>	Self-tuning step height	5...100[%]	100	<b>100</b>
<b>Xlimit</b>	Switch-off point for correcting variable step (process value change)	0,5...999999	1	<b>1</b>
<b>Tdrift</b>	Time window for drift determination (process value)	0...999999	30	<b>30</b>
<b>Tnoise</b>	Time window for noise determination (process value)	0...999999	30	<b>30</b>
<b>Kp</b>	Control amplification	0,1...999,9[%]	100	<b>100</b>
<b>Tn 1</b>	Integral time (Tn = 0 → I action is not effective)	0,0...999999[s]	10	<b>10</b>
<b>Tv 1</b>	Derivative time (Tv = 0 → D action is not effective)	0,0...999999[s]	10	<b>10</b>
<b>Tp1 1</b>	Cycle time heating (3-point controller)	0,4...999,9[s]	5	<b>5</b>
<b>Tp2 1</b>	Cycle time cooling (3-point controller)	0,4...999,9[s]	5	<b>5</b>
<b>VD</b>	Derivative gain (Td/T1)	1...999999	4	<b>4</b>
<b>bW_p</b>	Setpoint weighting factor proportional action	0...1	1	<b>1</b>
<b>cW_d</b>	Setpoint weighting factor derivative action	0...1	0	<b>1</b>
<b>Tsat</b>	Time constant for integral action in Y limiting (anti-reset wind-up)	1...999999	50	<b>50</b>
<b>Xsh</b>	Neutral zone for integral action	1...999999	0	<b>0</b>

<sup>1)</sup> Neutral zone  $x_{sn}$  with 3-point stepping controllers is dependent on  $T_{puls}$ ,  $T_m$  and  $x_{p1}$  (→ V. Hints for self-tuning)..

<sup>2)</sup> for gradient control → page 243

<sup>3)</sup> for self-tuning → page 46 ff



## Configuration data PIDMA

Configuration	Description	Values	Default
<b>XCFunc</b>	Regelverhalten:	2-point controller	<b>2-point</b>
		Stetiger Regler	<b>3-point</b>
		3-point controller (heating/cooling switching)	<b>Cont</b> ←
		3-point controller (heat.continuous/cool.switching)	<b>Cont/swi</b>
		3-point controller (heat.switching/cool.continuous)	<b>Swi/Cont</b>
		3-point stepping controller	<b>Stepping</b>
		3-point-stepping controller with pos. feedback Yp	<b>Step+Yp</b>
		Continuous controller with split-range operation	<b>splitRang</b>
<b>CType</b>	Controller type	Continuous controller with position feedback Yp	<b>stetig Yp</b>
		Standard controller	<b>Standard</b> ←
		Ratio controller	<b>Ratio</b>
<b>WFunc</b>	Setpoint function	3-element controller	<b>3-elem.</b>
		Setpointcontrol	<b>setp</b> ←
		Setpoint/cascade control	<b>Sp/casc</b>
<b>CMode</b>	Output action	Inverse output action	<b>Invers</b> ←
		Direct output action	<b>Direct</b>
		Neutral	<b>Neutral</b>
<b>CFai1</b>	Behaviour with sensor error	Ypid = Ymin (0%)	<b>Ymin</b> ←
		Ypid = Ymax (100%)	<b>Ymax</b>
		Ypid = Y2 (no adjustment via front panel)	<b>Y2</b>
		Ypid = Y2 (automatic) oder Yman (manual operation)	<b>Y2/Yman</b>
		No override control	<b>off</b> ←
<b>COVC</b>	Output limiting	Override-Control +	<b>OVC+</b>
		Override-Control -	<b>OVC-</b>
		Override-Control + / -	<b>OVC+/OVC-</b>
<b>WTrac</b>	Tracking des int. Sollwertes	Int. tracking	<b>off</b> ←
		Setpoint tracking	<b>SP-track</b>
		Process value tracking	<b>PV-track</b>
<b>Ratio</b>	Ratio controller function:	$(x1 + N0) / x2$	<b>Typ 1</b> ←
		$(x1 + N0) / (x1 + x2)$	<b>Typ 2</b>
		$(x2 - x1 + N0) / x2$	<b>Typ 3</b>
<b>XDp</b>	Digits behind the decimal point (process value)	0...3	0
<b>Disp</b>	Contents of bargraph line:	Output variable	<b>Y</b> ←
		Control deviation	<b>XW</b>
		Xeff	<b>Xeff</b>
<b>Xn0</b>	Span start	-29999 ... 999999	0
<b>Xn100</b>	Span end	-29999 ... 999999	100
<b>SFac</b>	Factor stoichiom. ratio	0,01 ... 99,99	1,00

### 3.16.10. Controller characteristics and self-tuning with PIDMA

As opposed to CONTR and CONTR+, the PIDMA includes a modified parallel controller structure, which is taken into account in the following additional parameters.

#### Additional parameters for PIDMA

Parameter	Description	Values
<b>PType</b>	Process type (a-priori information)	1: with compensation 2: without A.(integral)
<b>Drift</b>	Drift compensation of the actual value at the beginning of the self-tuning	0: off 1: on
<b>CSpeed</b>	Required control loop dynamics	1: slow 2: normal 3: fast
<b>Tpause</b>	Minimum positioning step time (stepping controller)	0,1...999999[s]
<b>thron</b>	Switch-on threshold for OPEN and CLOSE (stepping controller)	0,2...100%
<b>throff</b>	Switch-off threshold for OPEN and CLOSE (stepping controller)	0,2...100%
<b>Xlimit</b>	Switch-off point for output step change (process value change)	0,5...999999
<b>Tdrift</b>	Time window for process value drift determination	0...999999
<b>Tnoise</b>	Time window for process value noise determination	0...999999
<b>Kp</b>	Control gain (replaces Xp1;/Xp2 of CONTR)	0,001...999,9[%]
<b>VD</b>	Derivative gain (Td/T1)	1...999999
<b>bW_p</b>	Setpoint weighting factor of proportional action	0...1
<b>cW_d</b>	Setpoint weighting factor of D action	0...1
<b>Tsat</b>	Time constant for I action in Y limiting (anti-reset wind-up)	1...999999
<b>Xsh</b>	Neutrale Zone, in dem der I-Teil festgehalten wird	0 ... 999999

#### Three-point stepping (Yp):

Tpause, thron and throff complete the effective parameters for stepping motor control. Tpause permits adjustment of the minimum pause in addition to limiting of the minimum pulse via Tpuls.

#### thronoff:

The initially provided parameters for motor stepping controller structure in PIDMA are ineffective in the present realization. Only parameter xsh can be used for stabilizing the positioning activities.

#### Xsh:

Xsh can be used to influence the motor actuator switching frequency and fine setting. Xsh determines the dead band of the control deviation in the main controller. The controller I action is stopped within this zone.

#### Integrated position controller:

With 3-point stepping controller with position feedback (step + Yp), the PIDMA function block comprises two controllers: the main controller controls the process value and provides a required actuator position to an integrated position controller. By means of position feedback, this position controller ensures that the actuator position is as required.

#### Self-tuning:

PType, Drift, Cspeed, Xlimit, Tdrift and Tnoise complete parameter dYopt which is also effective with CONTR. These parameters define the conditions during self-tuning.

Ptype determines, if the process is without compensation (the new process value after a correcting variable pulse is higher, e.g. level of a container without outlet or well- insulated furnace).

An even decrease or increase of the process value before self-tuning can be detected by means of drift monitoring and taken into account when self-tuning is done for the next time

CSpeed can be used to determine if, during subsequent operation, the controller should reach the setpoint quickly, with a slight overshoot, or slowly with gentle approach to the setpoint. Using CSpeed, the parameter can be switched over also after self-tuning, provided that the controller parameters were not changed manually.

After self-tuning start, timer Tdrift for process value drift detection and timer Tnoise for noise detection (variations independent of the correcting variable) are started. Dependent on process, the timers should be long enough to permit detection of an interference-independent drift and multiple "ups" and "downs" of interference effects.

After elapse of these timers, the actual correcting variable is increased by dYopt. When the process value has increased by more than Xlimit under consideration of drift and noise, the correcting variable is reset to the initial value. However, self-tuning is completed only, when the process value has decreased to nearly half of the initial value after exceeding the maximum. During decrease after the correcting variable pulse, the estimated remaining time until self-tuning end is displayed continuously. After completion of self-tuning, the determined parameters K, Ti and Td are displayed on the self-tuning page, taken over into the function block together with parameters VD, BW\_p and CW\_d and activated for the running process.

### Control parameters of PIDMA:

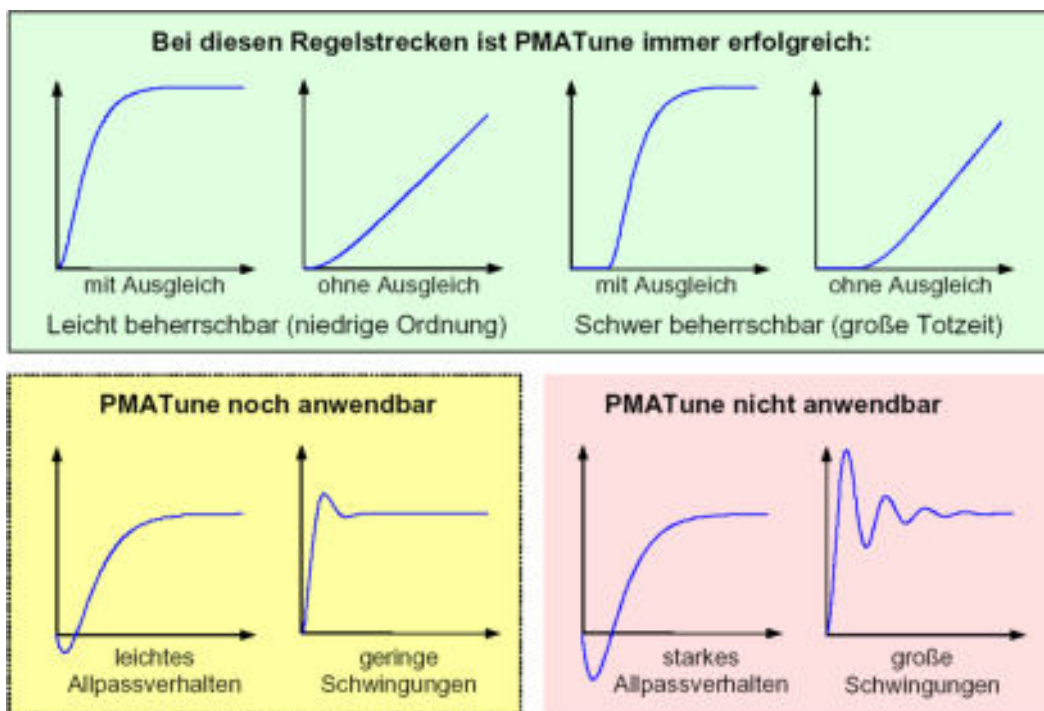
Unlike CONTR, PIDMA does not have separate parameters for heating and cooling. Parameter K which is valid for both ranges determines the control gain of a parallel controller structure.

Further parameters permit independent weighting of individual controller components:

VD: In addition to the control gain, the derivative gain ( $T_d/T_1$ ) permits an increase or reduction of the derivative action.

BW\_p: Setpoint weighting factor of proportional action.

CW\_d: Setpoint weighting factor of derivative action.



Parameters BW\_p and CW\_d can reduce the effect of a setpoint change on the controller reaction. I.e. different controller behaviour after setpoint changes (control behaviour) or process value changes (disturbance behaviour) can be selected. A factor within 0 and 1 can be applied to the setpoint effect.

In the course of dynamic process control, the control algorithm can temporarily determine values below 0 or above 100 for the correcting variable. If necessary, however, these values can be reset to the limits by means of accelerated integral behaviour (Tsats).

Tsat time constant for integral action in Y limiting (anti-reset wind-up).

## Self-tuning → controller adaptation to the process (PIDMA)

Self-tuning can be started to determine the optimum parameters for a process. The function is applicable for the following processes.

### Preparation

Adjusting the required control behaviour.

- P-controller:  $T_n = 0.0$        $T_v = 0.0$
- PD-controller:  $T_n = 0.0$        $T_v > 0.0$
- PI-controller:  $T_n > 0.0$        $T_v = 0.0$
- PID-controller:  $T_n > 0.0$        $T_v > 0.0$

Parameters **Tn** or **Tv** can be switched off by setting = **0.0**. I.e. these parameters do not participate in self-tuning.


- Correcting variable step change **dYopt** must be determined. This is the value by which the correcting variable changes from the actual value. The step change can be positive or negative.
- Xlimit must be determined. It should be set to roughly half of the expected process value change.

### 'Process at rest' monitoring:


The PIDMA does not provide monitoring for the rest condition. The commissioning engineer is responsible for selection of a suitable start time. Optimum results are only achieved, if the process is lined out, i.e. all dynamic actions have decayed. Only in few cases in which parameter determination is impossible due to decaying dynamics the algorithm provides a "restart" error message.

### Self-tuning start



Self-tuning can be started and stopped from automatic or manual operation on the self-tuning page.

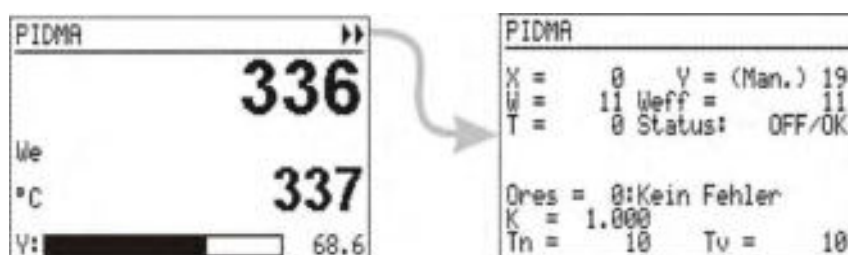
Select the self-tuning page by marking and confirming the two arrows. Select function **Stat: OFF/OK** (inverse display) and confirm it with .




**Stat: OFF/OK** **blinks** and can be switched over to **Stat: Start** by pressing .

Pressing key  starts the self-tuning attempt. The setpoint can be changed at any time. However, this is not necessary as opposed to CONTR. A change after starting from automatic mode would even cause faulty process evaluation.

### Self-tuning cancelation

Self-tuning can be stopped at any time by pressing key manual/automatic  on the controller front, provided that key  was not disabled (1-signal on input **oplook**).



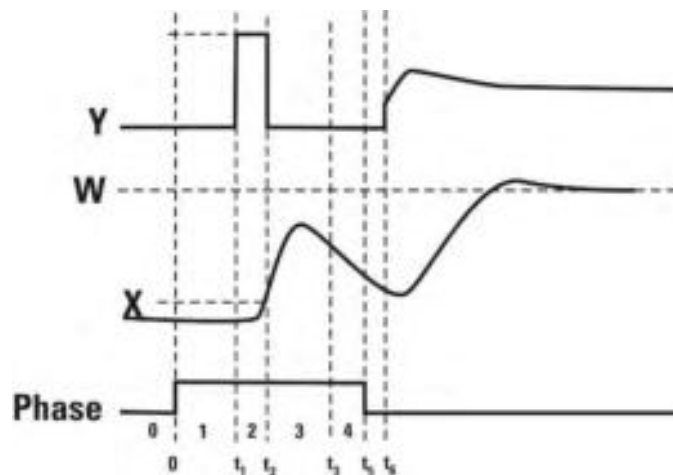
Moreover, cancelation is possible on the self-tuning page of the required controller. For this purpose, select the Stat: -line (inverse display) on the self-tuning page, press , Stat: line blinks. Press  until Stat: Stop blinks. Press , the self-tuning attempt is stopped and the controller continues operating in automatic mode.


**Start in manual mode or in automatic mode:**

Basically, the PIDMA self-tuning algorithm does not distinguish between these two start conditions. In both cases, the operator must ensure that the process conditions are stable. In automatic mode, however, the PIDMA works with the non-optimized parameters until start of the correcting variable pulse. This means that, in the majority of cases, better stability of process conditions, i.e. better self-tuning results, are possible in manual mode. When changing to manual mode, the correcting variable output last is taken over as manual correcting variable and used during estimation.

After self-tuning start, the estimation timer for drift detection and noise detection is started at first. In the second phase, the correcting variable is changed by correcting variable step change **dYOpt**. When the process value has changed by more than **Xlimit**, the correcting variable is reset to the original value. In the third phase, the PIDMA waits for the maximum value of the increasing process value. Subsequently, it monitors the decaying process value in the fourth phase. During this time, an estimation of the remaining time until completion of the self-tuning attempt is output.

After a successful self-tuning attempt, the controller goes to the automatic mode and controls the setpoint using the new parameters. Parameter **Ores** indicates the result of self-tuning completion (→ see page 48)




⚠ When self-tuning is finished with an error (**Ada\_Err**), the stable correcting variable is output, until self-tuning is finished by the operator via the system menu, front panel key , or via the interface.

**Self-tuning procedure with heating and cooling processes:**

((3-point / split-range controller and mixed controllers)

With PIDMA, different control gains for heating and cooling cannot be specified. For this reason, the 2-step self-tuning attempt is omitted.

**Signification of self-tuning messages ORes**

 After successful self-tuning, parameter **CSpeed** can be used to increase or reduce the attenuation, when self-tuning was done with the setting for **CSpeed** = "normal". Moreover, only an increase or reduction of **Kp** should be considered. After manual change of the controller parameters, the **CSpeed** switch-over stops being effective

### 3.16.11. Controller applications:

The following chapter describes the common characteristics of controller block connection, which are independent of the CONTR and PIDMA controller kernel, such as and correcting variable switchover and limiting as well as process value pre-processing.

#### Controller front-panel operation

##### Controls on the controller page

Multilingual operation is not provided for the controller operating pages. Texts such as title and unit should be language-independent, if necessary.

- ① Page title (block name)
- ② Setpoint source (Wint, Wext, W2)
- ③ Physical unit
- ④ Bargraph of correcting variable Y or XW or Xeff
- ⑤ Entry into the self-tuning page
- ⑥ Effective process value
- ⑦ Controller setpoint
- ⑧ Value of correcting variable Y or XW or Xeff
- ⑨ Self-tuning/command entry status
- ⑩ Self-tuning result heating
- ⑪ Process characteristics heating
- ⑫ Self-tuning result cooling
- ⑬ Process characteristics cooling

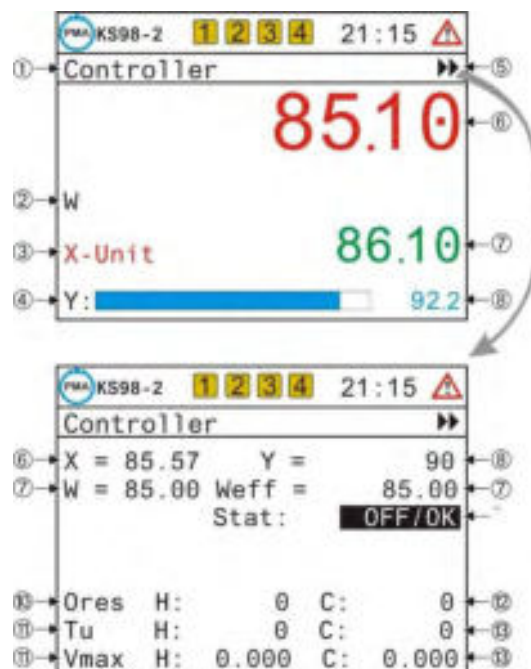


See also:

Chapter "Operation" on page 30

Chapter "Operating pages" on page 39

Chapter "Controller" on page 44

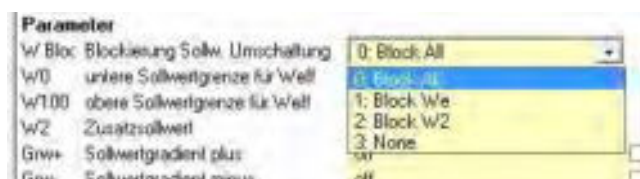


##### Switchover disabling

In many applications, switchover via the front panel should be not possible. Unwanted, accidental process interventions should be prevented by all means.

For these cases, switching over via front panel operation may be disabled. This is done by parameter W Block, which is intended for blocking individual or all switchover operations purposefully.

In default setting, all switchover operations are blocked and the switchover field for front-panel operation cannot be selected..



Switchover to Wext is blocked by configuration Spfunc = Set-point.



