Concept
Block library IEC
Part: CONT_CTL

Volume 1

840 USE 504 00 eng Version 2.6

33002211.00

Table of Contents

The chapters marked gray are not included in this volume.

About the book

At a Glance

Document Scope

This documentation will assist you when configuring functions and Function blocks.

Validity Note

This document applies to Concept 2.5 under Microsoft Windows 98, Microsoft Windows 2000, Microsoft Windows XP and Microsoft Windows NT 4.x.

Note: Additional up-to-date tips can be found in the README data file in Concept.

Related Documents

Title of Documentation	Reference Number
Concept Installation Instructions	840 USE 502 00
Concept User Manual	840 USE 503 00
Concept-EFB User Manual	840 USE 505 00
Concept LL984 Block Library	840 USE 506 00

User Comments

We welcome your comments about this document. You can reach us by e-mail at TECHCOMM@modicon.com

General information about the block library CONT_CTL

I

Overview

At a glance

This section contains general information about the block library CONT_CTL.

What's in this part?

This part contains the following chapters:

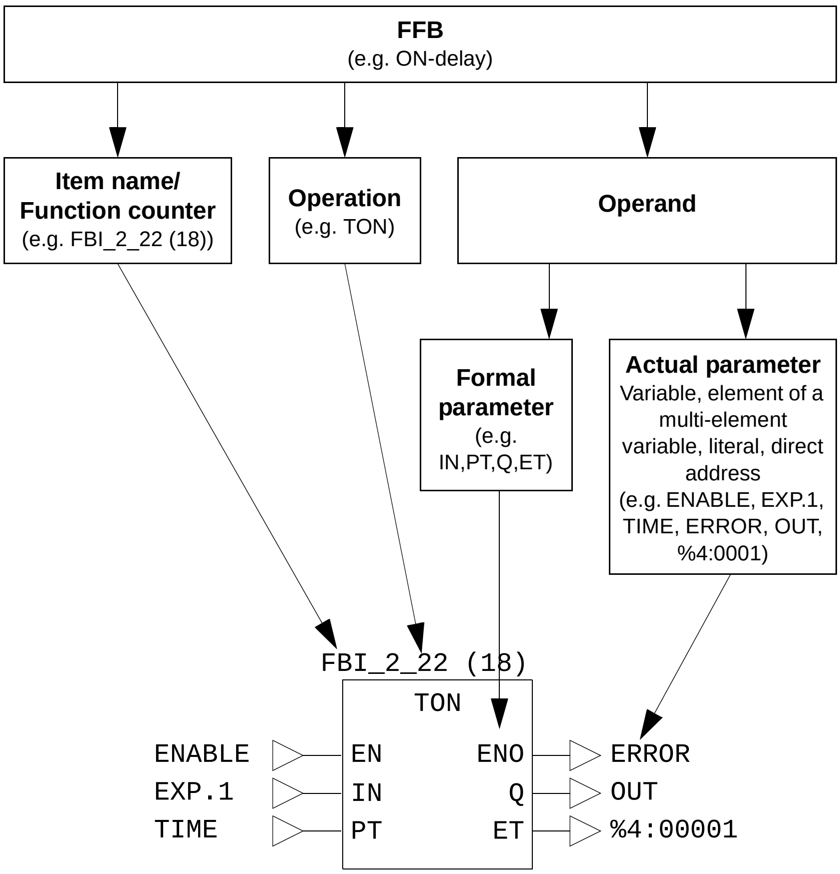
Parameterizing functions and function blocks

1

Parameterizing functions and function blocks

General

Each FFB consists of an operation, the operands needed for the operation and an instance name or function counter.



Operation

The operation determines which function is to be executed with the FFB, e.g. shift register, conversion operations.

Operand

The operand specifies what the operation is to be executed with. With FFBs, this consists of formal and actual parameters.

Formal/actual parameters

The formal parameter holds the place for an operand. During parameterization, an actual parameter is assigned to the formal parameter.

The actual parameter can be a variable, a multi-element variable, an element of a multi-element variable, a literal or a direct address.

Conditional/ unconditional calls

"Unconditional" or "conditional" calls are possible with each FFB. The condition is realized by pre-linking the input EN.

- Displayed EN
  conditional calls (the FFB is only processed if EN = 1)
- EN not displayed
  unconditional calls (FFB is always processed)

Note: If the EN input is not parameterized, it must be disabled. Any input pin that is not parameterized is automatically assigned a "0" value. Therefore, the FFB should never be processed.

Calling functions and function blocks in IL and ST

Information on calling functions and function blocks in IL (Instruction List) and ST (Structured Text) can be found in the relevant chapters of the user manual.

General information on the CONT_CTL block library

2

Introduction

At a glance

This section contains general information on the CONT_CTL block library.

What's in this chapter?

This chapter contains the following topics:

Groups in the CONT_CTL block library

Overview of the groups

The "Continuous Control"(CONT-CTL) library consists of 7 groups with Elementary function blocks (EFBs):

Groups	Contents
CLC	Contains closed loop control function blocks such as filters, controllers, integrators and Deadtime devices
CLC_PRO	Contains a further selection of closed loop control function blocks
Conditioning	EFBs for processing the measurement or another discrete variable
Controller	Controller EFBs and automatic closed control loop blocks
Mathematics	EFBs for mathematical control functions
Output Processing	EFBs for controlling the various actuator types
Setpoint Management	EFBs for generating and selecting the setpoint

"CLC" group

This group contains the following EFBs:

Block	Meaning
DELAY	Deadtime device
INTEGRATOR1	Integrator with limit (Operating modes, Manual, Halt, Automatic)
LAG1	Time lag device: 1st order
LEAD_LAG1	PD device with smoothing
LIMV	Velocity limiter: 1st order
PI1	PI controller
PID1	PID controller
PIDP1	PID controller with parallel structure
SMOOTH_RATE	Differentiator with smoothing
THREEPOINT_CON1	Three point controller
THREE_STEP_CON1	Three-step step-action controller
TWOPOINT_CON1	Two-step controller

"CLC_PRO" group

This group contains the following EFBs:

Block	Meaning
ALIM	Velocity limiter: 2nd order
COMP_PID	Complex PID controller
DEADTIME	Deadtime device
DERIV	Differentiator with smoothing
FGEN	Function generator
INTEG	Integrator with limit
LAG	Time lag device: 1st order
LAG2	Time lag device: 2nd order
LEAD_LAG	PD device with smoothing
PCON2	Two-step controller
PCON3	Three point controller
PD_or_PI	Algorithm-adaptive PD/PI controller
PDM	Pulse duration modulation
PI	PI controller
PID	PID controller
PID_P	PID controller with parallel structure
PIP	PIP cascade controller
PPI	PPI cascade controller
PWM	Pulse width modulation
QPWM	Pulse width modulation (simple)
SCON3	Three-step step-action controller
VLIM	Velocity limiter: 1st order

"Conditioning" group

This group contains the EFBs for processing procedures which come before the controllers in general, such as the processing of the measurements of the controlled variables, the disturbance variables or other discrete variables.
This group also contains delay and summation functions beyond filters and other classic functions.
This group contains the following EFBs:

Block	Meaning
DTIME	Delay function, for increased precision or for dynamic (online) modification of the delay value
INTEGRATOR	Integrator with limit (Tracking and automatic operating modes)
LAG_FILTER	Time lag device: 1st order
LDLG	PD device with smoothing (phase advance/delay)
LEAD	Differentiator with smoothing
MFLOW	Controller for mass flow, e.g. for processing the differential pressure measurement of a throttle device
QDTIME	Deadtime device, delay function for quick parametering (Q = Quick)
SCALING	Scaling of all discrete variables
TOTALIZER	An integrator for integrating a flow and thereby calculating a flow volume. Very small values can be taken into account with this EFB, even if the total volume is large. It has a partial amount and a total amount counter.
VEL_LIM	Limiting the input or intermediate variable velocity

"Controller" group

The contents of this group a block for autotuning (AUTOTUNE). This block is standardized with the PI_B and PIDFF controller blocks. Self-tuning controller applications can be programmed with this.
This group contains the following EFBs:

Block	Meaning
AUTOTUNE	Autotuning
PI_B	Simple PI controller
PIDFF	Complete PID controller
STEP2	Two-step controller
STEP3	Three point controller

"Mathematics" group

Arithmetic functions are often used in connection with dead zones and weightings in the regulation zone.
This group covers directly applicable arithmetic functions on the basis of this principle.

- Multiplication / division with weighting: MULDIV_W
- Summation with weighting: SUM_W
- Comparison with dead zone and hysteresis: COMP_DB
- Square root with division and weighting K_SQRT

This group contains the following EFBs:

Block	Meaning
COMP_DB	Comparison
K_SQRT	Square root
MULDIV_W	Multiplication / division
SUM_W	Summer

"Output processing" group

It is often not possible to use the controller output directly to control the actuator.
If for example, as in the case of many processes, electric server motors are in use, a SERVO function block must be switched to the controller.
If two actuators are affecting the same variable, the SPLRG function block should be used. This function block functions both as a three step controller (when the actuators have an opposing effect) and in the "Split range" operating mode (when the actuators have an equal effect).
The PWM1 block enables pulse width modulation, for example of a setting variable of a pre-enabled continuous controller (PI, PID).
Although all the controller blocks can work in manual operating mode, it is often necessary to used the MS function block for this purpose.
This block enables extended control of manual operation mode

- The variable to be controlled is not the control output directly
- The output is not controlled via a servo loop
- The servo loop has a long sampling interval (1s and over)

This group contains the following EFBs:

Block	Meaning
MS	Manual control of an output
PWM1	Pulse width modulation
SERVO	Control for electric server motors
SPLRG	Controlling two actuators

Setpoint Management group

The classic 'Select Setpoint' function is integrated into the SP_SEL function rather than the control elements. This modular structure enables greater flexibility and improved user comfort without losing extended functions.
This includes the following:

- Tracking the process value if the servo loop is set to manual mode
- Bumpless switchover internal/external
- Bumpless extern/intern changeover (with setpoint tracking)

Two other function blocks make it possible to generate the setpoint to be switched to the controller: the RATIO function block, which is used to control a variable depending on a different variable (relationship control) and the RAMP block, which makes it possible to generate a setpoint in ramp form.
This group contains the following EFBs:

Block	Meaning
RAMP	Ramp generator
RATIO	Ratio controller
SP_SEL	Setpoint switch

Operating mode

Operating mode

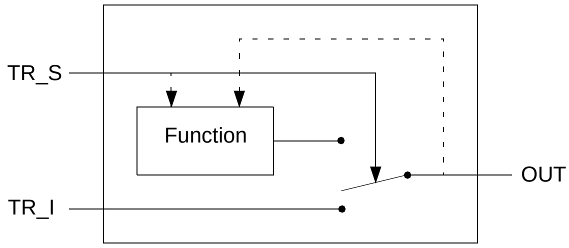
Several function blocks have integrated operating mode control available.
A choice can be made between the following operating mode:

- Tracking
- Manual/Automatic

The Order of priorities of the operating mode is explained further.

Tracking

This operating mode makes it possible to set a function block to the ‘Sub Controller’ operating mode. Two inputs make it possible to control this operating mode: a binary input TR_S (TRacking Switch), and a signal input TR_I (TRacking Input). If a function block is in tracking mode (TR_S = 1), its main output (e.g. OUT with a PIDFF controller) is assigned the input value TR_I and the internal variables of the different algorithms are updated. In this way a bumpless changeover is guaranteed when the function block is switched to manual or automatic mode.
The OUT output of the FFB is controlled with the TR_I input in tracking mode.
Tracking operating mode

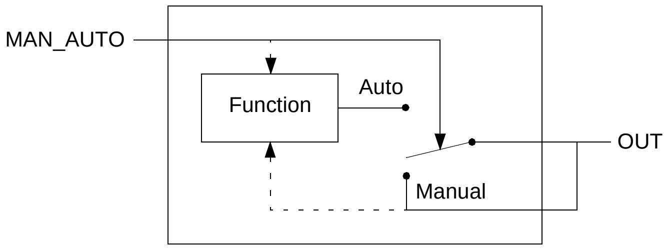


This operating mode can be used in various situations:
- Initializing during the start phase,
- Tracking operating mode with a redundant PLC, to guarantee a bumpless start for the Standby device,
- Controlling the operating mode using a program, for example to avoid direct control of the manipulated variable, when an automatic controller setting is in progress, etc.