When  $W < W2$  switchover is blocked and  $Wext < Wint$  switchover is not possible ( control), the field is skipped when selecting.

### Further status displays on the operating page

During optimization or with cascade control, further display elements may appear on the operating page.


#### Statuses during optimization

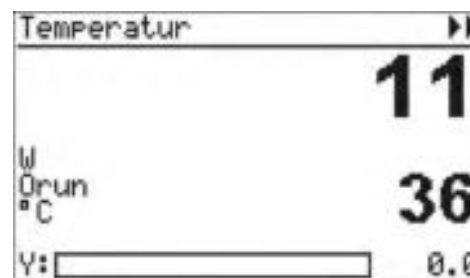
The optimization statuses are displayed with priority in the display field for manual operation.

Optimization running: display: **ORun**

Faulty optimization: display: **OErr**

When the optimization is finished with an error, the unit waits for acknowledgement by the operator.

By pressing key  twice or by input of the stop command on the optimization page, the controller returns to the initial status




### Cascade control operation

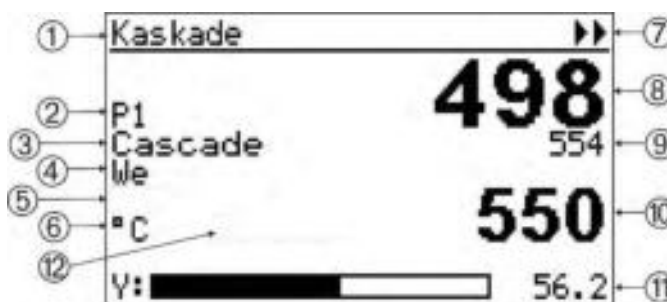
The cascade is one of most frequent controller structures with coupled control loops.

To facilitate construction and handling of these cascades, measures for connection and operation were taken at and in the controller blocks.

- A cascade consists of min. two controllers: a master controller the process value of which provides the main correcting variable and a slave controller on the process value of which the main variable is dependent.
- For building a cascade, the correcting variable output (Yout1) of the master may be wired to the input ((Wext) of the slave controller via a scaling function (SCAL).
- The slave controller is informed on the cascade by connecting the master block number output to the slave cascade input.

 The special operating functions of a controller cascade for master and slave are grouped on the common operating page of the slave controller.

- ① Title of the operating page
- ② Parameter set selection, if available
- ③ Cascade mode toggle field (open/close)
- ④ Master source (Wint, Wext, W2)
- ⑤ Display field for manual mode (otherwise empty)
- ⑥ Physical unit (master or slave)
- ⑦ Entry into self-tuning
- ⑧ Master process value
- ⑨ Slave process value
- ⑩ Setpoint (from master during automatic mode, from slave with open cascade)
- ⑪ Bargraph and display (Y from slave or X/XW from master)
- ⑫ Display of slave selection with open cascade (otherwise empty)

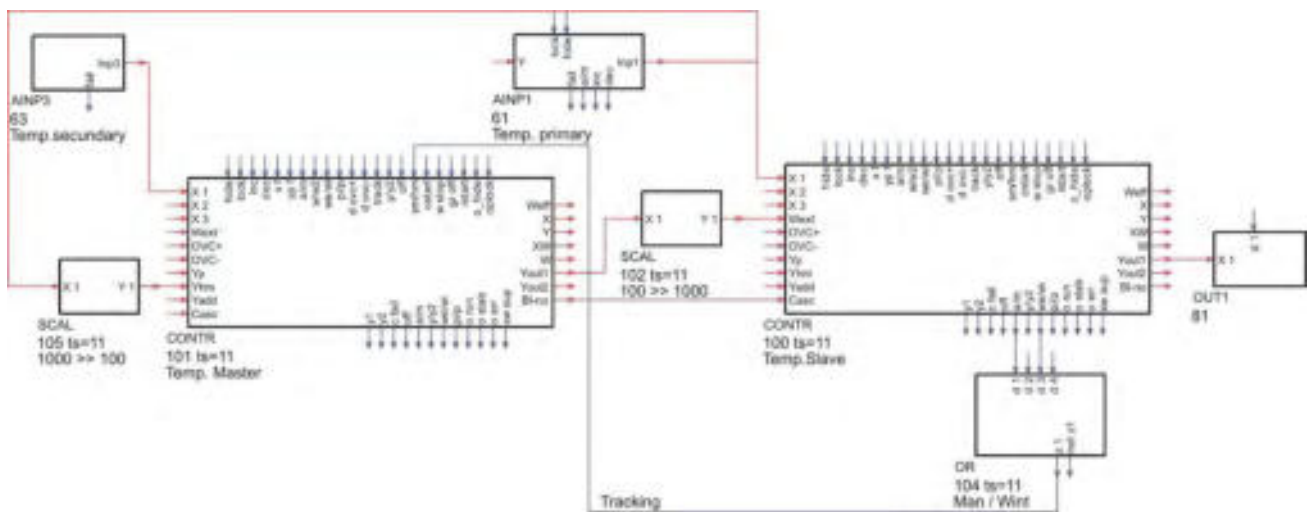





### Cascade operation is possible in the following operating statuses

(see also section operating pages on page 39):

- During automatic mode, the master variables (and process value) are the relevant variables in the process. The master controller is directly adjustable. The process value of the slave controller is only displayed. "Cascade" is displayed.
- Like every slave controller, the slave can be switched over to its internal setpoint or to W2 via control inputs. In this case, display is "Casc-open" as when opening via the operating field. Now the slave controller becomes the process-relevant variable and can be adjusted via the field (display "Slave" left in front of the setpoint). The master circuit process value is provided by the slave circuit rather than being controlled. Switching over between master or slave setpoint operation is possible at any time.
- During manual mode, the process is influenced directly by the slave controller correcting variable. The slave correcting variable is adjustable during manual mode. "Man" is displayed. During manual mode, or when the slave works with the internal setpoint or W2, the cascade is opened. The slave does not react on the master correcting variable any more. Suitable measures for master correcting variable tracking by the slave process value should be provided in the engineering, in order to ensure bumpless switchover to automatic mode (see following example).



Setpoint switchover disabling on the slave by means of parameter W Block prevents cascade opening via front panel operation! This parameter can be used to influence the W/We/W2 selection via the front panel..

 Closing of the cascade will cause automatic slave switchover to the external We. For identification of the data source, text "Slave" is displayed right beside the unit field when the cascade is open. A longer unit text can be overwritten partly.

 Now only the first 4 characters of the unit are visible.


In cascade mode, the information of the master is displayed in the fields Reference value, Reference source, Phys. Unit and X / XW- Bar graph. If the cascade is open ("slave" display), the information about the slave is displayed there.

### Optimizing the cascade

In a cascade, the slave controller and, subsequently, the master controller must be optimized. Self-tuning entry ►► on the cascade operating page always refers to the slave!

For master optimization, the master must be selected purposefully via the operating menu!

### Manual operation

Changing between automatic and manual mode is by pressing key . The manual mode influences only the slave controller. The master is concerned only indirectly.

The bargraph display switches over to slave variable Y. Adjustment of the correcting variable is via the value beside the bargraph

 The setpoint switchover and adjustment operations influence the master controller.



## The following rules apply to the bargraph display:

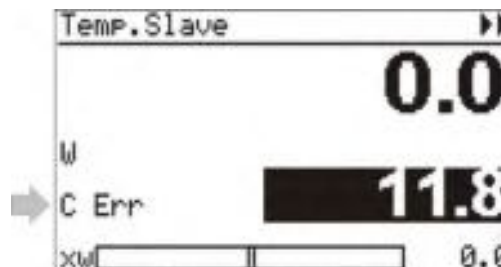
- When X or XW display is selected for the master bargraph display, the display value from the master controller is taken over.
- If Y display is selected, the slave bargraph value is used.

## Faulty wiring of a controller cascade

If an invalid cascade circuit was built up in the engineering, e.g. with the cascade input not connected to output BI-no of a master controller, the control function is not operable.

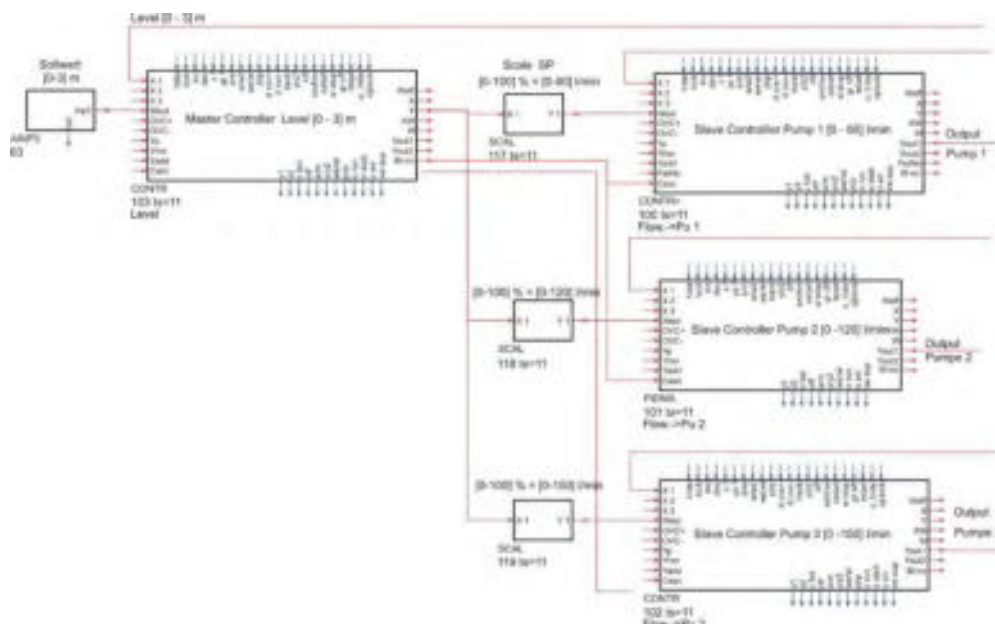
Error signalling is in the display field for the cascade:

Display: **C Err**



## Multiple cascade

A cascade control loop can be built up from a master controller with one or several slave controllers (see Fig. 76 : Level control example with three subordinated flow controllers). Cascade operation is from the slave controller page. Display of the master operating page should be suppressed (hide=1).



Operator interface activation for cascade control is automatic for controllers the Casc input of which is connected with the BI-no output of another controller.

In the above example, 3 flow controllers operate as slave controllers for level control. All three slave controllers offer the operator interface for level control with a separate operating page. Master tracking during manual mode of the slave as specified for simple cascade control in the example cannot be used without detailed considerations, because

1. two further cascade branches are still intact, when one controller is in manual mode
2. it is not clear, which process value should be used for tracking, when all controllers are in manual mode

### 3.16.12. Setpoint functions

#### Terminology

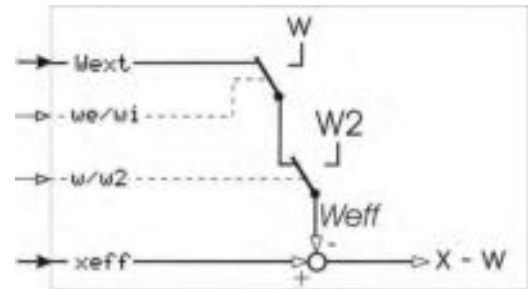
w	internal setpoint
w <sub>e</sub>	external setpoint
w <sub>2</sub>	second (internal) setpoint
W <sub>eff</sub>	effective setpoint
xw	control deviation (x-w r process value - )

#### General

Several possible setpoints are available. For the priorities, see the drawing shown opposite. "Safety setpoint" w<sub>2</sub> is given priority over the other setpoints. Switchover between setpoints is possible via interface or via the digital inputs of the controller block.

If gradient control was activated, a change will be made effectively continuously instead of being made effective by a step → gradient control page 243.

By activating digital input w stop, the instantaneously effective setpoint is maintained. In this case, neither a setpoint change nor switchover to another setpoint becomes effective.



#### Setpoint / setpoint/cascade

Configuration word **WFunc** can be used to select, if switch-over to the external setpoint (setpoint / cascade) is also possible in addition to the internal setpoint.

#### Setpoint

(**WFunc = Setpoint**) Setpoint control means that the setpoint is firmly predefined by the internal setpoint W.

#### Festwert/Folge

(**WFunc = Fest / Folg**) Setpoint / cascade control permits switchover between internal setpoint **W** and external setpoint **W<sub>e</sub>**. Switchover is via digital input w<sub>e</sub>/w<sub>i</sub> or via the interface.

Unless this input is connected, or if a 0 signal is applied, the external setpoint is used as effective setpoint. Unless digital input **w<sub>e</sub>/w<sub>i</sub>** as well as analog input **w<sub>ext</sub>** are connected, the controller invariably uses the internal setpoint.

#### W2 - safety setpoint

The second setpoint **W2** can be activated at any time and has highest priority. The change-over between internal setpoint and **W2** can be triggered via interface or the digital control input **w/w2**. In order to make the **W2** effective, on **w/w2** is a logic 1 to be attached. If the internal setpoint is to be active, a logic 0 must be given on **w<sub>e</sub>/w<sub>i</sub>**.

In the past **W2** was designated as "safety setpoint". Whether **W2** takes over safety functions or only a pre-defined starting position in certain process conditions, becomes determined only by the kind of the use and integration into an automation concept.

### External setpoint Wext

Switching between the internal setpoint (**w<sub>i</sub>**) and the external setpoint (**w<sub>e</sub>**) is possible only if the parameter **WFunc** is adjusted to **W/Casc**.

The change-over can be triggered via interface or the digital control input **w<sub>e</sub>/w<sub>i</sub>**. In order to make the internal setpoint effective, on **w<sub>e</sub>/w<sub>i</sub>** must be attached a logic 1. If the external setpoint is to be active, a logic 0 must be given on **w<sub>e</sub>/w<sub>i</sub>**. If the digital control input **w<sub>e</sub>/w<sub>i</sub>** is not wired, the external setpoint is effective.

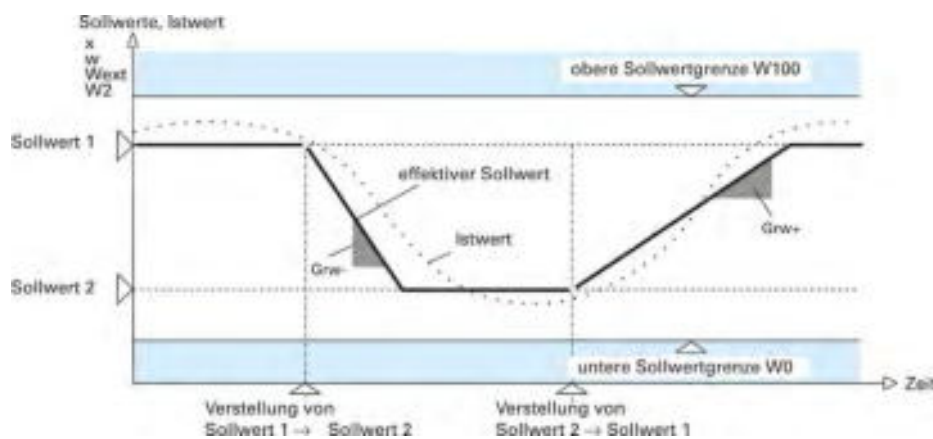
The internal setpoint **W** is evaluated with priority. If in a place (interface or the digital control input **w<sub>e</sub>/w<sub>i</sub>**) is switched to internal setpoint it is not possible to switch over to the external setpoint **W<sub>ext</sub>** from another source.

### Gradient control - setpoint changes with gradients

Normally, setpoint changes occur stepwisely. Unless this behaviour is required, a gradient can be set-up using parameters **Grw+** and **Grw-** or **Grw2**.


If these parameters are set, the setpoint changes are made bumplessly. With digital input 'gr\_off' not set, effective setpoint **W<sub>eff</sub>** runs linearly towards the changed setpoint (target value), whereby the slope is determined by gradients **Grw+** and **Grw-** adjustable at parameter setting level (r see Fig.: 78). For the second **W2**, an independent gradient **Grw2** was introduced, which is valid for both change directions and for **w r W2** switchover.

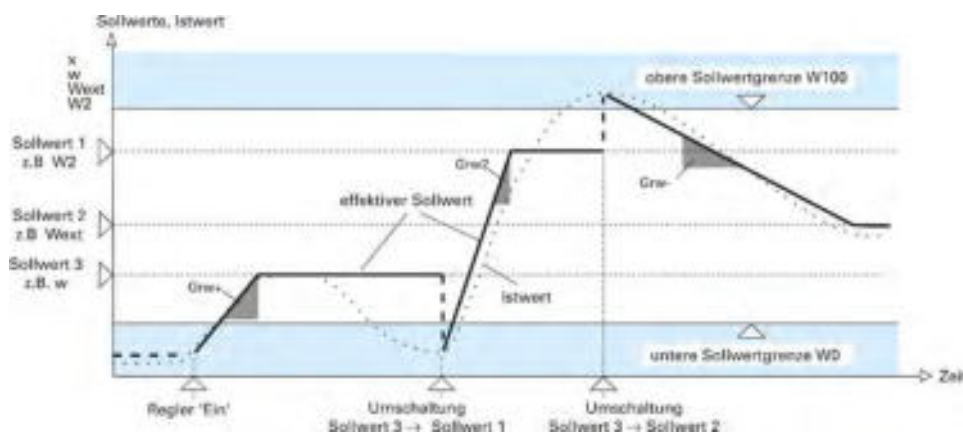
The gradient function is switched off when setting **Grw+** and **Grw-** or **Grw2** to "----" (engineering tool = off), or with digital input **Gr off** set to 1.



### Switch-over with gradients (W r W2, W r Wext, controller"on")

The new setpoint is linear started outgoing from the momentary process-value. The slope of the ramp is determined related to the direction of **Grw+**, **Grw-** and/or **Grw2**.

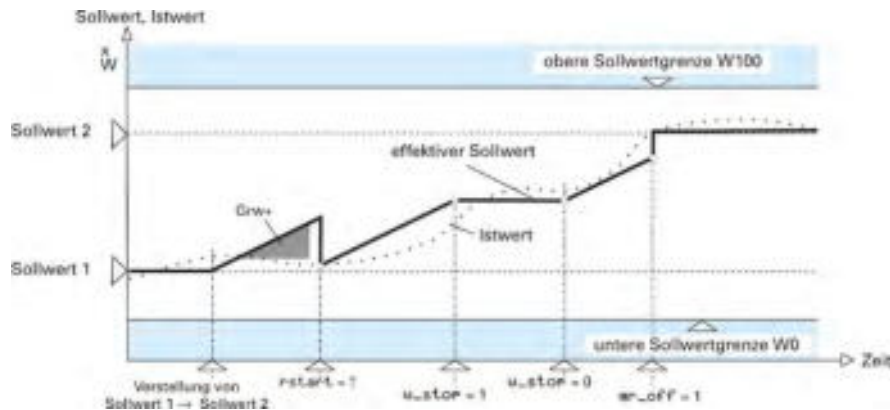
 This principle applies, even if the process-value is outside the adjustable range **W0/W100** at switch-over time (e.g. when starting).



### Controlling the setpoint

The digital input **rstart** reacts to a positive signal slope and sets the effective setpoint to the process value. The new goal is started on the basis of the controlled variable **xeff**. Such a ramp can only be started with activated gradient function (**Grw+**, **Grw-**, **Grw2** and digital input **gr\_off** not set).

The digital input **w\_stop** freezes the effective **Weff**, i.e., the effective setpoint is held to the current value, even if the effective setpoint straight approaches a new goal or a new goal is selected.

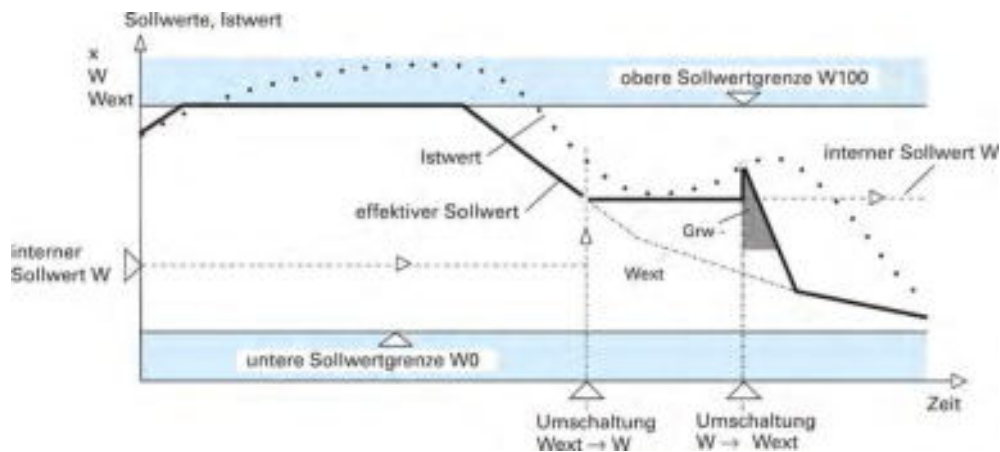


### Setpoint-Tracking

During the change-over of **Wext** → **W** it can come to unwanted jumps. To avoid these jumps setpoint-tracking can be applied. Setpoint-tracking takes care that in switch-over of **Wext** → **W**, the past **Wext** is taken over as int. **W**.

The digital input "track" enables Tracking. When switching back (**W** → **Wext**) **Wext** is started with the attitude of **Grw+** / **-**. With the configurationword is determined if the controller shall follow process value or setpoint-tracking **Wtrac**. Tracking can be activated via interface or by operating the switchover **Wext** → **W**. Tracking is evaluated with priority.

If a source (interface or digital input) is activating setpoint-tracking, switching by operation switching **Wext** → **W** from another source is not possible!



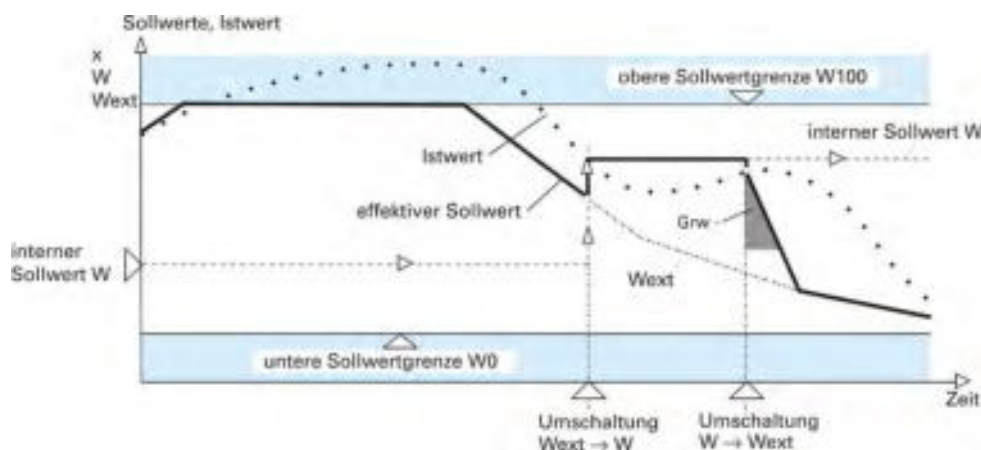
### Process-value tracking

It can occur that the is far distant from the momentary process-value (e.g. when starting a plant).

In order to prevent the jump developing here, the function process-value tracking can be used.

Process-value tracking causes a take-over of the process-value on the internal setpoint, when changing over **Wext** → **W**. When shifting (**W** back → **Wext**) **Wext** is started with the attitude of **Grw+ / -** (see → Figure below).

If the controller shall follow process value or setpoint-tracking is determined with the configurationword **Wtrac**. The digital input "track" enables Tracking. Tracking can be activated via interface or by operating the switchover **Wext** → **W**. Tracking is evaluated with priority.



### Setpoint and correcting variable behaviour after setpoint switch-over

After setpoint and correcting variable switch-over, control behaviour or start-up behaviour has priority. The PID characteristic must be partly suppressed. The previous history which is important for the integral action and especially for the derivative action is largely insignificant with setpoint change due to the new target setpoint.

#### Switch-over operations which might affect the control behaviour are:

1	Manual -> automatic	Switch-over from manual to automatic mode
2	Off -> start-up	Start-up after off-line (power failure/configuring)
3	$W_{old} \rightarrow W_{new}$	Setpoint change
4	$W \rightarrow W_2$	Switch-over to 2nd setpoint
5	$W_2 \rightarrow W$	Switch-over from 2nd setpoint to normal setpoint
6	$W_e \rightarrow W_i$ , without tracking	Switch-over from external to internal setpoint without tracking
7	$W_i \rightarrow W_e$	Switch-over from internal to external setpoint
8	$W_e \rightarrow W_i$ , with tracking	Switch-over from external to internal setpoint with tracking

The approach to a new setpoint may be affected by further parameters. Parameters **Grw+** (positive setpoint gradient), **Grw-** (negative setpoint gradient) and **Grw2** (setpoint gradient during the approach to **W2**) can be used for gradual approach to a new target setpoint via a ramp function.

Unless a gradient is defined (**Grw** = off), approach to the new setpoint starts with a step change at the previous setpoint or at the actual process value.

To influence the correcting variable when switching over, any after-effect of the derivative action is eliminated internally or the integral action is adapted to avoid correcting variable bumps.

The following table gives a survey of the controller switch-over behaviour implemented from operating version 8.

Switch-over	Without gradient function	With gradient function
<b>Man -&gt; Auto</b>	After correcting variable adaptation with deletion of a still effective derivative action, the approach to the setpoint is bumpless	The effective setpoint ramp continues running in the background during manual mode. After switching over to automatic, the correcting variable is adapted and the derivative action is deleted and the setpoint is set to the actually reached ramp setpoint (bumpless).
<b>Off -&gt; On</b>	The effective setpoint is set to the process value first and after deleting a still effective derivative action, a setpoint step change to the target is made. During this step change, the PID parameters are effective. The derivative action is a result of the step change (not bumpless).	At first, the effective setpoint is set to the process value. After deleting the derivative action, the approach to the target setpoint is via a ramp. During this transition, the PID parameters are effective (bumpless starting with 0).
<b>W<sub>old</sub> -&gt; W<sub>new</sub></b>	After deleting a still effective derivative action, a step change from the instantaneous to the target setpoint is made. During this step change, the PID parameters are effective. The derivative action is a result of the new step change (not bumpless).	After deleting the derivative action and adapting the correcting variable, changing from the old to the new target setpoint is done via a ramp (bumpless).
<b>W -&gt; W2 W2 -&gt; W We -&gt; Wi, without Tracking Wi -&gt; We</b>	After deleting a remaining derivative action, a setpoint step change from the instantaneous to the target setpoint is made. During this step change, the PID parameters are effective. The derivative action is only a result of the new step change (not bumpless).	The effective setpoint is set to the process value. After deleting the derivative action and adapting the correcting variable, setpoint changing from the process value to the target setpoint is done via a ramp (bumpless).
<b>We -&gt; Wi, with Tracking</b>	The internal target setpoint is set to the actual process value or to the external setpoint. Subsequently, any still effective derivative action is deleted and the correcting variable is adapted (bumpless).	The internal target setpoint is set to the actual process value or to the external setpoint. Subsequently, any still effective derivative action is deleted and the correcting variable is adapted (bumpless).

### Gentle line-out to the target setpoint with ramps

When using a setpoint ramp, a process value overshoot at the ramp end may occur. Due to the difference between setpoint and process value in the course of the ramp, an integral action is built up and must be removed after the end of the ramp. The longer the ramp, the higher the integral action. And the more exact the process value follows the setpoint, the higher the probability that any integral action will cause an overshoot.

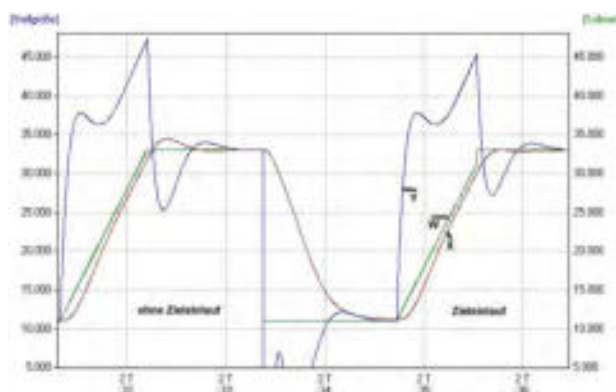
The target line-out function is used to adapt the integral action to the actual PD action at an adjustable distance before reaching the ramp end value, the D-dynamics is initialized and the setpoint is set to the ramp end value. Now the controller dynamics re-starts bumplessly related to the new setpoint.

Controller parameter "a" can be used to define at which distance to the final setpoint the target orientation should be switched over to the final setpoint. The target line-out function is activated under the following conditions :

1.  $W < W_{end}$
2.  $W > W_{end} - 2a$
3.  $X > W_{end} - a$

Marginal conditions / restrictions:

With internal setpoint ramps, the controller knows the future target setpoint. When using external setpoints with ramp function (programmer), the ramp end value must be bound to input X3 of the controller block. When the internal ramp is active, line-out to the target setpoint is always related to the internal ramp end value, and the value at X3 is ineffective.

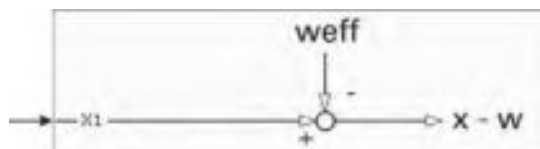


Target line-out is activated only, if the external ramp setpoint changes continuously. The function can be used both with differentiation of control deviation (XW) and differentiation of process value (X). With 3-element control, target line-out is omitted. The signification of parameter "a" is different and connection of an external end setpoint is not possible. With ratio control, target line-out is only restricted with fixed distance (1 in units of the physical quantity). The signification of parameter a is different.

### 3.16.13. Process value calculation

#### Standard controller

The process variable measured via analog input **X1** is used as process value by the controller.



#### Ratio controller

Process control frequently requires various components to be mixed into a product. These components must be mixed according to a given ratio.

The main component is measured and used as reference for the other components. With increasing flow of the main component, the flow of the other components will increase accordingly. This means that process value x used by the controller is determined by the ratio of two input variables rather than being measured as one process variable.

For optimum combustion, the fuel-air ratio must be controlled. With stoichiometric combustion, the ratio is selected so that there are no inflammable residues in the waste gas. In this case, the relative rather than the physical ratio is displayed as process value and adjusted as setpoint.

If the transmitters used by the controller are designed with a stoichiometric ratio  $\lambda = 1$  is met exactly with restless combustion. With a process value display of 1,05, the instantaneous air excess is clearly 5%. The amount of air required for atomizing is taken into account by constant 'N0'. For selecting a ratio controller, **CType = Ratio** must be selected.

Moreover, configuration word '**Ratio**' must be taken into account.

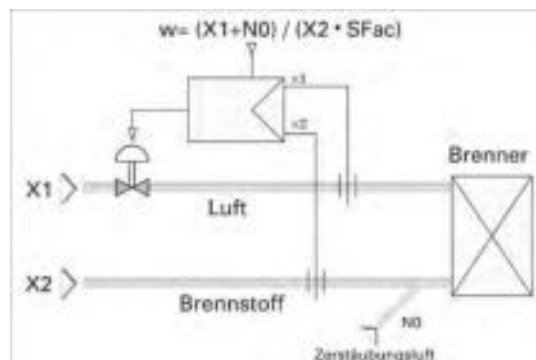
With ratio controller, note that parameters Xn0 and Xn100 must be set to the input range of connector X1.

#### Example of standard ratio control:

Standard ratio control at the example of stoichiometric combustion. Analog input INP1 is configured to 4...20 mA with physical unit m3/h (air). Values 0 and 100 are allocated to input variables 4 mA (**x0**) and 20 mA (**x100**).

Atomizing air N0 is added to this input. E.g. INP5 is selected as second ratio input. This input is also configured for 4...20 mA and m3/h (gas). x0 and x100 values 0 and 100 are allocated to the input variables.

Weff effective as relative ratio is multiplied by stoichiometric factor **SFac** (e.g. SFac = 10), i.e. a „stoichiometric“ flow ratio can be used for calculation of the control deviation. The instantaneous (controlled) process value is calculated from the physical ratio, multiplied by 1/SFac and displayed as relative value.



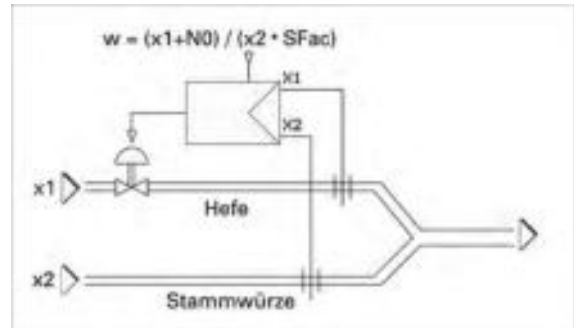
### Material batching and mixing

The following examples are intended to show that various control possibilities can be used. This is necessary, since the materials to be mixed (e.g. paste) are not always directly measurable due to their consistency. Other cases may require a component to be controlled in relation to a total rather than to another component.

$$\text{Ratio = Type 1} \quad W = \frac{X1+N0}{X2 \cdot SFac}$$

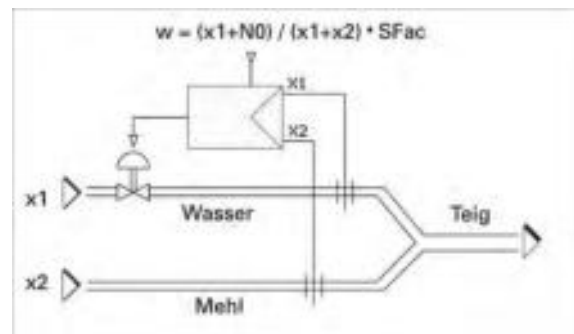
The first case is obvious. Almost everybody knows what happens during brewing.

Yeast (x1) must be batched in relation to the original wort (x2). The is adjusted in '%yeast', e.g. W= 3%. The ratio inputs are scaled in equal units. The control deviation is multiplied by 'SFac = 0,01' and calculated according to equation  $xw = (x1 + N0) - 0,03 w x2$ , so that exactly 3% of yeast are batched with  $xw = 0$ . Process value display is also in %. Constant N0 is without importance (N0 = 0)



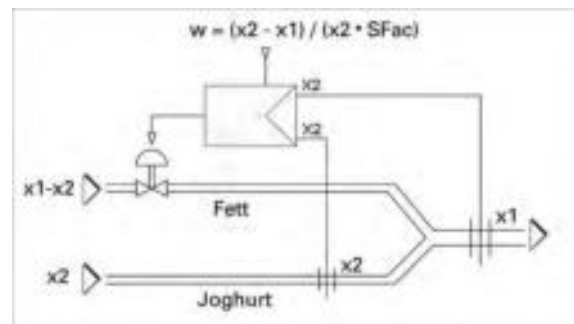
$$\text{Ratio = Type 2} \quad W = \frac{X1+N0}{(X1+X2) \cdot SFac}$$

In this example, water (x1) must be batched as a percentage of the total (paste;  $x1+x2$ ). As the paste quantity is not available directly as a measurement signal, the total is calculated internally from x1 and x2. N0 = 0 must also be adjusted in this case.



$$\text{Ratio = Type 3} \quad W = \frac{X2-X1+N0}{X2 \cdot SFac}$$

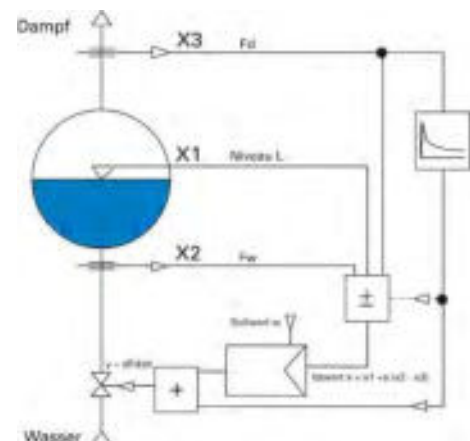
Unlike the previous examples, yoghurt (x2) and the final product (x1) are measured in this case.



### Three-element control

With three-element control, process value calculation is according to equation  $x_{eff} = x1 + a \cdot (x2 - x3)$  whereby term  $(x2 - x3)$  is the difference between the steam and water flow rates. Factor b for flow range matching used so far is omitted, because the mA signals are converted directly into physical units during input value conditioning (x0, x100). The calculated process value is displayed on the process value display.


For selecting a three-element controller, 'CFunc = 3-e1em' must be entered in the configuration.

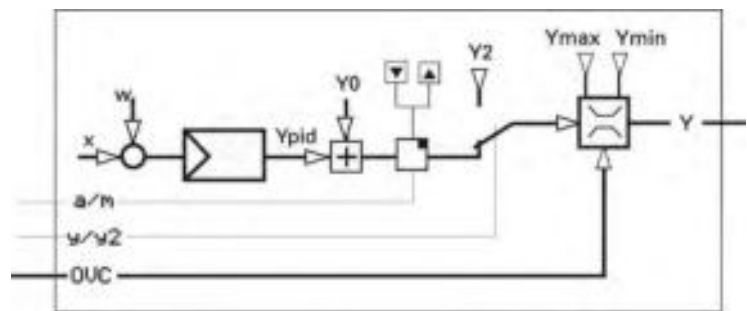




## Correcting variable processing


The following considerations related to correcting variable processing are applicable to continuous controllers, two-point, three-point and three-point stepping controllers with position feedback. The diagram opposite shows the functions and interactions of correcting variable processing.

Both the manipulated variable adjustment from the front  and the control inputs "inc" and "dec" in manual mode in 0.1% increments incremented or decremented. The adjustment speed is one second per 1%.



## Second correcting value

Similar to processing, switch-over to a second preset correcting value **Y2** is possible. Switching over is done via digital input **y/y2**. Whether **Y2** has safety functions, or whether it is only a pre-defined start position in defined process conditions is determined only by the use and integration into an automation concept.

 Second correcting value **Y2** is evaluated with priority. When switching over to **Y2** is done at one point (interface or digital control input '**y/y2**'), switching over at the other point is not possible.

## Correcting variable limits

Parameters **Ymin** and **Ymax** determine the limits of the correcting variable range within 0...100 %.

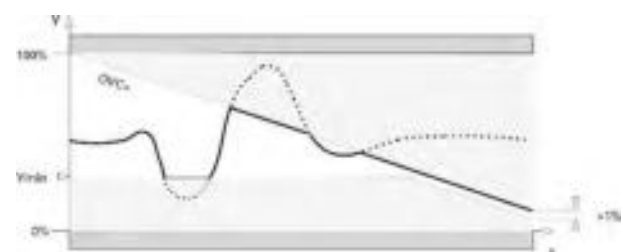
With three-point and continuous controller „split-range“, the correcting variable limits are within -100 ... +100 %. Parameters **Ymin** and **Ymax** are used to specify fixed correcting variable limits.



## External correcting variable limiting (override control)

Dependent of 'COVC' setting, the lowest (OVC-), the highest (OVC+) or lowest and highest correcting value (OVC+/OVC-) can be limited by analog input signals.

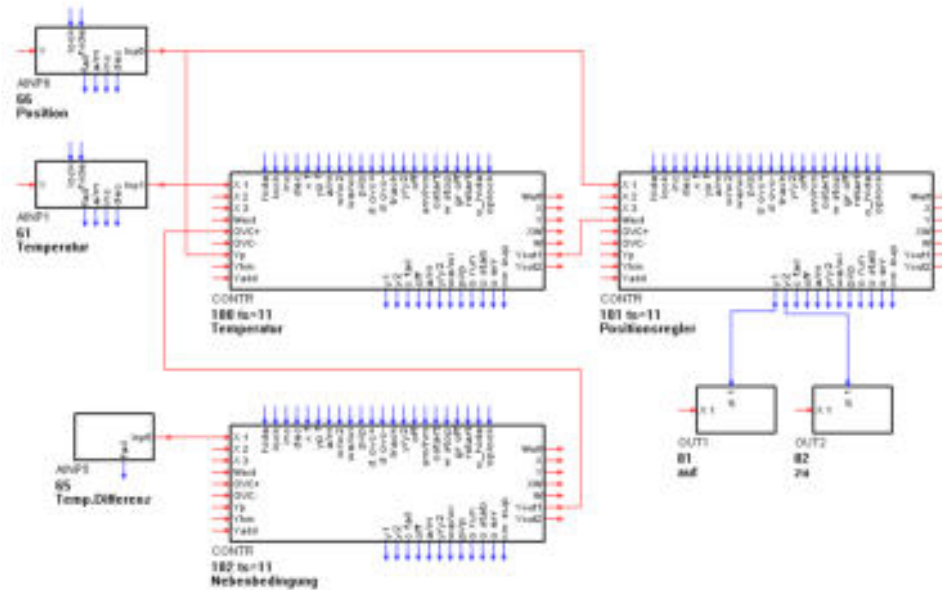
Override control is used where bumpless switch-over to another controller when reaching defined process conditions and mainly according to other criteria is required. The basic principle is that two controllers act on the same motor actuator



### Limiting control

Limiting with continuous output. Limiting control with three-point stepping output can be realized by using a continuous controller with the OVC function. A position controller (three-point stepping) provides override control.