A limit can be assigned to the function block’s output if it is in tracking operating mode: this should be decided separately for the individual function blocks.

Manual/ Automatic

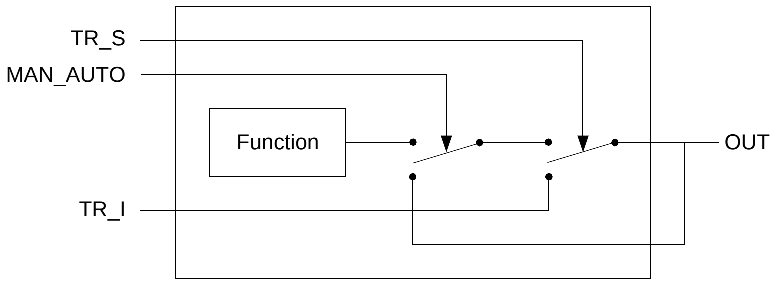
If a function block is in automatic mode, its algorithm calculates the value to be assigned to the output. Manual mode can be used to bar the adjustment of the main output (OUT) of a function block, to permit control via a user dialog, for example. The MAN_AUTO input permits control of this operating mode (0 : Manual, 1: Automatic). Manual/Automatic mode



The function block reads this output, however, and thus permits a bumpless changeover between the Manual <-> Automatic modes. A limit can be assigned to the function block’s output if it is in manual or automatic mode: this should be decided individually for each function block.

Order of priorities of the operating mode

If a function block has both operating mode available, the tracking operating mode has priority over the manual/automatic mode:



The connections between the function and the operating mode of the function block are not displayed to ensure a better overview. The same applies to the effectively assigned setpoint.

Scanning

Scanning

The control algorithms are based on scan values where the time interval between two consecutive cycles should be taken into account. The function blocks calculate the value of this interval automatically, which means they can be placed anywhere in the Concept section without any need to take the time management into account.
The following control functions can be done with a fixed time interval :
- Run time optimization of the PLC program by dividing the control operations into several cycles,
- improved control quality, where scanning the servoloop too frequently is prevented
- Minimizing the demands on the tuning device

For example, the SAMPLETM function block can be used, which should be attached to the input EN of the function block to be scanned.
If the scan interval of the servoloop exceeds 1 second, the function block *MS: Manual control of an output, p. 171* should be switched to the function blocks *PIDFF: Complete PID controller, p. 275* and *PI_B: Simple PI controller, p. 229* so that the servoloops can be controlled manually independently of the scan interval.

Error management

Principle

Most of the function blocks of the groups "Conditioning", "Controller", "Output Processing" and "Setpoint Management" have a STATUS output word available. The error recording and notification procedures used by these function blocks are described in this chapter.
Each bit of the STATUS parameter can be used for notifying an error, an alarm or some information. The meaning of the first 8 bits of the STATUS word is the same for all modules. The meaning of the subsequent bits (bits 8 to 15) is different for each function block.

Status word

The following table shows the meaning of the bits common to all the function blocks in the first byte of the STATUS word. Further information can be found in the description of each function block.

Bit	Meaning	Type
Bit 0 = 1	Error in a calculation with floating point values (e.g. calculation of the square root of a negative number)	Error
Bit 1 = 1	An unauthorized value being recorded on a floating point input can be caused by the following: ● the value is not a floating point value ● the value is infinite (e.g. the result of a calculation previously enabled to the function block)	Error
Bit 2 = 1	Division by zero with calculation in floating point values	Error
Bit 3 = 1	Capacity overflow with calculation in floating point values	Error
Bit 4 = 1	An input parameter is outside the zone. The value internally used by the function block is capped.	Warning or information (Note 1)
Bit 5 = 1 (Note 2)	The main output of the function block has reached the lower threshold	Information
Bit 6 = 1 (Note 2)	The main output of the function block has reached the upper threshold	Information
Bit 7 = 1	The lower and upper threshold of the input parameter zone are identical	Error

Note 1 (input parameter)

Note: If the value originates from a parameter zone with derived data types (typically the PARA parameter), a warning is given because of the capping and bit 4 is set to 1. If the value originates from a simple type of inputs, no warning is given, but bit 4 of the STATUS word is set to 1.

Note 2 (thresholds)

Note: If the upper and lower threshold parameters of an output have been invented (e.g.. out_min >= out_max), the function block switches the output to the lowest value (i.e. to out_max).

Convention

Specifying the convention

If a Boolean parameter is used to differentiate between 2 operating mode or 2 states of a function block, its name often has the following form: mode1_mode2 (Example: MANU_AUTO, SP_RSP). It is usually specified that the mode1 corresponding value is 0 and the mode2 corresponding value is 1. If for example the MANU_AUTO parameter of a function block is 0, the function block is in manual mode. It is in automatic mode when MANU_AUTO is equal to 1.

EFB Descriptions (A to PH)

II

Overview

Introduction

The EFB descriptions are arranged in alphabetical order.

Note: The number of inputs of some EFBs can be increased (up to a maximum of 32) by vertically resizing the FFB symbol. For information on which EFBs have this capability, please see the descriptions of the individual EFBs.

What's in this part?

This part contains the following chapters:

ALIM: Velocity limiter: 2nd order

3

Overview

At a glance

This chapter describes the ALIM block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block produces velocity limiter: 2nd order.
The function block individually contains the following properties:

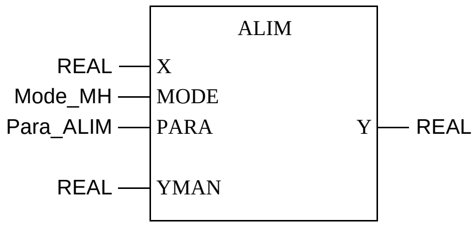
- Operating mode, Manual, Halt, Automatic
- Output limiting

EN and ENO can be projected as additional parameters.