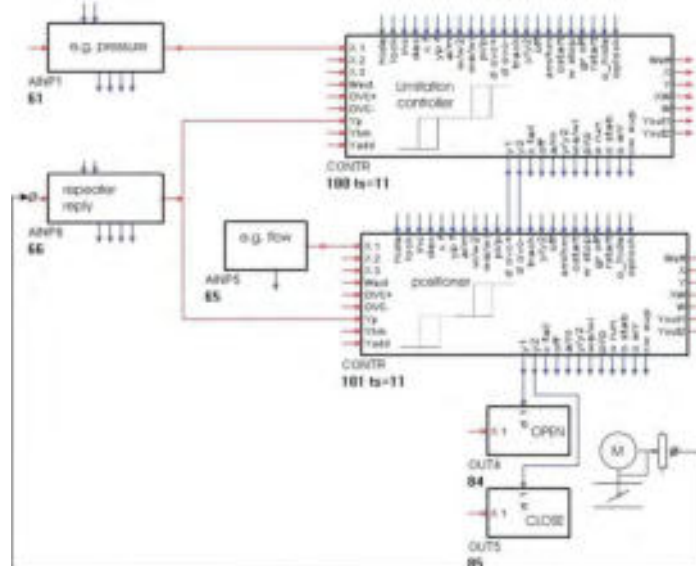
#### Limit control with continuous controller



### Override (limiting) control using a three-point stepping output

Override control is also possible by means of a classical three-point stepping controller. The positioning signals of the limiting controller must be connected as shown in the example.

#### Limit control with motor stepping controller

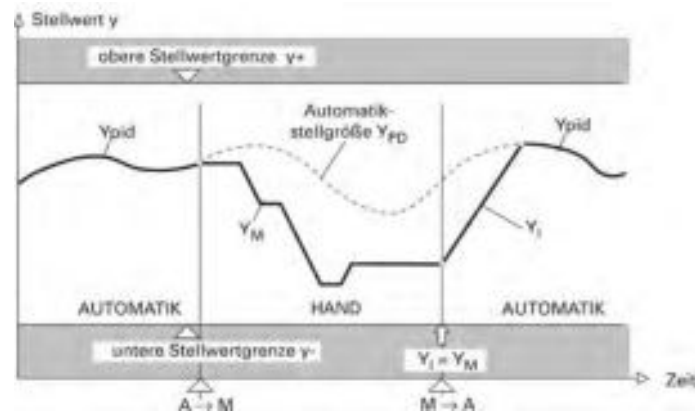


Which one of the two controllers influences the process is decided in the slave controller logic. The first "close" pulse coming from the limiting controller switches over to override control. The limited controller automatically retrieves the positioning authority, when it first tries to close the actuator further.

### Bumpless auto/manual switch-over

Sudden process interventions by control mode switch-over are usually not desired. Excepted is purposeful switch-over  $y \rightarrow Y_2$ .

A  $\rightarrow$  M switch-over is always bumpless; the last correcting value is frozen and can be changed manually. M  $\rightarrow$  A switch-over is different. Correcting value differences are compensated as follows: when switching over, the controller integral action is set to correcting value  $Y_M$  output last plus correcting variable portions of the controller P and D action running in the background ( $Y_I = Y_M + Y_{PD}$ )



### 3.16.14. Small controller ABC

Some operating principles, which are realized in the controller (✓) or which are possible by means of an additional engineering are explained in the following section (✎). Cross references are shown in italics.


#### ✓ *Anti-Reset-Wind-Up*

Measure which prevents the controller integrator from saturation.

#### ✓ *Working point (y0)*

The working point of a P or a PD controller indicates the value output to the process with process value = . Although this value is only important for P and PD controller, it can also be of interest for controllers with integrator (automatic working point).

#### ✓ *Automatic operation*

Normal controller operation. The controller controls the process by means of the adjusted control parameters. Automatic operation is effective with **a / m** (di7) set to 0 (automatic) AND automatic selected via front-panel key  AND **sm / hm** (di16) set to 0 (soft manual). Contrary: manual operation.

#### ✓ *Cutback*

Reset of the integral action shortly before reaching the end setpoint with setpoint ramps.

#### ✓ *Cycle time*

The duration of a switching cycle (pulse and pause) at 50 % power control of a 2-point controller.

#### ✓ *Line-out to the target*

By early setpoint switch-over to the ramp end setpoint, the controller is given a new target orientation for smooth line-out to the target..

#### ✓ *Bandwidth control*

With program control or gradient control, there may be a considerable control deviation if the process is slow. This can be prevented by monitoring the control deviation for an adjusted tolerance band by means of additional function blocks. With out-of-tolerance, the change is stopped (**w stop** with controller or **stop** with program controller).

#### ✓ *Three-element control*

Particularly suitable for processes in which load changes would be detected too late (e.g. level control for steam boilers). In this case, a disturbance variable is used at which the mass balance (steam removal, feed water) is evaluated, subtracted and added to the control variable (after differentiation, if necessary).


#### ✓ *Feed-forward control*

Especially suitable for processes with long delay time, e.g. pH-control. A disturbance variable is used, at which the evaluated, differentiated or delayed value of an analog input (**YAdd**) is added directly to the controller output for avoiding the controller time behaviour.



#### ✓ *Gradientenregelung*

Particularly suitable for processes in which energy shocks or quick setpoint changes must be avoided. setpoint changes are bumpless in both directions, since the effective setpoint always runs towards the changed (destination ) by means of gradients **Grw+** or **Grw-**. For the second **w2**, gradient **Grw2** acts in both directions, also with **w** → **w2** switch-over.



#### ✓ *Manual operation*

When switching over to manual operation, the automatic sequence in the control loop is interrupted. Modes soft manual and hard manual are available. Switch-over automatic → manual and vice versa are bumpless. Manual operation is effective with **a / m** (di7) set to 1 (manual) OR manual selected via front-panel key  OR **sm / hm** (di16) set to 1 (hard manual). Contrary: automatic.



If automatic remains selected via key , the controller changes to automatic after omission of the di7 signal. With manual selected additionally via key , the controller remains in manual mode after omission of the di7 signal!

### ✓ *Hard manual (sm / hm)*

Safety output value **Yhm**. The controller output goes to the preset value immediately, when hard manual is active (the controller is switched to manual mode directly). Keys  /  are without effect. Switch-over to automatic mode is Bumpless.

### ✓ *Cascade control*

Particularly suitable for temperature control in e.g. steam boilers. A continuous master controller (load controller) provides its output signal as an external to a slave controller, which varies the output value.

### ✓ *Override control (OVC) → page 250*

Limiting of the min. (**OVC-**) or max. (**OVC+**) output value to the value of an analog input. Limitation by override control can be used e.g. with control continued by a different controller dependent of different conditions when reaching defined process statuses. The transitions from unlimited → limited output value and vice versa are bumpless.

### ✓ *Program control*

The effective setpoint follows the profile of a programmer (APROG with APROGD) connected to input **Wext**; the controller must be set to **we** (di9=0).

### ✓ *Process at rest*

For a clear optimization attempt during self-tuning, the control variable must be in a stable position. Various rest conditions can be selected:

Process behaviour with constant output value	Recommended setting	Stability <b>PIR_H</b> is reached, if
A constant process value is reached in relatively short time (standard process).	<b>Grad=0</b>	the process value is constant during 1 minute.
After a relatively long time, a constant process value is reached (slow process).	<b>Grad&lt;0 / &gt;0</b>	the process value decreases constantly during 1 minute (controller inverse) or increases constantly during 1 minute (controller direct).
The process is affected from outside.	<b>grad&lt;&gt;0</b>	the process change is constant during 1 minute. The output action is unimportant.

### ✓ *Ramp function*

changes in ramps rather than in steps. See gradient control.

### ✓ *Control parameters*

For controller optimization, the controller must be matched to the process characteristics (→ see page 224 ff). The effective parameters are **Xp1**, **Tn**, **Tv** and **Y0**. Dependent of controller operating principle, the following additional parameters are possible: **Tp1** (with 2-point/3-point controllers), **Xp2** and **Tp2** (with 3-point controllers), **Xsh** and **Tpu1s** and **Tm** (with 3-point stepping controllers).

### ✓ *Control behaviour*

Generally, fast line-out to the without overshoot is required. Dependent of process, various control behaviours are desirable for this process:

- easily controllable processes ( $k < 10\%$ ) can be controlled with PD controllers,
- processes with medium controllability ( $k 10...22\%$ ) using PID controllers and
- badly controllable processes ( $k > 22\%$ ) with PI controllers.



### ✓ *Controller OFF (off)*

With input **off** =1, there are no pulses at the switching output and the continuous outputs are 0%.

### ✓ *Self-tuning*

For optimum process control, the controller must be matched to the process requirements. The time required for this purpose can be reduced considerably by self-tuning (→ Fehler! Textmarke nicht definiert. Self-tuning). During self-tuning, the controller makes one adaptation attempt during which the control parameters are determined automatically from the process characteristics for fast line-out to the without overshoot..

### ✓ *Soft-Manual*

Usual manual operation: with automatic → manual change-over, the last output value remains active and can be adjusted via keys  / . Transitions automatic → manual and vice versa are bumpless.

### ✓ *Setpoint switch-over*

In principle, the following setpoints are possible: internal **wi**, second internal **w2** and external **we**. With program control, external **we** must be selected. The analog comes from APROG and is applied to input **Wext**.

### ✓ *y feedback control*

Particularly suitable for processes in which load changes lead to process value drops. A load-dependent change to (preferable) or process value is made. The evaluated and filtered output value is added to the in a separate function block. Use the **Wext** input and set the controller to **we**.

### ✓ *PI/P switch-over*



When optimizing slow processes, e.g. big furnaces, the controller I action can cause problems: if starting up was optimized, line-out can be slow; with optimization of the disturbance behaviour, there may be an important overshoot. This effect is prevented by switching off the I action during start-up or with high control deviations (e.g. by applying a limit contact to the control deviation) and switching it on again only when the process value approaches the Setpoint. To prevent permanent control deviations, the limit contact must be further away from the than the permanent control deviations.

### ✓ *Tracking*

During switch-over from external or program setpoint to internal setpoint, setpoint or output value step changes may occur. By means of the tracking function, the transition is bumpless.

- Process value tracking: During switch-over, the effective process value is used as internal setpoint.
- Setpoint-tracking: During switch-over, the external or program setpoint used so far is taken over as internal setpoint

### ✓ *Behaviour with fail (configuration of the controller behaviour with sensor failure, xf)*

Selected behaviour	Effect with 3-point stepping controllers	Effect with other controllers
<b>Neutral</b>	No output pulses	No output pulses or 0%
<b>Ymin</b>	Actuator is closed	Ymin ( $\triangleq$ limiting)
<b>Ymax</b>	Actuator is opened	Ymax ( $\triangleq$ limiting)
<b>Y2</b>	Not selectable	Y2 fixed, also with manual operation
<b>Y2 / Yman</b>	Not selectable	Y2, adjustable in manual mode with  

### ✓ *Ratio control*

Particularly suitable for controlling mixtures, e.g. fuel-air mixture for ideal or stoichiometric combustion. For taking e.g. the atomizer air into account, zero offset **N0** can be added..

### ✓ *x/xw differentiation*

Dynamic changes of process value or setpoint affect control differently. x-differentiation: Process value changes (disturbances) are used dynamically to permit better control results. xw-differentiation: Changes of process value (disturbances) and setpoint are used dynamically to permit a better control result. In this case, the improvement is dependent of both disturbance and control behaviour.

### ✓ *Controller operating principle*

The static operating principles for controllers with P or PD behaviour with adjustable working point Y0 are shown. On controllers with I action, the working point is shifted automatically. The outputs ( $\odot$ ) are described with **h** („heating“), **c** („cooling“), („open“) and („close“).

### 3.17. Inputs / outputs

The KS98-2 offers a comprehensive modular expandable input / output configuration.

#### *Basic configuration*

Each basic unit initially offers the following inputs / outputs:

- One universal input (UNI\_IN) for direct connection of sensors or standard signals.
- A voltage source TPS for sensor supply.
- Two digital control inputs for 24 volt logic signals.
- 2 or 4 relais outputs.

#### *Internal I / O extension*

Plug-in I / O modules are available for individual task-related expansion of the inputs and outputs.

The basic version of the KS 98-2 has 2 module slots with terminals in terminal row A and in the version with 2 relais an additional 2 slots with terminals in terminal row P.

For terminal rows B and C, carrier modules with 4 option module slots or modules with digital inputs / outputs can be ordered. (10DI, 4DO)

Each plug-in module has one or two input / output channels.

#### **Option modules**

##### *Analog inputs*

- U: 1x Universal Input (UNI\_IN)
- R: 2x Resistive Measurements (R\_IN)
- T: 2x Thermocouple, mV, mA
- V: 2x Voltage (Re >> 1GΩ)
- P: 1x 0/4...20mA with Transmitter Power Supply

##### *Analog Outputs*

- L: 2x Linear output 0/4...20mA
- B: 2x Bipolar output -10V...10V, 0/2...10V

##### *Digital-In-/Outputs*

- D: 2x 24V Digital I/O
- A: 2x SSR Driver

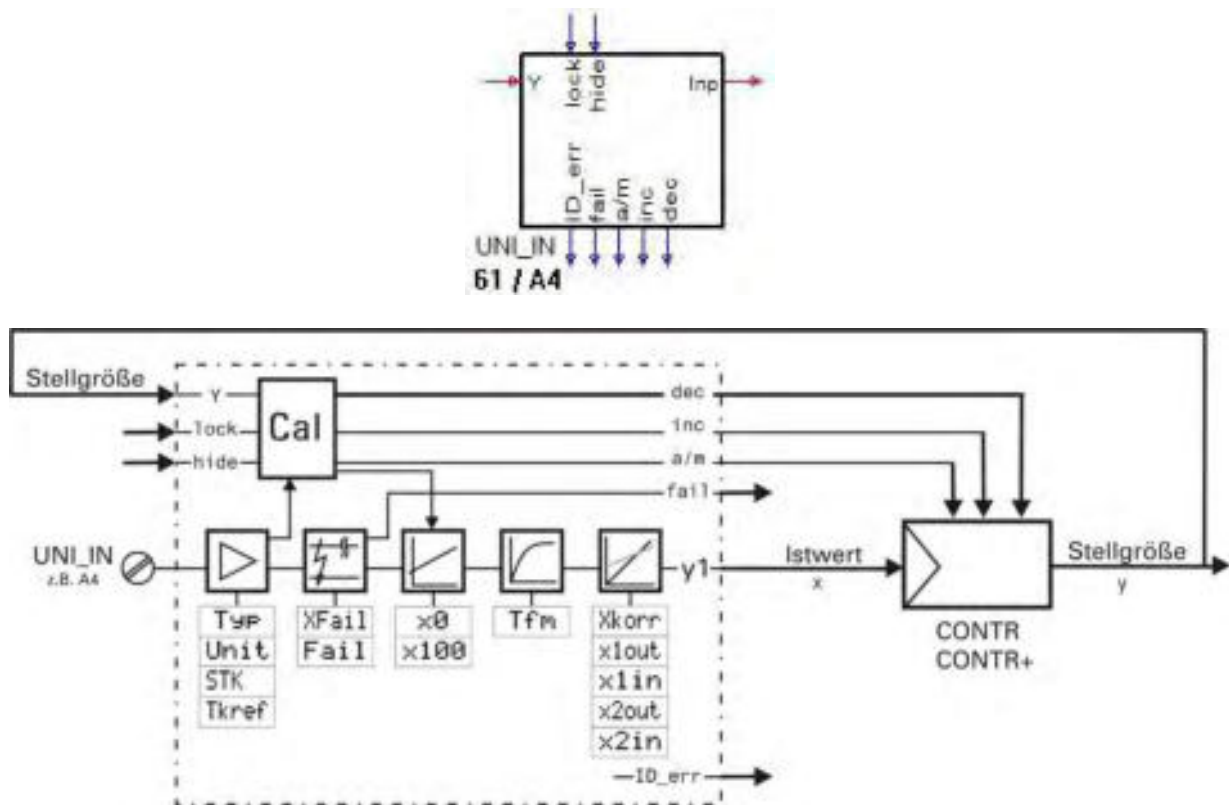
Both channels of the D-Module can be configured separately for input or output usage. Signal state can be reversed.

#### *Limitations to take into account*

To avoid inadmissible self-heating the number of output extension modules is limited. This concerns L, B, A and P modules. The following rule applies.

- When using fieldbus option Max 4 modules per unit
- When not using fieldbus option Max 5 modules per unit

### 3.17.1. UNI\_IN (analog universal input-Modul U)



For direct connection of temperature sensors, for potentiometric transducers and standard signals

#### General

Function 'UNI\_IN' is used for configuration and parameter setting of analog input. The function provides a corrected measurement value and a measurement value status signal at its outputs

#### Inputs/outputs

##### Digital inputs:

lock	Calibration locked (with <b>lock</b> = 1 calibration is locked)
hide	Display suppression (with <b>hide</b> = 1 the calibration page is not displayed)

##### Digital outputs:

fail	Signals an input error (short-circuit, polarity error, ..)
a/m	Manual signal, switches the controller during the calibration of the potentiometer in manual mode.
inc	Increment signal
dec	Decrement signal
ID_err	0 = correct module detected    1 = wrong module detected

##### Analog inputs:

Y	Output variable (is only used when calibrating a remote control input)
---	--

##### Analog outputs:

Inp1	Signal input
------	--------------



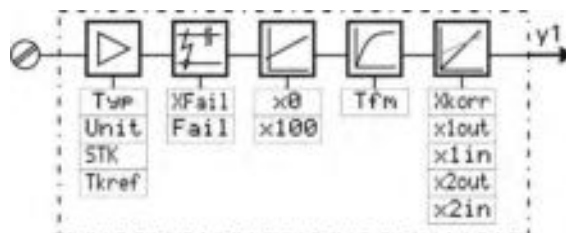
## Parameter and configuration data

Parameter	Description	Values	Default
x1in	Measured value correction P1, input	-29999 ... 99999	0
x1out	Measured value correction P1, output	-29999 ... 99999	0
x2in	Measured value correction P2, input	-29999 ... 99999	100
x2out	Measured value correction P2, output	-29999 ... 99999	100

Configuration	Description	Values	Default
Typ	Typ L -200...900 °C	Typ L	
	Typ J -200...900 °C	Typ J	
	Typ K -200...1350 °C	Typ K	
	Typ N -200...1300 °C	Typ N	
	Typ S -50...1760 °C	Typ S	
	Typ R -50...1760 °C	Typ R	
	Typ T -200...400 °C	Typ T	
	Typ W 0...2300 °C	Typ W	
	Typ E -200...900 °C	Typ E	
	Typ B (25) 400...1820 °C	Typ B	
	Pt 100 -200...850,0 °C	Pt100 850	←
	Pt 100 -200...250,0 °C	Pt100 250	
	2x Pt 100 -200...850 °C	2Pt100 85	
	2x Pt 100 -200...250,0 °C	2Pt100 25	
	0...20 mA	0...20mA	←
	4...20 mA	4...20mA	
	0...10 V	0...10V	
	2...10 V	2...10V	
	Transducer 0...500 $\Omega$	Pot.trans	←
	Resistance 0...500 $\Omega$ linear	0...5000hm	
	Resistance 0...250 $\Omega$ linear	0...2500hm	
	Fail function off	disabled	
Fail	Digital output fail = 1, y1 = x100	Upscale	←
	Digital output fail = 1, y1 = x0	Downscale	
	Digital output fail = 1, y1 = XFail	Subst.val.	
Xkorrr	Measured value correction off	off	←
	Measured value correction adjustable	on	
Unit	Unit = °C	only effective with thermocouple and Pt100 setting	←
	Unit = °F	°F	
STK	Internal temperature compens.	int.TK	←
	External temperature compens.	ext.TK	
x0	Physical value at 0%	-29999 ... 99999	0
x100	Physical value at 100%	-29999 ... 99999	100
XFail	Substitute value with sensor error	-29999 ... 99999	0
Tfm	Filtertime constant [s]	0 ... 99999	0,5
Tkref	Reference temperature at STK = ext.TK	0 ... 140	0

## Measured value conditioning

Before the pre-filtered (time constant ...; limiting frequency ...) analog input signals are available as digitized measurement values with physical quantity, they are subjected to extensive measured value conditioning.



### **Input circuit monitor**

#### ☐ **Thermocouples**

The input circuit monitor monitors the thermocouple for break and polarity error. An error is determined if the measured thermovoltage signals a value which is by more than 30 K below the span start.

#### ☐ **Pt100 measurements and transducers** are monitored for break and short-circuit.

#### ☐ **Current and voltage signals**

With current (0/4...20mA) and voltage signals (0/2...10V), monitoring for out-of-range ( $I > 21,5 \text{ mA}$  or  $U > 10,75 \text{ V}$ ) and for short circuit ( $I < 2 \text{ mA}$  or  $U < 1 \text{ V}$ ) with "life zero" signals is provided..

Sensor errors are output as digital output (**fail**). In case of error, statuses '**Upscale**', '**Downscale**' or '**Subst.val**.' defined in the configuration can be preset for the input circuit.

Thermocouples and Pt100 are generally recorded over the entire physical measuring range according to the data sheet and linearized according to their allocation table. The linearization is realized by approximating the error curve with up to 28 interpolation points.

### **Scaling**

mA and V standard signals are always scaled according to the physical measuring range of the transmitter (**x0**, **x100**).

With transducer measurements, "calibration" is according to the proven method. Bring the transducer to start and then to end position and "calibrate" it to 0 % or 100 % by key pressure. In principle, calibration corresponds to a scaling, whereby gradient and zero offset are calculated automatically by the firmware.

### **Additional measurements**

Dependent of configured sensor type, additional and corrective measurements are required.

The amplifier zero is checked with all measurement types and included into the measurement value. The lead resistances with Pt100 and transducer, and the cold-junction reference temperature (internal TC) are measured additionally.

### **Filter**

A 1st order filter is adjustable in addition to the analog part filtering of each input signal.

For measured value processing, a filter time constant with a numerical value between 0.0 and 999999 can be set ( $\rightarrow$  **Tfm**).

### **Sampling intervals**

The sampling interval for the UNI\_IN is 100ms.

### **Linearization error**

Thermocouples and Pt100 are linearized over the overall physical measuring range. Linearization is with up to 28 segments, which are placed optimally on the error curve by a computer program and thus compensate the linearity errors. As error curve approximation is provided only by segments (polygons) rather than by an nth order polynomial, there are points on the characteristic in which the residual error is zero. Between these "zero points", however, the residual error has very small, but measurable values. For the reproducibility, however, this error is irrelevant, because it would repeat itself in exactly the same point and amount, if the measurement would be repeated under identical conditions.

### **Temperature compensation**

Measurement of the cold-junction reference temperature is using a PTC resistor. The temperature error thus determined is converted into mV of the relevant thermocouple type, linearized and added to the measured value as corrective value with correct polarity. The remaining error with varying cold-junction reference temperature is approx.

0,5K/10K, i.e. about one tenth of the error which would occur without compensation. Better results are possible with controlled external TC, which is adjustable within 0...+140°C at the cold junction reference dependent of controlled temperature. With cold-junction reference measurements for "reproducibility" assessment, however, utmost care must be taken that constant environmental conditions are not exceeded when working with internal TC. An air draft at the PTC resistor of the cold junction reference can be sufficient to falsify the measurement result

### Measured value correction


The measurement can be corrected in various ways using the measured value correction.

Pre-requisite: configuration **XKorr = ein**.

In most cases, the relative accuracy and reproducibility rather than the absolute one are of interest, e.g.:

- the compensation of measurement errors in one working point ( control)
- -minimization of linearity errors in a limited operating range (variable )
- -correspondence with other measuring facilities (recorders, indicators, PLCs, ...)
- -compensation of sample differences of sensors, transmitters, etc..

Measured value correction is designed for zero offset, gain matching and for both. It corresponds to scaling  $mx+b$ , with the difference that the KS 98-1 firmware calculates gain  $m$  and zero offset  $b$  from the value pairs for process value ( $x1in$ ;  $x2in$ ) and ( $x1out$ ;  $x2out$ ) of two reference points.

 For a comparison measurement with a calibrated measuring instrument, the default values for  $x1in$ ,  $x1out$  (0) and  $x2in$ ,  $x2out$  (100) must first be entered.

#### Example 1:

Zero offset

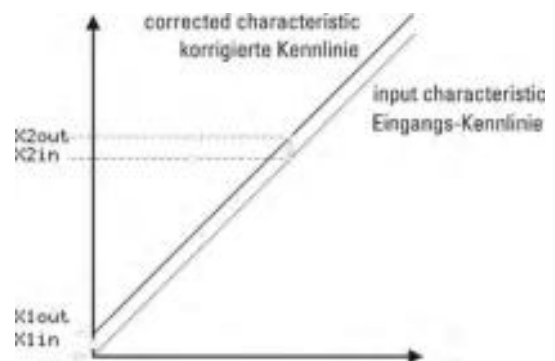
$x1in = 100$

$x1out = 100 + 1,5$

$x2in = 300$

$x2out = 300 + 1,5$

The corrected values are shifted evenly with reference to the input values over the complete range.



#### Example 2:

Gain change (rotation around the coordinate origin)

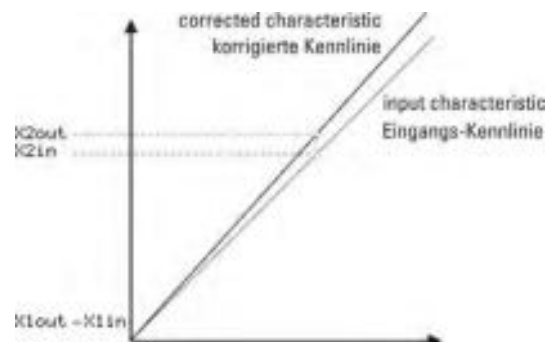
$x1in = 0$

$x1out = 0$

$x2in = 300$

$x2out = 300 + 1,5$

The corrected values diverge despite equality with the input values at  $x1in$  and  $x1out$ .



#### Example 3:

zero and gain matching

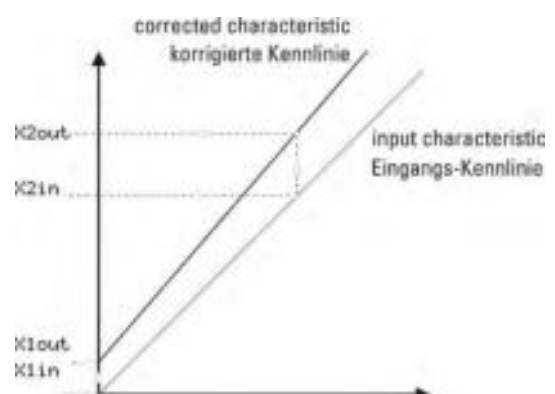
$x1in = 100$

$x1out = 100 - 2,0$

$x2in = 300$

$x2out = 300 + 1,5$

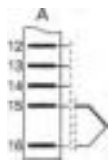
The corrected values are shifted already at values  $x1in$  and  $x1out$  and diverge additionally.



### Sensor types

The input sensor type can be determined as thermocouple, resistance thermometer, potentiometric transducer or as standard signal (current and voltage). The physical quantity is freely selectable.

#### Input thermocouple



The following thermocouple types are configurable as standard:

Type L, J, K, N, S, R, T, W, E and B according to IEC584.

The signal behaviour can be affected by the configuration of the following points. Distinction between internal and external temperature compensation (→ **STK**) is made.

- Internal compensation:  
the compensating lead must be taken up to the multi-function unit connecting terminals. No lead resistance adjustment is required.
- External temperature compensation:  
A separate cold-junction reference with a fixed reference temperature must be used (between 0 and 140°C configurable) (→ **Tkref**). The compensating lead must be taken only up to the cold-junction reference, the cable between reference and multi-function unit terminals can be of copper. No lead resistance adjustment is necessary.
- The action of the built-in TC break protection can be configured for upscale (<< process value) or downscale (>> process value) or fixed to a substitute value (→ **Fai1**).
- For measured value processing, a filter time constant with a numeric value between 0,0 and 200000 is adjustable (→ **Tfm**).
- A process value correction is configurable (→ **Xkorr**).

#### Resistance thermometer input

Resistance thermometer, temperature difference

With a resistance thermometer, the signal behaviour with sensor break can be determined (→ **Fai1**). No temperature compensation is required and is therefore switched off. With temperature difference measurement, calibration by means of short circuit is required. If lead resistance adjustment is necessary, it can be realized by means of a 10 Ω calibrating resistor (order no. 9404 209 10101).

Dependent of signal source type, the unit is configured for one of the following inputs:

- resistance thermometer Pt 100 with linearization
- temperature difference with 2 x Pt 100 and linearization
- linear potentiometric transducer

For measured value processing, a filter time constant with a numeric value within 0 and 999 999 can be adjusted (→ **Tfm**). Process value correction can be configured (→ **Xkorr**).

#### Resistance thermometer Pt 100

The two ranges -200,0...+250,0 °C and -200,0...+850,0 °C are configurable (→ type).

Connection in two or three-wire circuit is possible. Copper lead must be used for measurement. The input circuit monitoring responds at -130°C (sensor or lead break). The action is configurable for:

**Upscale** (Setpoint << process value)

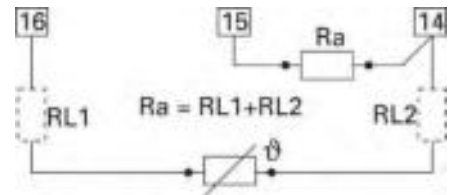
**Downscale** (Setpoint >> process value)

**Substitute value** ((the entered value is used as measured value in case of failure).

### Resistance thermometer in 2-wire connection:



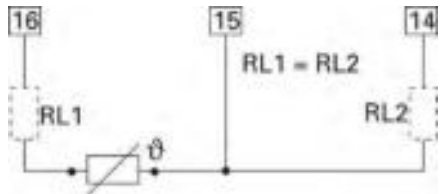
For lead resistance adjustment, disconnect the input leads from the multifunction unit terminals and short circuit them in the connecting head of the resistance thermometer. Measure the lead resistance by means of a resistance bridge and change lead adjusting resistor ( $R_a$ ) so that its value is equal.



### Resistance thermometer in 3-wire connection:



The resistance of each input lead must not exceed  $30\ \Omega$ . No lead resistance adjustment is necessary, provided that the resistances of the input leads  $RL$  are equal. If necessary, they must be equalized by means of a calibrating resistor.



### Temperature difference 2 x Pt100

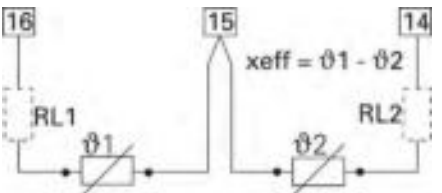


Range  $850^\circ\text{C}$ :  $X0 = -950^\circ\text{C}$ ;  $X100 = 950^\circ\text{C}$  (Typ = 2Pt100 85)  
 Range  $250^\circ\text{C}$ :  $X0 = -250^\circ\text{C}$ ;  $X100 = 250^\circ\text{C}$  (Typ = 2Pt100 25)  
 For lead resistance adjustment, the two thermocouples must be short-circuited in the connecting head.  
 Select the calibration accordingly:

Main menu → Device settings → Calibration

With blinking **Set Dif**, wait until the input has settled (minimum 6 s). Press → Cal done is displayed → Lead resistance adjustment is finished. Remove the two short-circuits.

These resistance values are stored as configuration X0, X100.



### Potentiometric transducer



GO overall resistance  $\leq 500\ \Omega$  incl.  $2 \cdot RL$ .  
 Calibration or scaling is with the sensor connected.

Before calibration, the mains frequency required for operation must be adjusted.  
 Main menu → Device settings → Device data → Frequ.

Calibration is as follows.

Calibration is as described in section 1.10.4 The user-specific calibration values are stored in configurations X0, X100.

### X0, X100 application principle

Configurations X0, X100 are used in different ways dependent on the input type:

- Current input: X0, X100 are scaling values of the signal source (e.g. temp. transmitter):  $0\ \text{mA} = X0$ ,  $20\ \text{mA} = X100$ .
- Potentiometric transducer input: X0, X100 are the user-specific calibration values. Within range X0, X100, display of 0...100% input signal is required.
- Temperature difference input: X0, X100 contain the lead resistance values after calibration by the user: X0 is the lead resistance of connected sensor 1. X100 is the lead resistance of connected sensor 2.

X0 and X100 are parameters of function block AINP1, i.e. part of the engineering. This means that the user calibration remains unchanged after replacing the unit, if it was reloaded into the engineering after calibration.

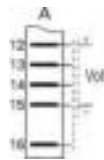
### Standard 0/4...20 mA current input



The input resistance is 50  $\Omega$ .

During configuration, distinction of 0...20 mA and 4...20 mA is made. For 4 ... 20 mA standard signal, the signal behaviour with sensor break can be determined (**Fai1**). Additionally, physical input signal scaling using a defined value of **X0** and **X100** is possible. For measured value processing, a filter time constant with a numeric value between 0,0 and 200000 can be adjusted (r **Tfm**).

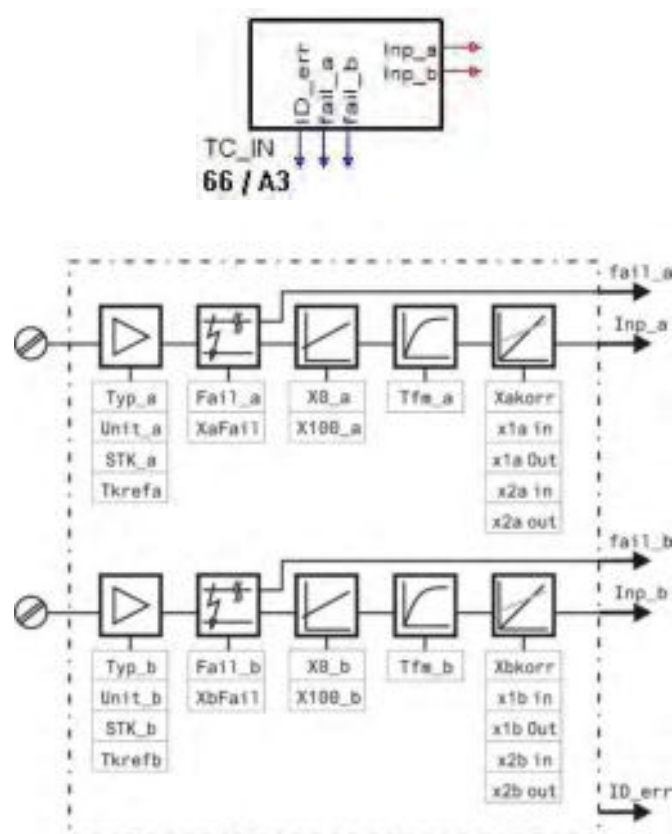
### 0/2...10 V voltage signal input



The input resistance is  $\geq 100 \text{ k}\Omega$

During configuration, distinction of 0...10 V and 2...10 V is made. For 2 ... 10 V standard signal, the signal behavior with sensor break can be determined (**Fai1**). Additionally, physical input signal scaling with a defined value of **X0** and **X100** is possible. For measured value processing, a filter time constant with a numeric value within 0,0 and 200000 can be adjusted ( $\rightarrow$  **Tfm**).

## 3.17.2. TC\_IN (analog input card TC, mV, mA)



The function TC\_INP is used for configuration and parameter setting of the analog inputs TC\_INP. Calculation of the inputs is fixed to once per time slot.

### Digital outputs:

ID_err	0 = correct module fitted 1 = wrong module fitted
fail_a	0 = no measurement error at channel a detected 1 = measurement error at channel a detected; e.g. sensor break
fail_b	0 = no measurement error at channel b detected 1 = measurement error at channel b detected; e.g. sensor break

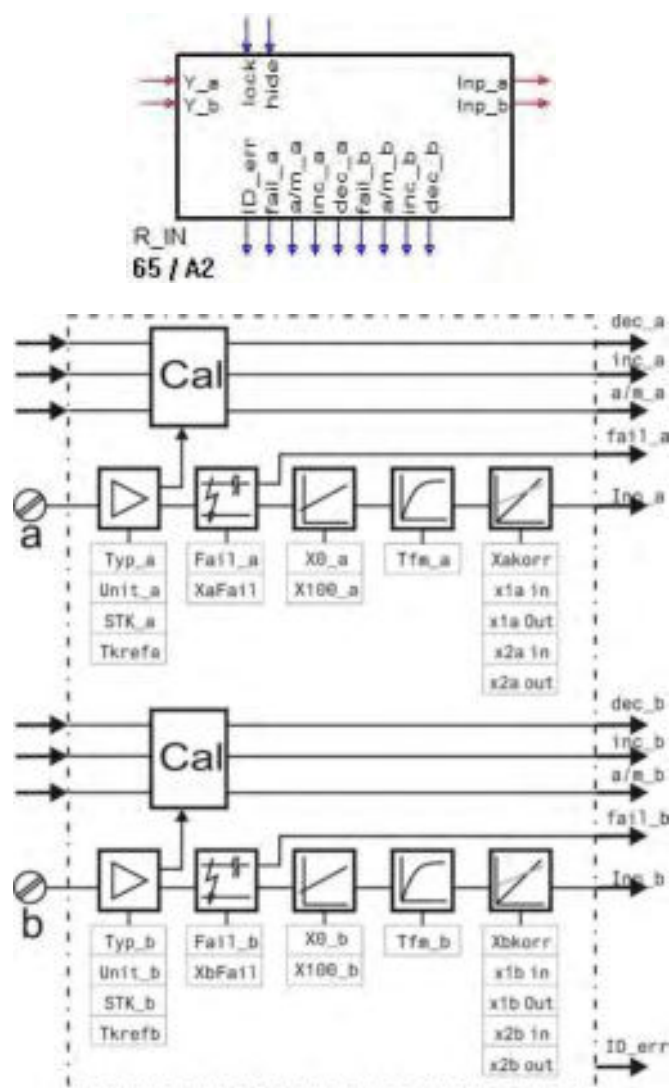
**Analog outputs:**

Inp\_a → measurement value channel a  
 Inp\_b → measurement value channel b

Parameter	Description	Values	Default
x1a in	Measured value correction Inp_a, P1 input value	Real	0
x1aOut	Measured value correction Inp_a, P1 output value		0
x2a in	Measured value correction Inp_a, P2 input value		100
x2aOut	Measured value correction Inp_a, P2 output value		100
x1b in	Measured value correction Inp_b, P1 input value		0
x1bOut	Measured value correction Inp_b, P1 output value		0
x2b in	Measured value correction Inp_b, P2 input value		100
x2bOut	Measured value correction Inp_b, P2 output value		100

Configuration	Description	Values	Default
Typ_a Typ_b	Typ L -200...900 °C	00	30
	Typ J -200...900 °C	01	
	Typ K -200...1350 °C	02	
	Typ N -200...1300 °C	03	
	Typ S -50...1760 °C	04	
	Typ R -50...1760 °C	05	
	Typ T -200...400 °C	06	
	Typ W(C) 0...2300 °C	07	
	Typ E -200...900 °C	08	
	Typ B 0...1820 °C	09	
	Typ D 0...2300 °C	10	
	Voltage 0...30mV	27	
	Voltage 0...100mV	28	
	Voltage 0...300mV	29	
	Standard signal 0...20mA	30	
	Standard signal 4...20mA	31	
	Switched off	0	
Fail_a	Upscale, Inp_a (Inp_b) = x100_a (x100_b)	1	1
Fail_b	Downscale, Inp_a (Inp_b) = x0_a (x0_b)	2	
	Substitute value, Inp_a (Inp_b) = XaFail (XbFail)	3	
Xakorr	Measured value correction Inp_a (b) switched off	0	0
Xbkorr	Measured value correction Inp_a (b) effective	1	
Unit_a	Unit of the measured value of Inp_a (b) = °C	1	1
Unit_b	Unit of the measured value of Inp_a (b) = °F	2	
STK_a	Internal temperature compensation	1	1
STK_b	External temperature compensation	2	
x0_a(b)	Physical value Inp_a (Inp_b) at 0%	Real	0
x100_a(b)	Physical value Inp_a (Inp_b) at 100%	Real	100
Xa(b)Fail	Substitute value for sensor error at Inp_a(b)	Real	0
Tfm_a(b)	Filter time constant of _a (Inp_b) in seconds	Real	0,5
Tkrefa(b)	Reference temperature for Inp_a(b) at STK_a(b)	Real	0

### 3.17.3. R\_IN (analog input card)



#### Analog input card for Pt100/1000, Ni 100/1000, resistance and potentiometer

Analog input, plugs into modular options card C. The R\_INP is used for configuration and parameter setting of analog inputs R\_INP. Input calculation is fixed to once per time slot.