Presentation

Symbol

Block display:



Parameter description ALIM

Block parameter description:

Parameter	Data type	Meaning
X	REAL	Input
MODE	Mode_MH	Operating mode
PARA	Para_ALIM	Parameter
YMAN	REAL	Manual value for output Y
Y	REAL	Output

Parameter description Mode_MH

Data structure description:

Element	Data type	Meaning
man	BOOL	"1" = Operating mode Hand
halt	BOOL	"1" = Halt mode

Parameter description Para_ALIM

Data structure description:

Element	Data type	Meaning
max_v	REAL	Maximum upper speed (maximum x') Unit: 1/[s]
max_a	REAL	Maximum speed increase (maximum x') Unit: $1/s^2$

Detailed description

Parametering

The parametering of the function block appears through determination of the maximum upper speed max_v as well as the maximum speed increase max_a. The maximum upper speed specifies to which value the output Y can change within one second. The maximum speed increase specifies the maximum value the output Y can change speed at.
The value of Y follows the value of X, but is limited by the maximum permitted speed and speed increase.

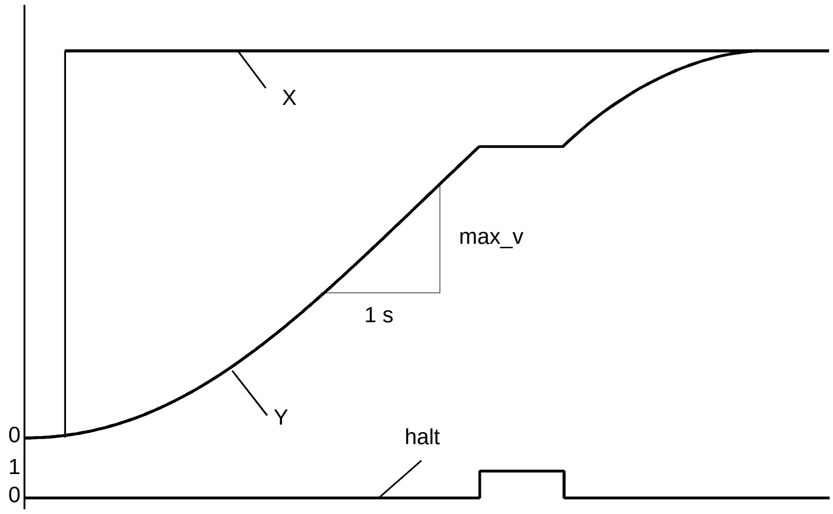
Operating mode

There are three operating mode selectable through the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	A new value for Y will be constantly calculated and issued.
Manual mode	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

Example

In the diagram the dynamic behavior of the function block is displayed as well as the reaction during HALY operating mode.



The jump at input X causes the function block to react with an accelerated increase of output Y. Output Y is accelerated with an acceleration increase determined by parameter max_a. Should the slew rate reach the max_v value, acceleration stops, but output Y continues to follow input X with the maximum slew rate max_v (see the straight section in the middle of the figure).
If the value of output Y is close enough to input signal value, the output is reversed to brake at a negative speed increase of –max_a, so that the output does not come to an abrupt stop, but slowly approximates the terminal point.

Runtime error

Error message

There is an Error message, if
- an invalid floating point number lies at input YMAN or X,
- max_a or max_v is ≤ 0.

AUTOTUNE: Automatic regulator setting

4

Overview

At a glance

This chapter describes the AUTOTUNE block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

This Function block enables the autotuning of the PID controller (*PIDFF: Complete PID controller, p. 275*, *PI_B: Simple PI controller, p. 229*).
Autotuning stabilizes the control when starting the system and, in so doing, saves time.
EN and ENO can be configured as additional parameters.

Algorithm

The algorithm is based upon heuristic controls, as with the Ziegler Nichols method. Initially, an analysis corresponding to approximately 2.5 times the reaction time of the open loop is performed. Through this, the process can be identified as a process of the first order with delay.
Building on this model, a control parameter set based on heuristic controls and historical data is created.
The parameter range is determined by the "perf." criteria. In this individual case, this factor gives the highest rank to the reaction time to disturbances or stability.
The algorithm is applied to the following process types :
- Processes with only one input / output
- Processes with natural stability or integral components
- Asymmetric processes within the limits authorized by the algorithm of the PID controller
- Processes controlled via pulse width modulation output (PWM).

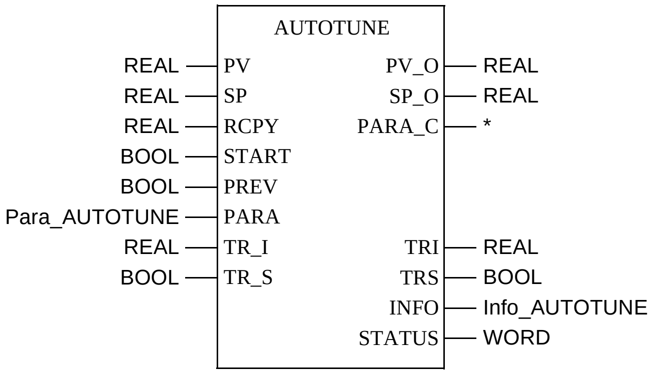
Important characteristics

The block has the following characteristics
- Pre-estimation of the control for the types PIDFF and/or PI_B
- Diagnostic function
- Parametering of the control dynamic
- Recovery of previous control settings

Representation

Symbol

Block representation



***** Parameters of the autotuned controller (Para_PIDFF, Para_PI_B,…etc.)

AUTOTUNE parameter description

Block parameter description

Parameter	Data type	Meaning
PV	REAL	Process value
SP	REAL	Setpoint
RCPY	REAL	Copy of the actual manipulated variable
START	BOOL	"0 → 1" : Starting the autotuning
PREV	BOOL	Reverting to the previous controller settings
PARA	Para_AUTOTUNE	Parameter
TR_I	REAL	Start input
TR_S	BOOL	Start command
PV_O	REAL	Copy of the process value PV
SP_O	REAL	Copy of the SP input
PARA_C	Parameters of the autotunable controller (Para_PIDFF or. Para_PI_B)	Control parameters
TRI	REAL	Copy of the TR_I input
TRS	BOOL	Copy of the TR_S input
INFO	Info_AUTOTUNE	Information
STATUS	WORD	Status word

Parameter description Para_AUTOTUNE

Data structure description

Element	Data type	Meaning
step_ampl	REAL	Value of the output actuating pulse (expressed in output scale values out_inf, out_sup)
tmax	TIME	Duration of the actuating pulse in autom. Tuning
perf	REAL	Performance index between 0 and 1
plant_type	WORD	Reserved word

Info_AUTOTUNE parameter description

Data structure description

Element	Data type	Meaning
diag	UDINT	Double word used for diagnosis
p1_prev	REAL	Previous value of parameter 1
p2_prev	REAL	Previous value of parameter 2
p3_prev	REAL	Previous value of parameter 3
p4_prev	REAL	Previous value of parameter 4
p5_prev	REAL	Previous value of parameter 5
p6_prev	REAL	Previous value of parameter 6

Principle of the autotuning

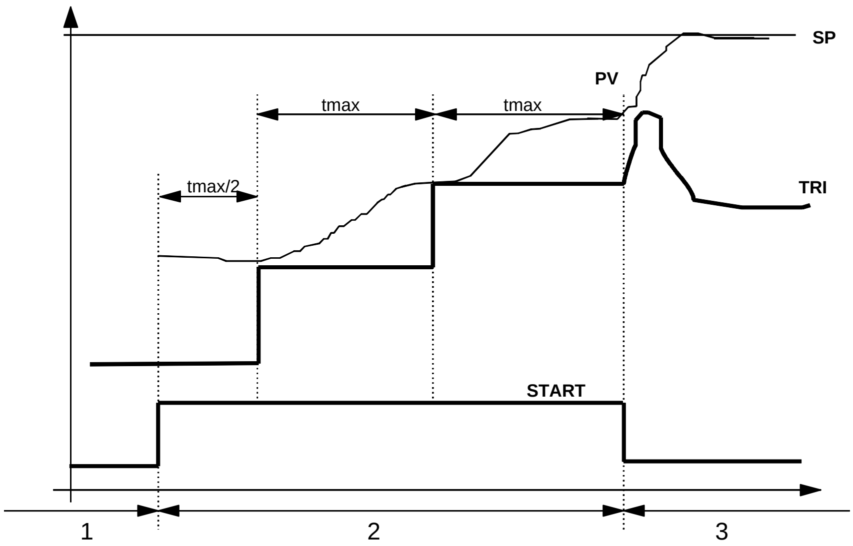
Two kinds of autotuning

Two kinds of autotuning are possible autotuning at a warm and cold system start The first phase of autotuning applies for both kinds of tuning: this involves a sound and stability test of the control process lasting 0.5 * tmax with constant outputs. Subsequent phases depend on the kind of tuning.

Autotuning at a cold start

Autotuning at a cold start is referred to when the deviation between the process and setpoint values exceeds 40% and the process value is less than 30%. In this case the TRI output of the function block is admitted with two actuator pulses of the same kind. Each actuator pulse has duration tmax. When autotuning ends, there is a smooth return to the previous operating mode for the servo loop:

Autotuning at a cold start



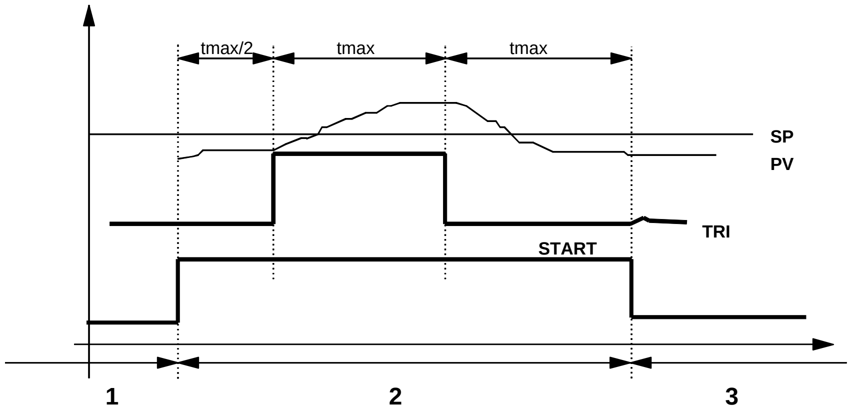
1 Automatic or manual mode

2 Autotune mode

3 Automatic or manual mode

Autotuning at a warm start

If the conditions for autotuning at a cold start are not fulfilled, tuning at a warm start takes place: the output is admitted with an actuator pulse, followed by an actuator pulse in the opposite direction. Each stage has duration tmax. When autotuning ends, there is a smooth return to the previous operating mode for the servo loop: Autotuning at a warm start



1 Automatic or manual mode

2 Autotune mode

3 Automatic or manual mode

Identification principle

Identification process

The identification process consists of 3 stages:
- a sound and stability analysis of the control process
- an initial analysis of the reaction to an actuator pulse, which is shown as the first identification model: a filter is created on the basis of this first estimate; this is used during the last phase
- a second analysis of the reaction to a second actuator pulse gives more precise information because of the data filter

Finally, a complete process model is created. If the results of the two previous phases are two far apart, the estimate is abandoned and autotuning fails.

Control principle

After both phases a parameter set is created for the controller being tuned. The resulting control parameters are based on the gain and on the ratio between reaction time and process delay.
The algorithm must be able to withstand the modification of the gain and the time constants in ratio 2 without losing stability. The asymmetrical processes are supported if they fulfil these conditions. If not, an error is displayed during diagnosis diag.