#### Digital inputs

lock = 1 → calibration disabled  
hide = 1 → calibration display suppressed

#### Digital outputs:

ID\_err 0 = correct module inserted  
1 = wrong module inserted  
fail\_a(b) 0 = no measurement error at channel a (b) detected  
1 = measurement error at channel a (b) ; e.g. sensor break  
a/m\_a(b) Status of manual key → 0 = automatic  
Status of manual key → 1 = manual  
inc\_a(b) = 1 → ▲ - key pressed  
dec\_a(b) = 1 → ▼ - key pressed



**Analog input:**

Y\_a(b) → position feedback

**Analog outputs:**

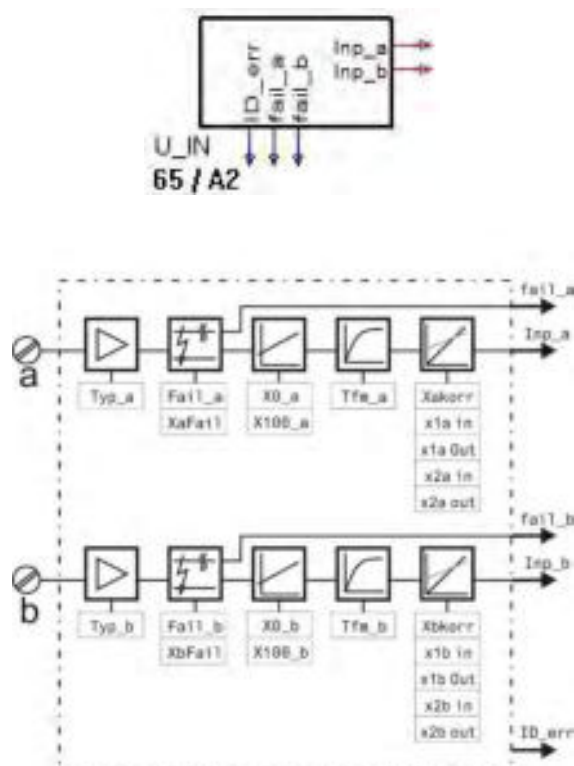
Inp\_a → measured value channel a

Inp\_b → measured value channel b

Parameter	Description	Value	Default
x1a in	Measured value correction Inp_a, P1 input value		0
x1aOut	Measured value correction Inp_a, P1 output value		0
x2a in	Measured value correction Inp_a, P2 input value		100
x2aOut	Measured value correction Inp_a, P2 output value	Real	100
x1b in	Measured value correction Inp_b, P1 input value		0
x1bOut	Measured value correction Inp_b, P1 output value		0
x2b in	Measured value correction Inp_b, P2 input value		100
x2bOut	Measured value correction Inp_b, P2 output value		100

Configuration	Description	Values	Default
Typ_aTyp_b	Pt100 (850) -200 ... 850 °C	00	
	Pt100 (100) -200 ... 100 °C	01	
	Pt1000 (-1) -200 ... 850 °C	02	
	Pt1000 (-2) -200 ... 100 °C	03	
	Ni100 -60 ... 180 °C	04	
	Ni1000 -60 ... 180 °C	05	
	R160 resistance 0 ... 160 Ohm	06	
	R450 resistance 0 ... 450 Ohm	07	00
	R1600 resistance 0 ... 1600 Ohm	08	
	R4500 resistance 0 ... 4500 Ohm	09	
	Potentiometer 160 Potentiometer 0 ... 160 Ohm	10	
	Potentiometer 450 Potentiometer 0 ... 450 Ohm	11	
	Potentiometer 1600 Potentiometer 0 ... 1600 Ohm	12	
	Potentiometer 4500 Potentiometer 0 ... 4500 Ohm	13	
	Switched off	0	
Fail_a	Upscale, Inp_a (Inp_b) = x100_a (x100_b)	1	
Fail_b	Downscale, Inp_a (Inp_b) = x0_a (x0_b)	2	1
	Substitute value, Inp_a (Inp_b) = XaFail (XbFail)	3	
Xakorr	Measured value correction Inp_a (b) switched off	0	
Xbkorr	Measured value correction Inp_a (b) effective	1	0
Unit_a	Unit of the measured value of Inp_a (b) = °C	1	
Unit_b	Unit of the measured value of Inp_a (b) = °F	2	1
Mode	Inp_a and Inp_b: 2-wire connection	0	
	Inp_a: 3-wire connection no Inp_b	1	0
	Inp_a: 4-wire connection no Inp_b	2	
x0_a(b)	Physical value Inp_a (Inp_b) at 0%	Real	0
x100_a(b)	Physical value Inp_a (Inp_b) at 100%	Real	100
Xa(b)Fail	Substitute value with sensor error at Inp_a(b)	Real	0
Tfm_a(b)	Filter time constant of _a (Inp_b) in seconds	Real	0,5
Ka1_1a(b)	1st calibration value Inp_a(b) (read only)	Real	0
Ka1_2a(b)	2nd calibration value Inp_a(b) (read only)	Real	100

### 3.17.4. U\_IN (analog input card -50...1500mV, 0...10V)



Analog input, plugs into modular options card. The U\_INP is used for configuration and parameter setting of the analog input U\_INP. Input calculation is fixed to once per time slot.

#### Digital outputs:

ID_err	0 = correct module fitted 1 = wrong module fitted
fail_a	0 = no measurement error at channel a detected 1 = measurement error at channel a detected; e.g. sensor break
fail_b	0 = no measurement error at channel b detected 1 = measurement error at channel b detected; e.g. sensor break

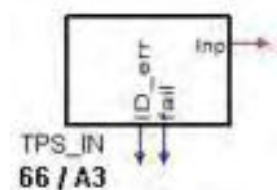
#### Analog outputs:

Inp_a	→ measured value channel a
Inp_b	→ measured value channel b

Parameter	Description	Werte	Default
x1a_in	Measured value correction Inp_a, P1 input value		0
x1aOut	Measured value correction Inp_a, P1 output value		0
x2a_in	Measured value correction Inp_a, P2 input value		100
x2aOut	Measured value correction Inp_a, P2 output value		100
x1b_in	Measured value correction Inp_b, P1 input value	Real	0
x1bOut	Measured value correction Inp_b, P1 output value		0
x2b_in	Measured value correction Inp_b, P2 input value		100
x2bOut	Measured value correction Inp_b, P2 output value		100

Configuration	Description	Value	Default
Typ_a	Voltage 0...10V	0	0
	Voltage -50...1500mV	1	
	Switched off	0	
Fail_a	Upscale, Inp_a = x100_a	1	1
	Downscale, Inp_a = x0_a	2	
	Substitute value, Inp_a = XaFail	3	
Xakorr	Measured value correction Inp_a switched off	0	0
	Measured value correction Inp_a effective	1	
Typ_b	Voltage 0...10V	0	0
	Voltage -50...1500mV	1	
	Switched off	0	
Fail_b	Upscale, Inp_b = x100_b	1	1
	Downscale, Inp_b = x0_b	2	
	Substitute value, Inp_b = XbFail	3	
Xbkorr	Measured value correction Inp_b switched off	0	0
	Measured value correction Inp_b effective	1	
a0_a	Physical value Inp_a at 0%	Real	0
x100_a	Physical value Inp_a at 100%	Real	100
XaFail	Substitute value with sensor error at Inp_a	Real	0
Tfm_a	Filter time constant of Inp_a in seconds	Real	0,5
x0_b	Physical value Inp_b at 0%	Real	0
x100_b	Physical value Inp_b at 100%	Real	100
XbFail	Substitute value with sensor error at Inp_b	Real	0
Tfm_b	Filter time constant of Inp_b in seconds	Real	0,5

### 3.17.5. TPS\_IN



#### Digital outputs:

ID_err	0 = correct module fitted 1 = wrong module fitted
fail	0 = no measurement error at channel a detected 1 = measurement error at channel a detected; e.g. sensor break

#### Analog output:

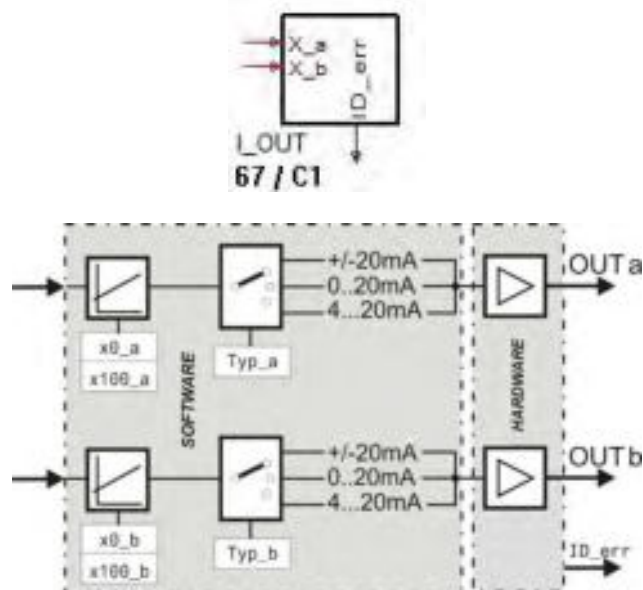
Inp	→ measured value channel a
-----	----------------------------

Configuration	Description	Values	Default
Typ_a	Typ L -200...900 °C	00	30
	Typ J -200...900 °C	01	
	Typ K -200...1350 °C	02	
	Typ N -200...1300 °C	03	
	Typ S -50...1760 °C	04	
	Typ R -50...1760 °C	05	
	Typ T -200...400 °C	06	
	Typ W(C) 0...2300 °C	07	
	Typ E -200...900 °C	08	
	Typ B 0...1820 °C	09	
	Typ D 0...2300 °C	10	
	Voltage 0...30mV	27	
	Voltage 0...100mV	28	
	Voltage 0...300mV	29	
	Standard signal 0...20mA	30	
	Standard signal 4...20mA	31	
	Switched off	0	
Fail	Upscale, Inp_a = x100	1	1
	Downscale, Inp_a = x0	2	
	Substitute value, Inp_a = XFail	3	
Xkorr	Measured value correction Inp_a switched off	0	0
	Measured value correction Inp_a effective	1	
Unita	Unit of the measured value of Inp_a = °C	1	1
	Unit of the measured value of Inp_a = °F	2	
STKa	Internal temperature compensation	1	1
	External temperature compensation	2	
x0	Physical value Inp_a at 0%	Real	0
x100	Physical value Inp_a at 100%	Real	100
XFail	Substitute value for sensor error at Inp_a	Real	0
Tfm	Filter time constant of Inp_a in seconds	Real	0,5
Tkref	Reference temperature for Inp_a at STK	Real	0

Parameter	Description	Values	Default
x1 in	Measured value correction Inp, P1 input value	Real	0
x10ut	Measured value correction Inp, P1 output value		0
x2 in	Measured value correction Inp, P2 input value		100
x20ut	Measured value correction Inp, P2 output value		100

**3.17.6. I\_OUT (analog output card 0/4...20mA, +/- 20mA)**

The I\_OUT is used for configuration and parameter setting of analog output I\_OUT. Output calculation is fixed to once per time slot.

**Digital output:**

ID\_err      0 = correct module fitted  
                  1 = wrong module fitted

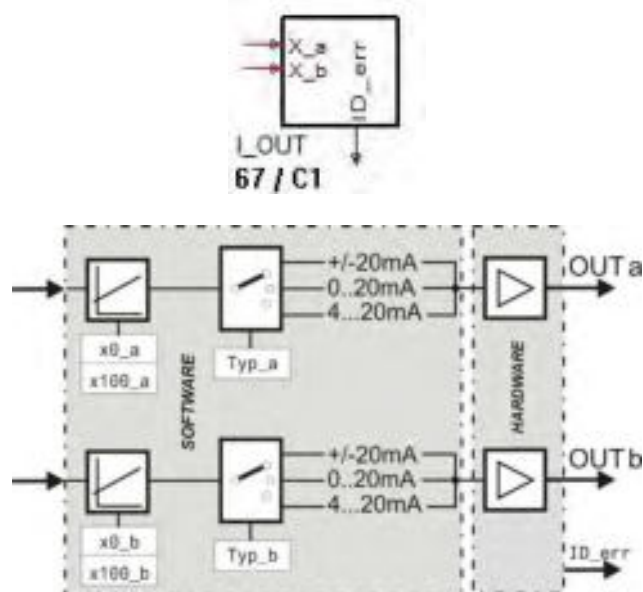
**Analog inputs:**

X\_a            → output value for channel a  
 X\_b            → output value for channel b

Configuration	Description	Values	Default
Typ_a	0...20mA	0	0
	4...20mA	1	
a0_a	Physical value Inp_a at 0%	Real	0
x100_a	Physical value Inp_a at 100%	Real	100
Typ_b	0...20mA	0	0
	4...20mA	1	
x0_b	Physical value Inp_b at 0%	Real	0
x100_b	Physical value Inp_b at 100%	Real	100

**3.17.7. U\_OUT (analog output card 0/2...10V, +/- 10V)**

The U\_OUT is used for configuration and parameter setting of analog output U\_OUT. Output calculation is fixed to once per time slot.

**Digital output:**

ID\_err      0 = correct module fitted  
               1 = wrong module fitted

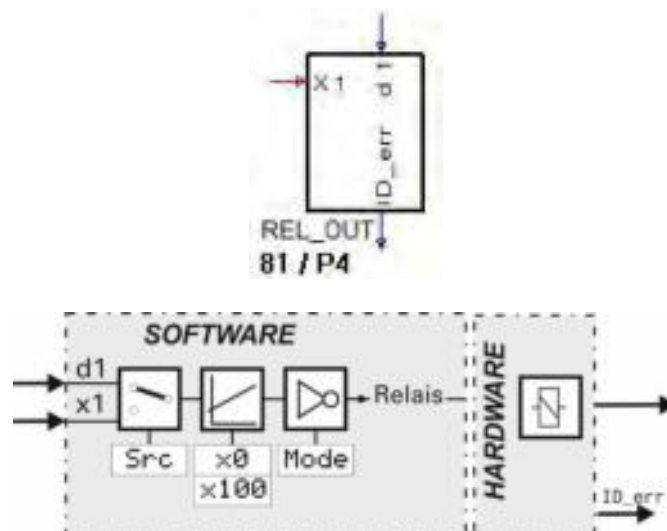
**Analog inputs:**

X\_a            → output value for channel a  
 X\_b            → output value for channel b

Configuration	Description	Values	Default
Typ_a	0...10V	0	0
	2...10V	1	
	+/-10V		
a0_a	Physical value Inp_a at 0%	Real	0
x100_a	Physical value Inp_a at 100%	Real	100
Typ_b	0...10V	0	0
	2...10V	1	
	+/-10V		
x0_b	Physical value Inp_b at 0%	Real	0
x100_b	Physical value Inp_b at 100%	Real	100

**3.17.8. REL\_OUT (Relais output)**

The REL\_OUT is used for configuration and parameter setting of relais output. Output calculation is fixed to once per time slot.

**Inputs / outputs****Digital input:**

d1 Input signal during digital signal conversion

**Analog input:**

x1 Input signal with analog signal conversion

**Digital output:**

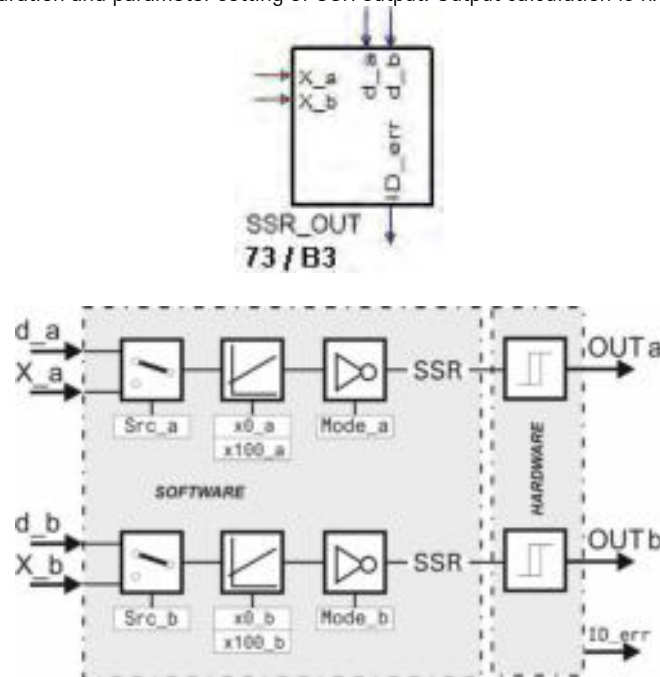
ID\_err 0 = correct module fitted  
1 = wrong module fitted

**Configuration parameter:**

Parameter	Description		Values	Default
Src	Signal source	digital input d1	Digital	←
		analog input x1	Analog	
Mode	Signal source action	direct/normally open	direct	←
		invers/normally closed	invers	
x0	Wert des analogen Eingangs x1 bei 0%		-29 999 ... 999 9990	0
x100	Wert des analogen Eingangs x1 bei 100%		-29 999 ... 999 999	100

**3.17.9. SSR\_OUT (Solid-State-Relais Ausgang)**

The SSR\_OUT is used for configuration and parameter setting of SSR output. Output calculation is fixed to once per time slot.

**Inputs / outputs****Digital inputs:**

d_a	Standard signal with digital signal conversion for channel a
d_b	Standard signal with digital signal conversion for channel b

**Analog inputs:**

X_a	Input signal with analog signal conversion for channel a
X_b	Input signal with analog signal conversion for channel b

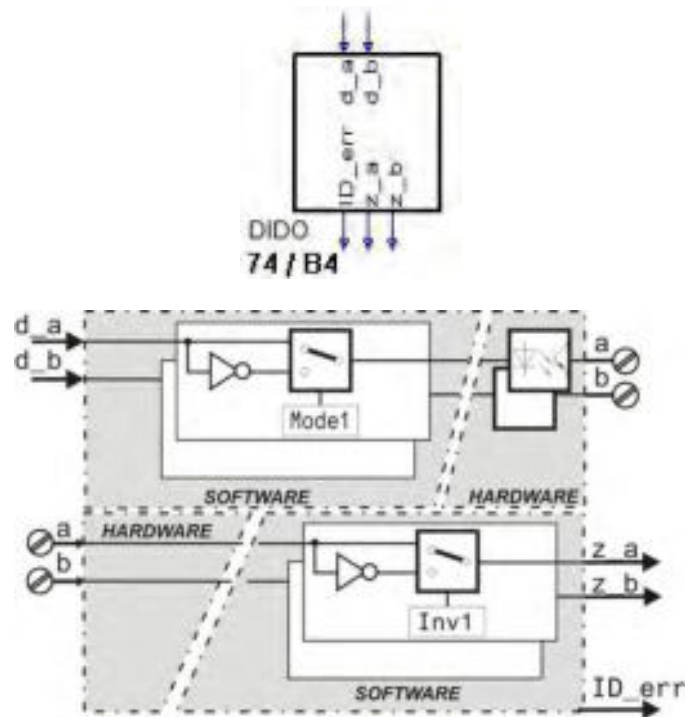
**Digital output:**

ID_err	0 = correct module fitted 1 = wrong module fitted
--------	--

**Configuration parameter:**

Parameter	Description	Values	Default
Src_a	Signal source channel a	Digital input d1	Digital ←
		Analog input x1	Analog
Mode_a	Signal source action channel a	direct/normally open	direct ←
		invers/normally closed	invers
Src_b	Signal source channel b	Digital input d2	Digital ←
		Analog input x2	Analog
Mode_b	Signal source action channel b	direct/normally open	direct ←
		invers/normally closed	invers
x0_a	Analog input value x_a at 0%	-29 999 ... 999 9990	0
x100_a	Analog input value x_a at 100%	-29 999 ... 999 999	100
x0_b	Analog input value x_b at 0%	-29 999 ... 999 9990	0
x100_b	Analog input value x_b at 100%	-29 999 ... 999 999	100



**3.17.10.DIDO (digital input/output card)**

Digital input/output card, plugs into modular options card C. The DIDO is used for configuration and parameter setting of digital inputs/outputs DIDO. Function block calculation is fixed to once per time slot.