Parametering

Parametering actuating pulse

During autotuning, the output TRI is turned up two actuating pulses. An actuating impulse is identified by two parameters: its time duration (tmax) and its amplitude (step_ampl.).
The following value ranges are valid for these parameters: tmax greater than 4 seconds and step_ampl greater than 1 % of the output scale (out_inf, out_sup). The function also monitors even if the TRI output exceeds the threshold for the output scale.
The check occurs when autotune is started.
The following table contains parameter values for some of the typical control methods:

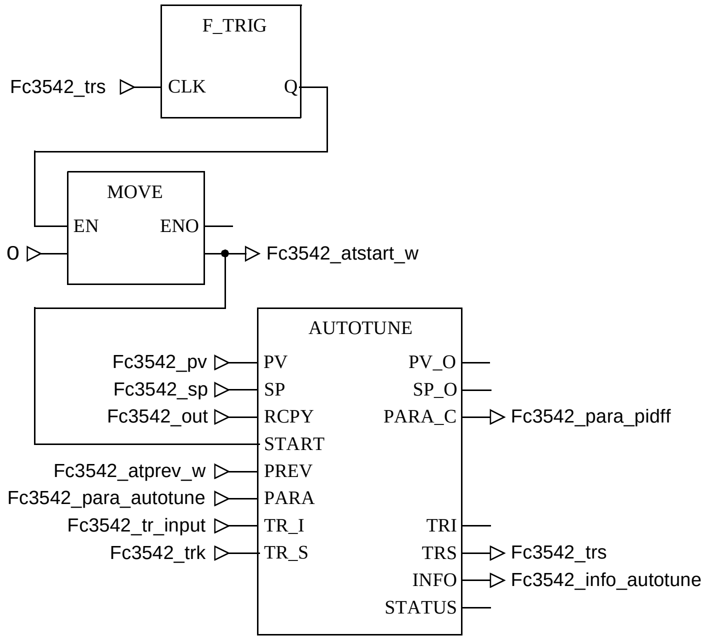
Diagram	tmax (s)	step_ampl (%)
Vol. flow or pressure from liquids	5-30	10-20
Gas pressure	60-300	10-20
Level	120-600	20
Steam temperature or pressure	600-3600	30-50
Module	600-3600	30-50

Performance index: perf

The controller can be modulated for each value in the performance index. The perf. performance index varies between 0 and 1, which enables the perf. parameter to stabilize close to 0 or to achieve a more dynamic control (and therefore optimize the reaction time of disturbance variables), if the perf. is set close to 1.

Starting the autotune: START

If this bit is set to 1, the function is activated. At the end of the setting process, this bit must be manually set to 0. If it has just been set automatically, setting the bit to 0 allows the function to be stopped. The PARA_C then retain the last active value. In the example below, the START bit is automatically reset by the program at the end of the setting process.
Example for starting the autotuning



Reverting to the previous setting: PREV

A modification of this bit value enables the exchange of current and previous parameters assuming that no controlling has occurred up to the given time (two consecutive modifications of this bit give the original configuration).
The following Info_AUTOTUNE structural parameters are valid for PIDFF type controllers:

Element of the data structure	Meaning
p1_prev	KP
p2_prev	TI
p3_prev	TD

The following Info_AUTOTUNE structural parameters are valid for the controllers of the PI_B type.

Element of the data structure	Meaning
p1_prev	KP
p2_prev	TI

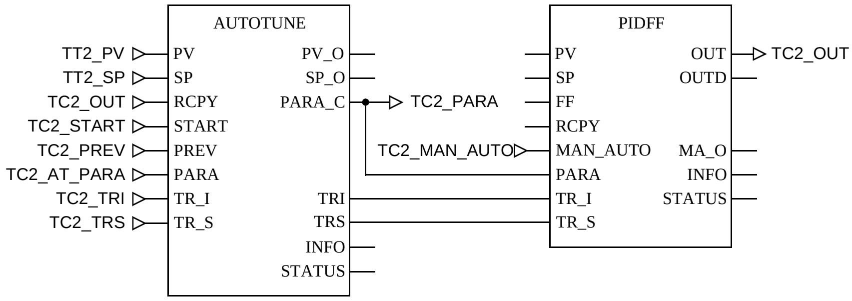
Diagnosis during autotuning: diag

The diagnosis data for the autotune is saved in a double word. The value of this word is retained until autotune is restarted. Additional details on this double word can be found in the Diagnosis section.

Controller coupling

Application example with a PIDFF controller type EFB

The following diagram is an application example of an AUTOTUNE EFB with a PIDFF controller type EFB :



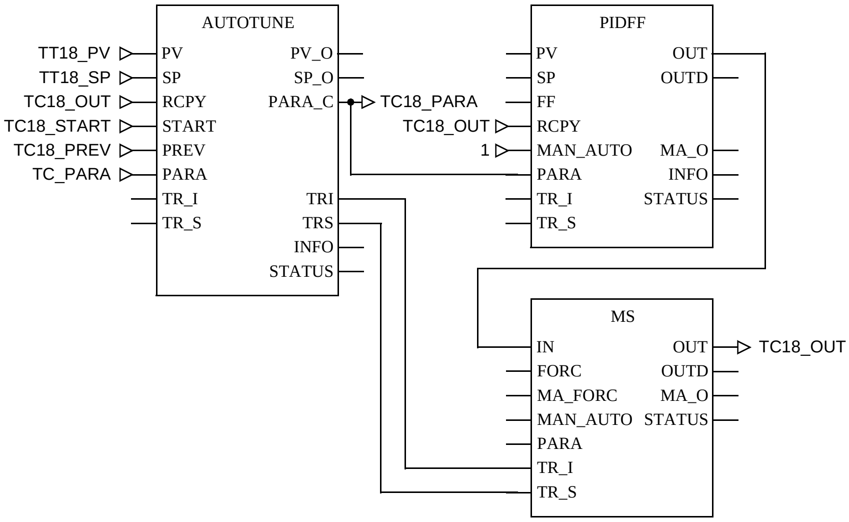
The AUTOTUNE EFB exchanges with the controller parameter: Access to the controller parameters is via the link between the output PARA_C of the AUTOTUNE function block and the input PARA of the controller. The PARA_C output is of the ANY type and enables the connection of the AUTOTUNE EFB to various controller types (PIDFF or PI_B).
The AUTOTUNE EFB and the controller also share the following interlinkable variables: PV, SP, TR_I and TR_S. These variables display AUTOTUNE inputs, which lead to the corresponding outputs, in order to switch to controller inputs
If the autotune is active, the TRS output transfers to 1 and the manipulated variable is attached at the TRI output. The purpose of these outputs is to connect to the inputs TR_I and TR_S of the function blocks following AUTOTUNE. In this way, these can be set to the tracking operation mode (PIDFF, PI_B, MS,…).

Example for connection: Servoloops with a simple PID controller

This section is concerned with the automatic setting of a single controller (most frequent case). The controller can be of PI_B or PIDFF type.
The AUTOTUNE EFB requires the scaling parameters of the controller (PARA_C structure parameters) pv_inf, pv_sup, out_inf, out_sup as well as the controller's structure type, which is specified via the mix_par bit. The EFB creates the parameters of the PID controller (KP, TI, TD) from this. The direction of action of the controller (rev_dir) is checked when testing the autotune and is compared to the sign for the gain of the model. When incompatibility occurs, an error is shown for the "diag." Parameters.

Example for connection: Servoloops with simple PID controller and MS function block

If the servoloop contains a MS-EFB, the structure can appear as follows:



When starting the autotune, the AUTOTUNE EFB sets the MS function block to tracking mode and hence controls the output of the servoloop directly. Using AUTOTUNE and PIDFF blocks' RCPY inputs enables a bumpless restart of the servoloop.

Operating modes

Operating modes

The various operating modes of the autotuning and their priorities in descending order of validity are shown in the following table:

Operating mode	TR_S	START
Tracking	1	1 or 0
Autotuning	0	1

On completion of the autotuning, the TRS output is set to 0, so as the servoloop is set back to its previous operating mode (manual or automatic). If the autotuning fails, the TRI variable will be set back to its value from before the autotuning was started and the servoloop will be set back to its previous operating mode.

Diagnosis

Overview of the diagnosis

There are a number of reasons that can lead to the autotuning not starting, being cancelled or failing. In such a case, depending on the cause of failure, it can be possible to supply a parameter set. Every bit of the diagnostic word diag. allows for a type of error to be created.
This word contains the current operating mode of the autotuning.
The following cases are explained:
- *Status of the autotuning, p. 39*
- *Causes of a faulty start, p. 39*
- *Causes of autotuning termination, p. 40*
- *Generating a test after stopping the autotuning, p. 42*

Diagnostic word

The meaning of the data structure Info_AUTOTUNE element diag can be found in this table.

Bit	Meaning
Bit 0 = 1	Autotuning is running
Bit 1 = 1	Autotuning aborted
Bit 2 = 1	Parameter error
Bit 3 = 1	Alteration of parameters, which have just been set automatically
Bit 4 = 1	Stop as a consequence of system error
Bit 5 = 1	Process value saturated
Bit 6 = 1	Alteration too small
Bit 7 = 1	Sampling interval invalid
Bit 8 = 1	Incomprehensible reaction
Bit 9 = 1	Non-stabilized measuring at the start
Bit 10 = 1	Length of actuating pulse (tmax) too short
Bit 1 1= 1	Too much noise/interference
Bit 12 = 1	Length of actuating pulse (tmax) too long
Bit 13 = 1	Process with significant exceeding of the thresholds
Bit 14 = 1	Process without minimum phase
Bit 15 = 1	Asymmetric process
Bit 16 = 1	Process with integral component

Status of the autotuning

Overview

The following bits of the diagnostic word (the diag element) show the status of the autotuning.

Bit	Meaning
0	1 = automatic regulator setting is running
1	1 = automatic regulator setting is stopped

Bit 0 of the element diag

This Bit indicates that the automatic regulator setting is running. On quitting the automatic regulator setting or terminating using the START-Bit, this is set to zero.

Bit 1 of the element diag

This Bit indicates that the user stopped the last control by means of the START-Bit or by setting the operating mode to Tracking.

Causes of a faulty start

Overview

The following bits of the diagnostic word (see element diag) indicate a faulty start:

Bit	Meaning
2	1 = Parameter error
7	1 = incorrect sampling interval

Bit 2 of the element diag

The following causes can lead to a faulty start :
- Length of actuating pulse too short (tmax < 4 s),
- Amplitude too weak (step_ampl < 1% of output range),
- Cannot perform this protocol: If the output + n x the amplitude of the actuating pulse (where n = 1 for adjustment during a warm start and n = 2 for adjustment during a cold start) is outside the output range (out_inf, out_sup), then the test protocol cannot be used. Step_ampl must be set to a value that is compatible with the current work point.

Bit 7 of the element diag

If the sampling interval is too large in relation to the length of the actuating pulse (> tmax / 25), then the response test is too imprecise and the automatic regulator setting will be blocked. This typically occurs during very rapid regular processes (where tmax is larger than the rise time of the process, a matter of a few seconds). In this case tmax can be increased, because the algorithm reacts only slightly to this parameter (in the ratio of 1 to 3), or alternatively, the sampling interval can be set to correspond.

Causes of autotuning termination

Overview

The following bits of the diagnostic word (see element diag) show the reason for terminating the autotuning:

Bit	Meaning
3	1 = Modification of parameters during tuning
4	1 = Terminated due to system error
5	1 = Process value saturated
6	1 = Ascent too small
8	1 = Illogical reaction

Bit 3 of the element diag

If the parameters tmax or step_ampl are modified during the tuning, the operation will be cancelled.

Bit 4 of the element diag

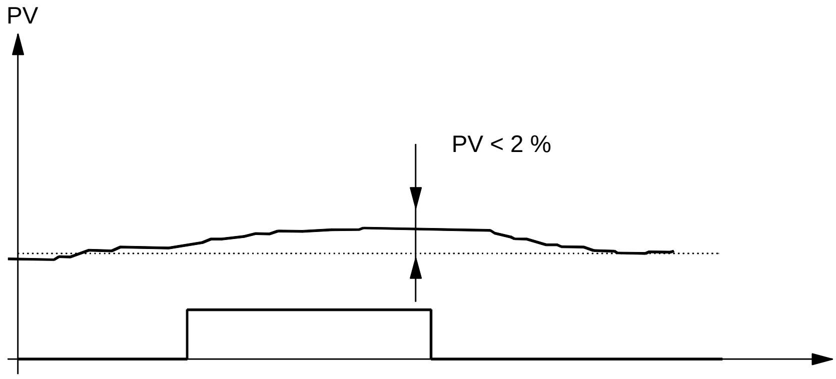
The autotuning will be cancelled if the PLC experiences a system error that prevents the completion of the chain. For example, the function will automatically stop should a voltage return occur.