**Ein- /Ausgänge****Digital inputs:**

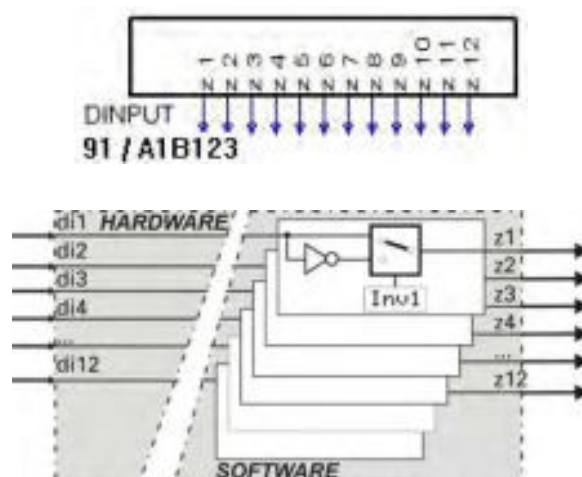
- d\_a → if configured as an output: hardware output a  
d\_b → if configured as an output: hardware output b

**Digital outputs:**

- ID\_err 0 = correct module fitted  
1 = wrong module fitted  
z\_a → status of hardware input a; if configured as an output: the output value read back  
z\_b → status of hardware input b; if configured as an output: the output value read back

**Configuration parameter:**

Configuration	Description	Values	Default
Inv_la	direct - HW- input a direct at z_a invers - HW- input a inverted at z_a	0 1	0
Inv_lb	direct - HW- input b direct at z_b invers - HW- input b inverted at z_b	0 1	0
Inv_Oa	direct - d_a direct on HW output a invers - d_a inverted at HW output a	0 1	0
Inv_Ob	direct - d_b direct on HW output b invers - d_b inverted at HW output b	0 1	0
Mode_a	Input - only HW input a at z_a Output - d_a at HW output a with feedback an z_a	0 1	0
Mode_b	Input - only HW input d_b at z_b Output - d_b at HW output b with feedback an z_b	0 1	0

**3.17.11.DINPUT (digital inputs (Nr. 121))**

Function 'DINPUT' is used for digital input configuration and parameter setting. The function is assigned firmly to block number 91 and is calculated invariably in each time slot. Inversion of each individual signal can be configured. If inputs di3...di12 are provided is dependent of the KS 98-2 hardware option.

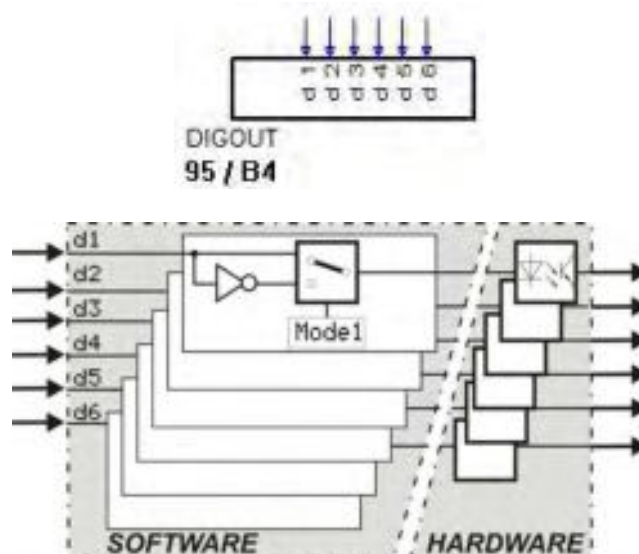
**Outputs****Digital outputs:**

z1 . . . z2 Signal at digital input di1 or di2 (Digital inputs di1 and di2 are available in each unit also without options)..

z3 . . . z12 Signal at digital input di3 or di12 (Digital inputs di3 to di12 are only provided with option "Digital I/O Extension").

**Parameter and configuration data**

Parameter	Description		Values	Default
Inv1	Transfer behaviour	direct output inverted output	direct invers	←
Inv2	Transfer behaviour	direct output inverted output	direct invers	←
.				
.				
Inv12	Transfer behaviour	direct output inverted output	direct invers	←

**3.17.12.DIGOUT (digital outputs (No. 122))**

Function 'DIGOUT' is used for digital output configuration and parameter setting. It is firmly allocated to block number 95 and is calculated invariably in each time slot. Inversion of each individual signal can be configured. If all digital outputs are provided is dependent the KS 98-2 hardware option.

**Inputs****Digital inputs:**

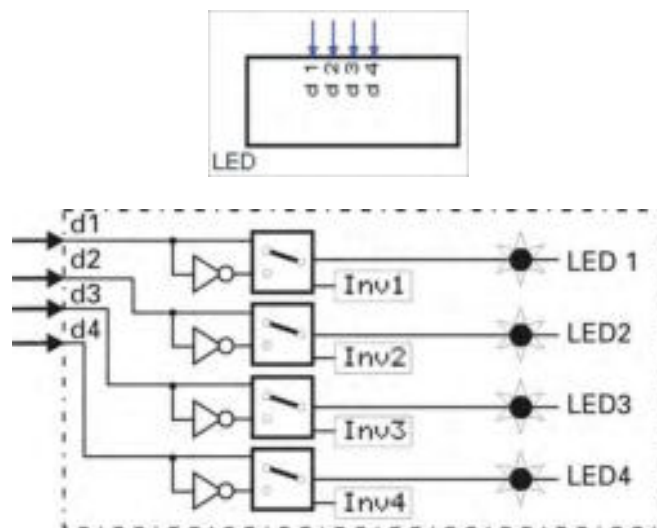
di1...di6      Signal sources for control of digital outputs do1 to do6.  
(only available in devices with the hardware option "Digital I / O extension").

**Parameter and configuration data**

Parameter	Description	Values	Default
Inv1	Transfer behaviour for d1	direct output inverted output	←
Inv2	Transfer behaviour for d2	direct output inverted output	←
.	.	.	.
Inv6	Transfer behaviour for d6	direct output inverted output	←

### 3.18. Additional functions

#### 3.18.1. LED (LED display (No. 123))



Function LED is used for control of the 4 LEDs. It is firmly allocated to block number 96 and is calculated in each time slot. The statuses of digital inputs d1 ...d4 are output to LED 1...LED 4. The statuses can be inverted via parameter Inv1 Inv4.

#### Inputs

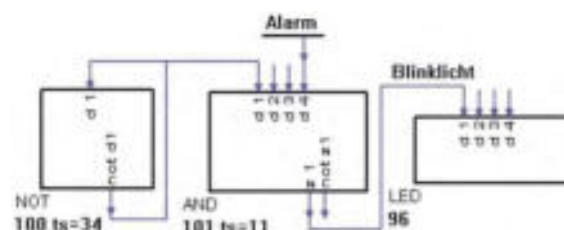
Inputs:	Description
d1	LED 1
d2	LED 2
d3	LED 3
d4	LED 4

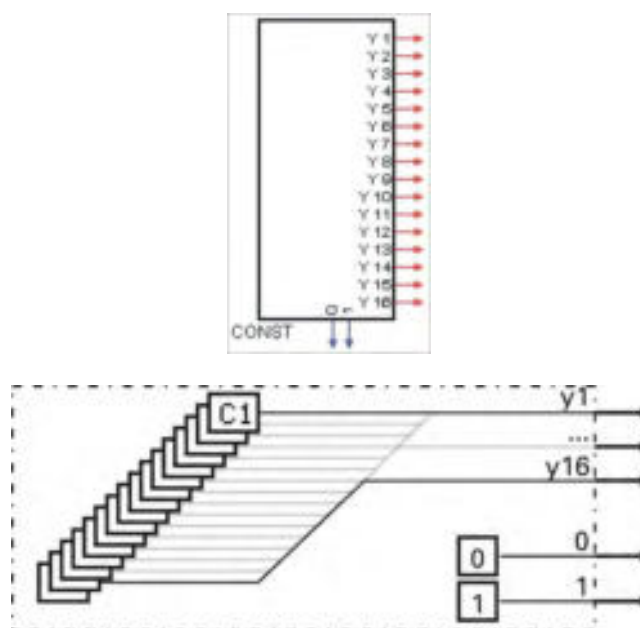
#### Parameter:

Parameter	Description	Values	Default
Inv 1	Inv1= 0 = d1=1 LED1 is lit Inv1 = 1 = d1= 0 LED1 is lit	0...1	0
Inv 2	Inv2= 0 = d2=1 LED2 is lit Inv2 = 1 = d2= 0 LED2 is lit	0...1	0
Inv 3	Inv3= 0 = d3=1 LED3 is lit Inv3 = 1 = d3= 0 LED3 is lit	0...1	0
Inv 4	Inv4= 0 = d4=1 LED4 is lit Inv4 = 1 = d4= 0 LED4 is lit	0...1	0

#### Example:

If a simple flashing function is to be produced, this is possible with the following example. The sampling-time period code of the NOT-function indicates the flash frequency.



**3.18.2. CONST (constant function (No. 126))**

16 analog constants at output y1...y16 and logic statuses 0 and 1 are made available. The block number is firmly configured with 99.

**Outputs:****Digital outputs**

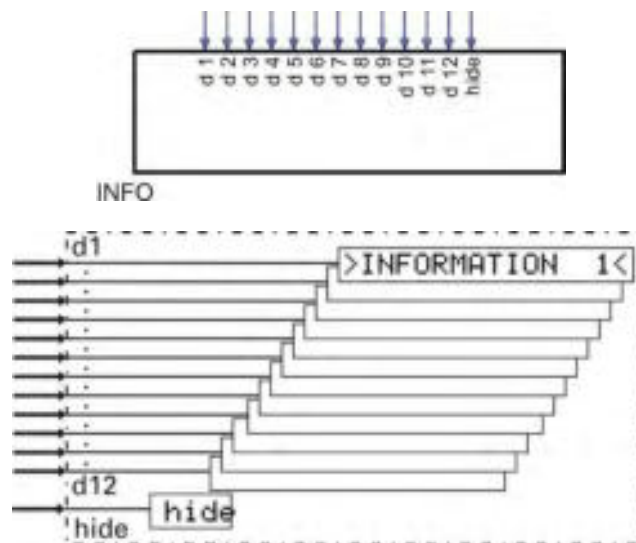
0	Logic 0 is always output at this output.
1	Logic 1 is always output at this output.

**Analog outputs**

Y1	Constant C1 is output.
Y2	Constant C2 is output.
Y3	Constant C3 is output.
Y4	Constant C4 is output.
Y5	Constant C5 is output.
Y6	Constant C6 is output.
Y7	Constant C7 is output.
Y8	Constant C8 is output.
Y9	Constant C9 is output.
Y10	Constant C10 is output.
Y11	Constant C11 is output.
Y12	Constant C12 is output.
Y13	Constant C13 is output.
Y14	Constant C14 is output.
Y15	Constant C15 is output.
Y16	Constant C16 is output.

**Parameters:**

Parameters	Description	Values	Default
C1..C16	Analog constants	-29 999...999 999	0

**3.18.3. INFO (information function (No. 124))**

This function can be used for display of 12 user texts with max. 16 characters each by setting the relevant input d1...d12. The information is displayed in the "header" of operating pages (level 1 data) in alternation with the description of the called up operating page. If several texts are available simultaneously, they are displayed successively.

The block number is fixed to 97 and calculated once per time slot.

The user texts are displayed on the operating pages and the operating page list.

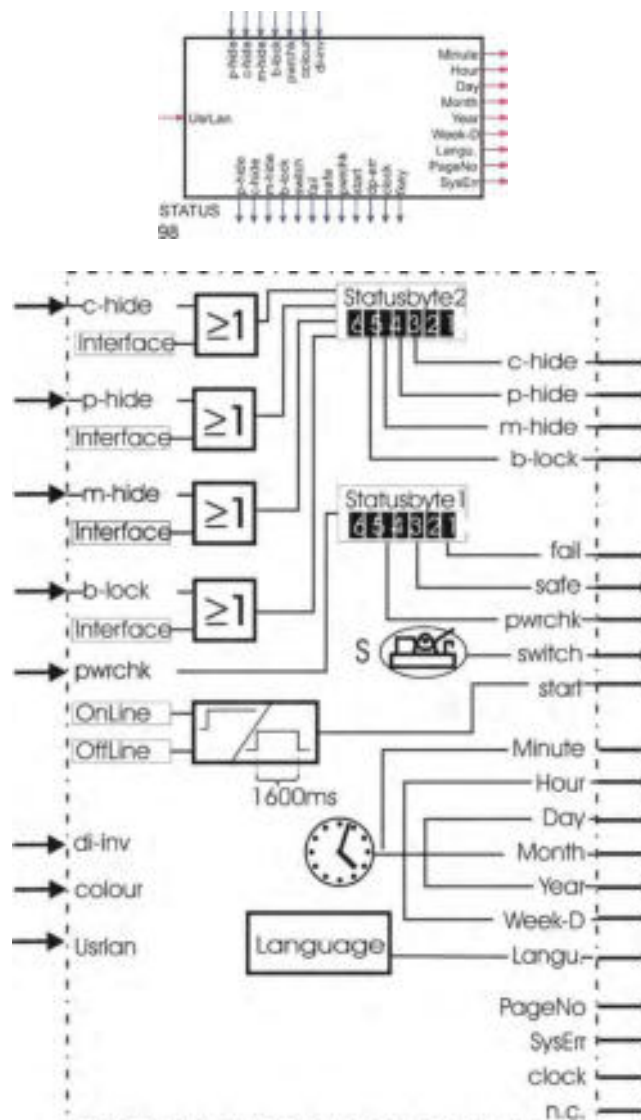
Display of all INFO texts can be suppressed by setting the Hide signal.

**Inputs:****Digital inputs**

d1	= 1 → the information configured in <b>Text 1</b> is displayed.
d2	= 1 → the information configured in <b>Text 2</b> is displayed.
d3	= 1 → the information configured in <b>Text 3</b> is displayed.
d4	= 1 → the information configured in <b>Text 4</b> is displayed.
d5	= 1 → the information configured in <b>Text 5</b> is displayed.
d6	= 1 → the information configured in <b>Text 6</b> is displayed.
d7	= 1 → the information configured in <b>Text 7</b> is displayed.
d8	= 1 → the information configured in <b>Text 8</b> is displayed.
d9	= 1 → the information configured in <b>Text 9</b> is displayed.
d10	= 1 → the information configured in <b>Text 10</b> is displayed.
d11	= 1 → the information configured in <b>Text 11</b> is displayed.
d12	= 1 → the information configured in <b>Text 12</b> is displayed.
hide	= 1 → all INFO texts are hidden, i.e. not displayed.

**Parameters:**

Parameter	Description	Values	Default
Text1	User text with max. 16 characters each	alphanumeric characters	>INFORMATION 1 <
...			...
Text12			>INFORMATION 12 <

**3.18.4. STATUS (status function (No. 125))**

The function provides information from the KS 98-1 instrument status byte at its digital outputs. The block is fixed to 98 and updated per time slot.

**Inputs****Digital Inputs**

- c-hide = 1 → configuration change via operation is disabled.
- p-hide = 1 → parameters/configurations via operation disabled
- m-hide = 1 → main menu is not displayed, operating pages are displayed only during online mode
- b-block = 1 → use of the bus interface is blocked
- pwrchk = 1 → Monitoring for temporary power failure is activated (→ See output pwrchk).
- colour green = 0, red = 1. → display background colour is changed.
- di-inv Display is inverted (background / text & graphics).

**Analog inputs**


- UsrLan Change to user language. Switch-over between text blocks connected via the language input. Three languages are selectable with 0 .. 2.

## Outputs

### Analog output

Minute	Minute of the real-time clock 0...59
Hour	Hour of the real-time clock 0...23
Day	Day of the real-time clock 0...31
Month	Month of the real-time clock 1...12
Year	Year of the real-time clock 1970...2069
Week-D	Weekday of the real-time clock 0...6 $\triangle$ Su...Sa
Langu	Language German = 0 language English = 1 Language selection is in
<b>Miscellaneous, Device data</b>	
PageNo	Output of the number of the function block the operating page of which is displayed instantaneously. "0" means that no operating page is displayed.
SysErr	Start-up problem. Unless this output is "0", a start-up error occurred (corresponds to KS 98-1 start-up error display). Bit assignment: Bit 1 = reset command, bit 2 = quartz, bit 4 = halt, bit 5 = SW watchdog (endless loop); the other bits are not used.

### Digital outputs

c-hide	= 1 $\rightarrow$ configuration change disabled
p-hide	= 1 $\rightarrow$ parameters/configurations disabled
m-hide	= 1 $\rightarrow$ The main menu is not displayed, the operating pages are displayed only during online mode
b-block	= 1 $\rightarrow$ the use of the bus interface is blocked
switch	DH-Swi = aus $\triangle$ swi tch = 0; DH-Swi = ein $\triangle$ swi tch = 1 . This information permits blocking via the „hardware“.
fail	= 1 $\rightarrow$ common message sensor error of inputs AINP1...AINP6
safe	= 1 $\rightarrow$ Sicherheitszustand gesetzt über Schnittstelle mit Code 22, Fbnr. 0, Fktnr. 0
pwrchk	Power-fail check. This value is always reset (0) by KS 98-1 after power-on. It can be set to active (1) by a signal at the digital input to permit detection of a temporary power failure.
start	With change from offline to online, start is 1 during 800 ms. During this time, all time groups were calculated at least once.
dperr	Collected error messages Profibus
clock	1 = real-time clock provided, 0 = no real-time clock option.
fkey	Status of the function key 

## Powerup KS98

### Start-up behaviour after power recovery

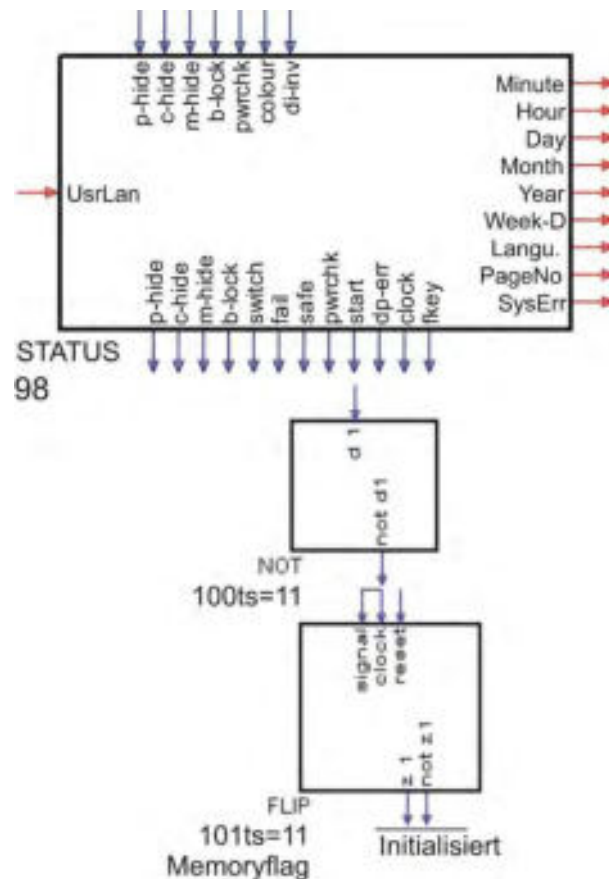
The procedure is as follows:

- ① All blocks are initialized. Unless a special start-up behaviour is configured (see programmer), the two possibilities are:
  - a. The memory contents are still unchanged. The block outputs keep the value before power failure.
  - b. After prolonged power failure, the memory contents are destroyed. The function blocks are initiated regardless of the function inputs.
- ② All input functions are calculated once.
- ③ The start bit of the status block is set to 1.
- ④ All blocks are calculated at intervals of 1.6 seconds (16 cycles of the 100ms time slot) in the order of block numbers.
- ⑤ The start bit of the status block is set to 0.



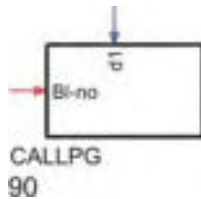
In the event of problems due to the start-up sequence, 2 items can be of importance:

- ① After power failure, KS 98 /98-1 continues running during fractions of seconds and may still detect already de-activated process signals.
- ② If the behaviour after power recovery is different depending on whether the function blocks are still in the previous condition or whether they were initialized, the following engineering can provide the initialization information within the first 1.6 seconds after starting up.



Another possibility for power failure detection is to set an internal flag in the status block via digital input "pwrchk" at the end of the initialization phase. This Flag is available at digital output "pwrchk". After power failure, this flag and thus digital output "pwrchk" is always initialized with 0.

### 3.18.5. CALLPG (Function for calling up an operating page (no. 127))



With function block CALLPG, which can be used only once, event-triggered call-up of a particular operating page is possible, unless an operation is being carried out on this page (waiting time 5 s). The required operating page is determined by the number of its function block. The block number is applied to input BI-no of CALLPG.


Switch-over is with the positive edge of the logic signal at digital input d1 of CALLPG. It permits e.g. changing to a particular operating page in case of an exceeded limit value.


Exceptions: Switching over is omitted with:


- active operation by the operator. Page changing is delayed and occurs only 5 seconds after the last key pressure.
- a wrong page number, or if the page is blocked at activation time.

Unless the page which should be activated is available the page survey is displayed. When leaving the operating page called up via CALLPG, the previously active operating page is displayed again.

Following functions blocks have an operating page APROG, DPROG, CONTR, CONTR+, PIDMA, VVWERT, VBAR, VTREND, VPARA, ALARM

 With activation by CALLPG from an already selected page, this page is not called up again. I. e. if a sub-page was selected, the multifunction unit remains on this page.

 With multiple page changing by activation of CALLPG, the sequence is not buffered. After leaving the page(s) activated by CALLPG the initial menu page is displayed again.

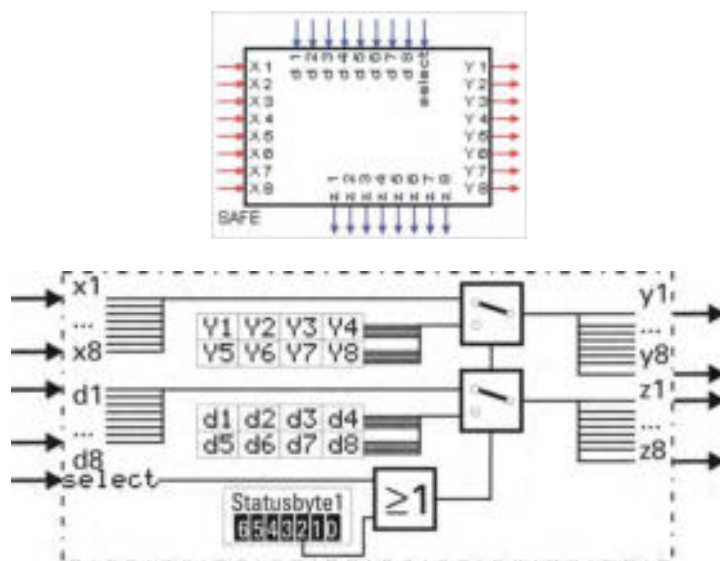
 In case of CALLPG whilst the multifunction unit is not at operating level (main menu: Parameter, ..., Miscellaneous), CALLPG call remains active in the background. Changing to the operating page activated by CALLPG is done when selecting the operating level for the next time.

#### Digital input

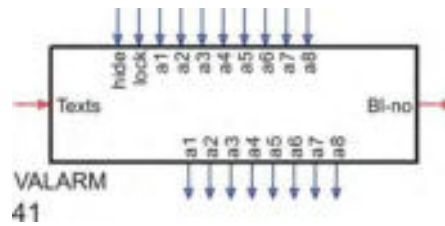
d1 Positive edge causes change to the operating page set at BI-no

#### Analog input

BI-no Number of the operating page which should be displayed

**3.18.6. SAFE (safety function (Nr. 94) )**

Function SAFE is used for generation of defined analog output values and digital statuses dependent of digital input select or of the status received via the interface. In the normal case select = 0 and status = 0, the values applied to the inputs are switched through to the outputs without change. For select = 1 or status = 1, configured data z1...z8 and y1...y8 are switched through to the outputs.

**3.18.7. VALARM (display of all alarms on alarm operating pages (function no. 109))****General**

Function block VALARM handles up to 8 alarms. Alarms are displayed and can be acknowledged, if acknowledgement via parameter setting is required. The alarm conditions are determined by digital inputs a1 ... a8 (0 alarm condition off, 1 alarm condition on).

**Inputs/outputs****Digital inputs:**

hide	Display of this alarm operating page is suppressed
lock	Operation of this operating page is disabled, i.e. alarm acknowledgement is not possible
a1...a8	Alarm inputs alarms 1 ... 8

**Analog output**

BL-no	Number of this block
-------	----------------------

**Digital outputs**


a1...a8	= 1 means that alarm 1 must be acknowledged
---------	---

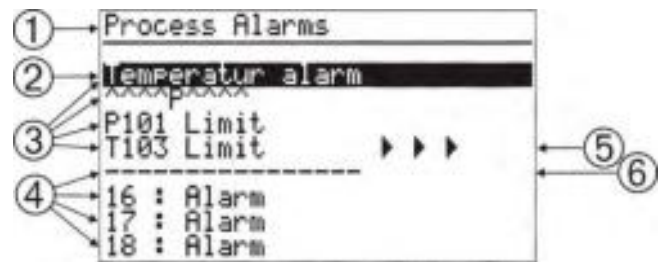
**Parameter und configuration data****Parameter**

a1 ... a8	=1 means that alarm 1 must be acknowledged
-----------	--

Parameter	Beschreibung	Werte	Default
Typ_a1...Typ_a8	Alarm function	Quit noQuit	←

## VALARM operating page

- ① Title
- ② Active alarm selected for acknowledging
- ③ Active alarms with texts from the TEXT-function
- ④ Active alarms with standard texts
- ⑤ Alarm no more active, but not acknowledged
- ⑥ Alarm no more active (if the display is rebuilt with -key, this alarm is no more displayed)



Several alarm blocks can be positioned. Block numbers 41-46 are available for this purpose. When using several alarm blocks, display of all alarm blocks except one should be suppressed, because all alarms including those of the blocks which are not selected are listed on each VALARM operating page.


The operating page title indicates the instantaneously selected block. With alarm blocks, user language selection related to the title is possible by enabling another block with a language-dependent title dependent on selected user language.


In order to avoid disturbance of the user-specific menu structure, the alarm page is displayed at the end of the operating page list independent of its block number.

Lines on the alarm page contain entries corresponding to the following classification

- No alarm: not existing or marked as deleted "———" refreshed at next page actualization
- Alarm active: Line blinks on the operating page
- Alarm aktiv und quittiert : Normale Darstellung auf der Bedienseite
- Alarm nicht aktiv, Quittierung fehlt: Normale Darstellung mit "> >" am Ende der Zeile

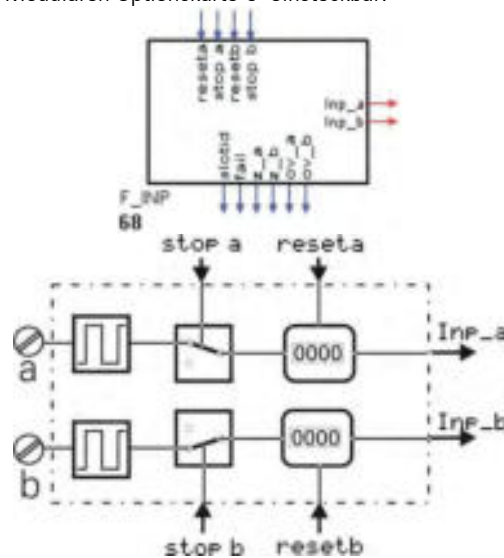
The alarms are displayed with the definable name in the order of occurrence. The displayed name is taken from two text blocks which should be connected with the ALARM block. Without connected text blocks, the alarm number is displayed. The alarm numbers are calculated from block number -40 and the number of the digital input. The block numbers are 41-46, i.e. the 3rd alarm in block 41 (1st block) becomes 13.

To prevent alarm position changing, "———" is displayed for disappearing alarms. New pending alarms are displayed only when rebuilding up the page. Build-up can be started also by pressing key .

-  The combination of digital input and output signals indicates the four statuses of an alarm: active + not acknowledged (di=1, do=1), active and acknowledged (di=1, do=0), not active any more and not acknowledged (di=0, do=1), and not active or not active any more and acknowledged (di=0, do=0).

### 3.18.8. F\_Inp (Frequenz-/ Zählereingang)

Der Frequenz-/ Zählereingang ist auf der Modularen Optionskarte C einsteckbar.



Die Funktion F\_INP dient zur Konfigurierung und Parametrierung des Einganges F\_INP. Der Eingang wird fest einmal pro Zeitscheibe berechnet.

#### Digitale Eingänge:

- reset a** → 1 = der Wert für **Inp\_a** wird zurück auf 0 gesetzt.
- stop a** → 1 = der momentane Wert für **Inp\_a** bleibt unverändert erhalten.
- reset b** → 1 = der Wert für **Inp\_b** wird zurück auf 0 gesetzt.
- stop b** → 1 = der momentane Wert für **Inp\_b** bleibt unverändert erhalten.

#### Digitale Ausgänge:

- slotid** → 0 = korrektes Modul eingesteckt  
→ 1 = falsches Modul eingesteckt
- fail** → 1 = eingestecktes Modul wird erkannt, aber keine Kommunikation zum Modul.
- z\_a** → Signalzustand von HW - Eingang a
- z\_b** → Signalzustand von HW - Eingang b
- ov\_a** → 1 = Frequenz am HW - Eingang a ist größer als die maximal zugelassenen 20kHz
- ov\_b** → 1 = Frequenz am HW - Eingang b ist größer als die maximal zugelassenen 20kHz

#### Analoge Ausgänge:

- Inp\_a** → Ausgabewert für Kanal a
- Inp\_b** → Ausgabewert für Kanal b

Konfiguration	Beschreibung	Werte	Default
Func_a	DigInput → Steuereingang	0	1
	Count_1 → Vorwärtszähler	1	
	Count_2 → Vor-/Rückwärtszähler	2	
	Count_3 → Vor-/Rückwärtszähler mit Richtungssignal	3	
	Count_4 → Quadraturzähler	4	
Func_b	Frequenz → Frequenzmessung	5	1
	DigInput → Steuereingang	0	
	Count_1 → Vorwärtszähler	1	
Time	Frequenz → Frequenzmessung	5	10
	für Frequenzmessung in Sekunden	0,1...20	

### 3.19. Function management

Max. 2000 function blocks can be used. Each function requires a defined portion of the working memory and a defined calculation time. The used-up resources can be examined in the engineering Tool under Help / Statistics.

#### 3.19.1. Memory requirement and calculation time

Function	Time %	Memory %	Function	Time %	Memory %	Function	Time %	Memory %
Scaling and calculating functions			DELA2			KS8x		
ABSV			FILT			CPREAD		
ADSU			TIMER			CPWRIT		
MUDI			TIME2			CSDO		
SQRT								
SCAL			Selecting and storage			Programmer		
10EXP			EXTR			APROG		
EEXP			PEAK			APROGD		
LN			TRST			APROGD2		
LG10			SELC			DPROG		
			SELD			DPROGD		
Non-linear functions			SELP					
LINEAR			SELV1			Controller		
GAP			SOUT			CONTR		
CHAR			REZEPT			CONTR+		
			2OF3			PIDMA		
Trigonometric functions			SELV2					
SIN						Modular Input-/Output-option		
COS			Limit value signalling / limiting			TC_IN		
TAN			ALLP			R_IN		
COT			ALLV			U_IN		
ARCSIN			EQUAL			TPS_IN		
ARCCOS			VELO			UNI_IN		
ARCTAN			LIMIT					
ARCCOT			ALARM			I_OUT		
						U_OUT		
Logic functions			Visualization			REL_OUT		
AND			TEXT			SSR_OUT		
NOT			VWERT					
OR			VBAR			Additional functions		
EXOR			VPARA			LED		
BOUNCE			VTREND			CONST		
FLIP						INFO		
MONO			Communication			STATUS		
STEP			L1READ			CALLPG		
TONOFF			L1WRITE			SAFE		
			DPREAD			VALARM		
Signal converters			DPWRIT					
A2BYTE								
ABIN			KS98-2 CANopen					
TRUNC			C_RM2x					
PULS			RM_DI					
COUN			RM_DO					
MEAN			RM_AI					
			RM_DMS					
Time functions			RM_AO					
LEAD			CRCV					
INTE			CSEND					
LAG1			C_KS98x					
DELA1								

### 3.19.2. Sampling intervals

Inputs and outputs are processed every 100 ms

Calculation of the other function blocks is at equal intervals according to their allocation to the 8 time slots of 100 ms each. Allocation of a block to one or several time slots (at intervals of 100, 200, 400 or 800 ms) is in the engineering.

For each block, the engineering tool provides an identification (ts) which can be used to determine the allocation from the table opposite.

The total of calculation times of all required function blocks must be < 100 % for each time slot.

ts	Zeitscheibe								Abtastzeit
	1	2	3	4	5	6	7	8	
11	X	X	X	X	X	X	X	X	alle 100 ms
21	X	-	X	-	X	-	X	-	alle 200 ms
22	-	X	-	X	-	X	-	X	alle 200 ms
31	X	-	-	-	X	-	-	-	alle 400 ms
32	-	X	-	-	-	X	-	-	alle 400 ms
33	-	-	X	-	-	-	X	-	alle 400 ms
34	-	-	-	X	-	-	-	X	alle 400 ms
41	X	-	-	-	-	-	-	-	alle 800 ms
42	-	X	-	-	-	-	-	-	alle 800 ms
43	-	-	X	-	-	-	-	-	alle 800 ms
44	-	-	-	X	-	-	-	-	alle 800 ms
45	-	-	-	-	X	-	-	-	alle 800 ms
46	-	-	-	-	-	X	-	-	alle 800 ms
47	-	-	-	-	-	-	X	-	alle 800 ms
48	-	-	-	-	-	-	-	X	alle 100 ms

### 3.19.3. EEPROM data

Data are stored in non-volatile EEPROM. The manufacturers specify approx. 100 000 permissible write cycles per EEPROM address, in reality, however, this value can mostly be exceeded by a multiple. If parameters and configurations are changed exclusively manually, exceeding the max. number of write cycles is almost precluded. With digital interface or automatic parameter changes, however, taking the maximum number of write cycles into account is indispensable, and measures against excessively frequent parameter writing must be taken.



## 3.20. Examples

During installation of the engineering tools, several examples were included. These are in path:  
C:\Pmatools\Et98\prj\example.

### 3.20.1. Useful small engineeringings

#### ***Cascaded counter with pulse generator***

**(ZAEHLER.EDG)**

An INTE is used for generating pulses. Max. parameter =1, time constant to 3600 sec. An input value at x1 of e.g. 20 weighted via the MUDI generates 20 pulses per hour. The first counter counts to 1000, the second counter counts the (1000s) overflows

#### ***Simple password function***

**(PASSWORD.EDG)**

A VVERT is used for password entry. The output is not fed back to the input, for suppressing display of the entered value after pressing the enter key. The current hour of the status block is used as a password (only with clock). The EQUAL block determines the condition for disabling the parameter level.

#### ***Password from the CONST block***

**(PASSWORD.EDG)**

A VVERT is used for password entry. The output is not fed back to the input, for suppression of the display of the entered value after pressing the enter key. A value of the constant block is used as a password. The EQUAL block determines the condition for disabling the parameter level and display suppression of the VVERT page.

#### ***Macro for dynamic alarm processing***

**(ALARMSEL.EDG)**

ASELV2 can be used to select one of 4 values for alarm processing. An ALLV compares the value with an upper and a lower limit definable via a VVERT. The alarms are displayed at the second VVERT and output to a relay via an OR. Each of the two VVERT can define or display two further alarm limits. Therefore, the configuration can be extended by another ALLV. As an example, possible alarm acknowledgement via a flipflop is provided. Alarms are held in the LED display and the alarm line until acknowledgment via the VVERT (alarms).

#### ***Alarm acknowledgement of 5 alarm bits***

**(ALAMQUIT.EDG)**

The flipflops hold the alarms individually until acknowledgement via the VVERT. The acknowledge output is fed back to the Store input instead of the corresponding input bit. Thereby, the acknowledge bit is reset automatically.

#### ***Alarm acknowledgment of 5 alarm bits that are not lost even after a longer power failure***

**(ALQITS.EDG)**

Flipflops are also used for storing. In this case, the status change of the flipflop must be stored in non-volatile recipe blocks. Moreover, the flipflops must be loaded with the content of the recipe block for restoring the last status after power recovery. In VVERT, the alarms are displayed and acknowledged, if necessary. Further display via LED, DIGOUT and INFO.

#### ***Parameter number display via texts***

**(PRNRE.EDG)**

The current parameter number (variable in VVERT) is compared with constants via EQUAL. With equality, a bit at VVERT is set, whereby a digital text is displayed.

#### ***Two-point operation of a programmer***

**(RUNFLIP1.EDG)**

As entry of commands via the operating page is not possible with a programmer, if the relevant digital inputs were connected, the toggle key (fkey:a/m) must be used for realizing the Run/Stop order on the operating page. A monoflop generates a short pulse on the positive and negative flank.

The external command (key or switch) from the control panel via d1 is also taken via a monoflop. With a key, only d1 (positive flank) is connected, with a switch, d1 and d2 are connected (positive and negative flank). The pulses are taken to a flipflop, which switches over between Run and Stop.

#### ***Weekly timer for a switch-on and a switch-off time***

**(SCHALTUHR.EDG)**

Prerequisite: options card B with clock. 3 ADSUs convert the day, hour, minute information from the status block and the switch-on/switch-off time from VVERT into a minute value. If the time from the status block is higher than the switch-on time, the flipflop is set, if the time is higher than the switch-off time, the flipflop is reset.

### **Recipe input via VVERT**

**(REZEPT2.EDG)**

Three configuration examples with different restrictions for operation.

The VVERT displays its own outputs, but not the actually selected recipe. Editing of an existing recipe is not possible. The VVERT displays the selected recipe, but only, when storage was done after editing. The current values disappear after pressing the Enter key. VVERT has an additional edit function. This bit was applied to the manual input of the recipe block for output of the currently changed values on the display via the operating page. When storing and switching over to the next the recipe number (ALLP), the edit mode is reset automatically via OR and AND (due to the handling order).

### **Limitation of the input value (operation) with the VVERT**

**(VVERT BEGRENZUNG.EDG)**

By using an ALLP block with appropriate limits, these limits are also used for the value of the VVERT block!

## **3.20.2. Controller applications**

### **Minimum controller configuration**

**(C\_SINGL.EDG)**

### **Ratio controller with split-range or three-point stepping controller with position feedback**

**(C\_V\_SPL.EDG)**

The position feedback input is defined as a potentiometric transducer (which can be calibrated) and linked to the controller with its fail, a/m, inc, dec outputs. The use of process outputs can be configured at the controller and OUT1/OUT2.

### **Slave controller for testing the start of internal switching functions**

**(C\_SW\_SL.EDG)**

### **Circuit proposal for cascade configurations**

**(KASK.EDG)**

The master controller correcting variable must follow the slave or process value, when the slave is switched to internal or manual mode, in order to ensure bumpless return to the automatic mode.

### **Setpoint selection via an analog value**

**(W\_SELECT.EDG)**

An analogue value is used to select a setpoint that can be set via VVERT pages. The second VVERT page is hidden in the main menu.

## **3.20.3. Programmer fragments**

### **Analog output with 4 recipes (2\*20 segments 2\*10 segments)**

**(PROG.EDG)**

Selection of the recipe/program no. is via the VVERT and cannot be selected any more via the programmer operating page. The ALLP limits the input range.

Caution: the display is correct, however, the edit buffer contains the last output value, which may be too high. Entry of the preset time is via the programmer operating page. For input of the preset time via a VVERT, the digital connection (PRESET) must be provided.

### **Programmer with coupled outputs**

**(PROG2.EDG)**

The programmer blocks are coupled for program number, elapsed net time and RUN / RESET commands.

### **Programmer output with 10 programs with 20 segments**

**(PROGRAM.EDG)**

### **Analog programmer with two-point operation**

**(PROG\_FK.EDG)**