Bit 5 of the element diag

If the measurement exceeds the range (pv_inf, pv_sup), then the autotuning will be cancelled, and the regulator set to the previous operating mode. Estimating the future measurements enables the autotuning to stop before the range is exceeded (if a first model has been identified).

Bit 6 of the element diag

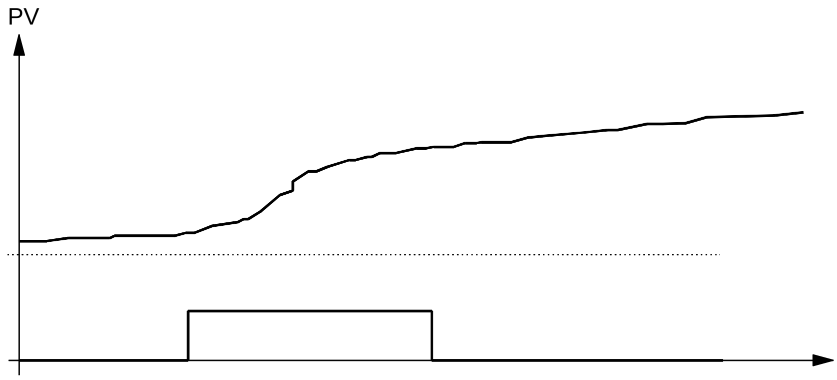
This picture shows the behavior when the ascent is too small:



The amplitude of the actuating pulse is too small too influence the process. In this case, the value of step_ampl can be increased.

Bit 8 of the element diag

This picture shows the behavior during an illogical reaction.



The reaction of the control process is incomprehensible (gain factors with various signs). This can be due to a larger disturbance, coupling with other servoloops or some other reason.

Generating a test after stopping the autotuning

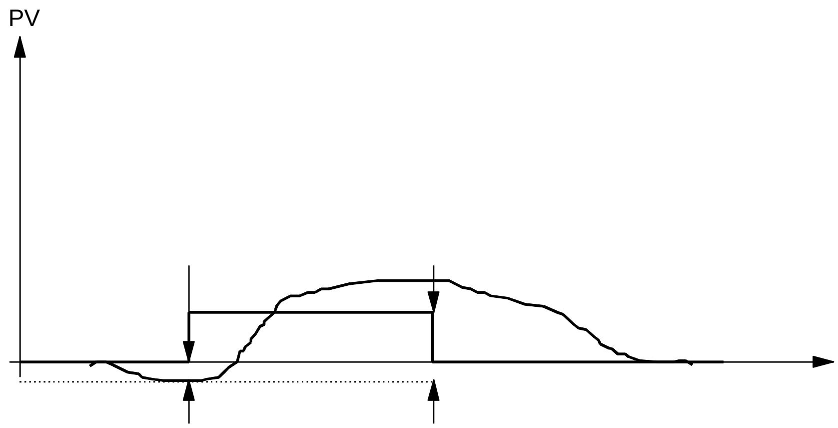
Overview

The following bits of the diagnostic word (see element diag) show the status of the autotuning:

Bit	Meaning
9	1 = Initial non-stabilized measurement
10	1 = Length of actuating pulse (tmax) too short
11	1 = Too much noise/interference
12	1 = Length of actuating pulse (tmax) too long
13	1 = Measured value has been significantly exceeded
14	1 = Process without minimum phase
15	1 = Asymmetrical Process
16	1 = Integrating Process

Bit 9 of the element diag

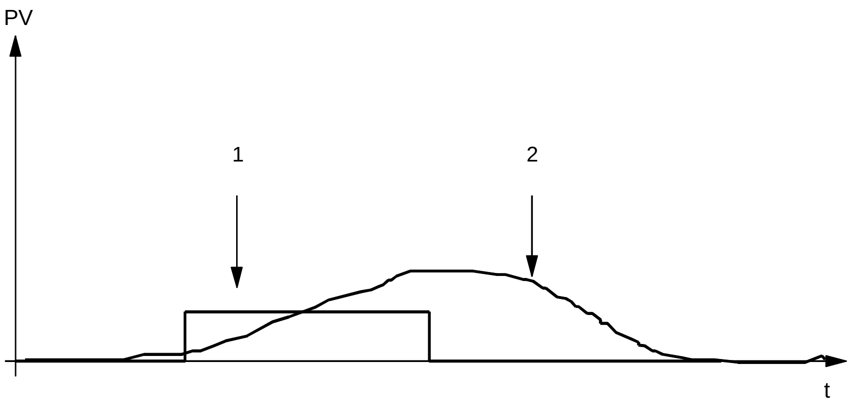
This image illustrates behavior when measurements are not initially stabilized:



The automatic regulator setting was implemented, although the measurement was not stable. If the measured change is large relative to the reaction of the actuating pulse, then the test results will be distorted.

Bit 10 of the element diag

This image illustrates behavior when the actuating pulse is too short:



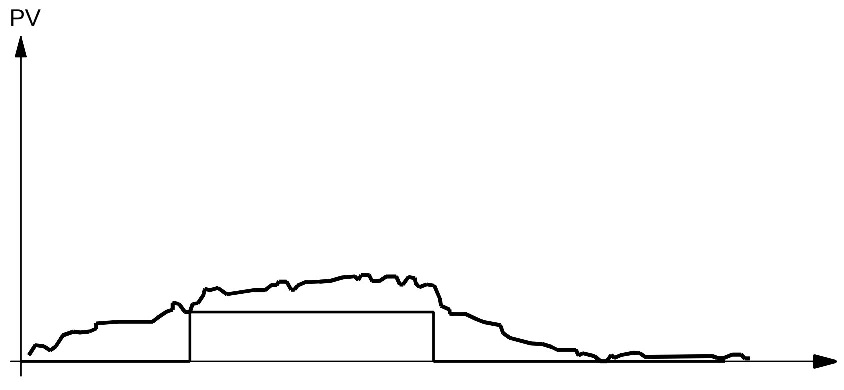
1 Actuating pulse test

2 Process reaction

The reaction will not be stabilized before returning to the original manipulated variable. The calculated parameters are therefore false.

Bit 11 of the element diag

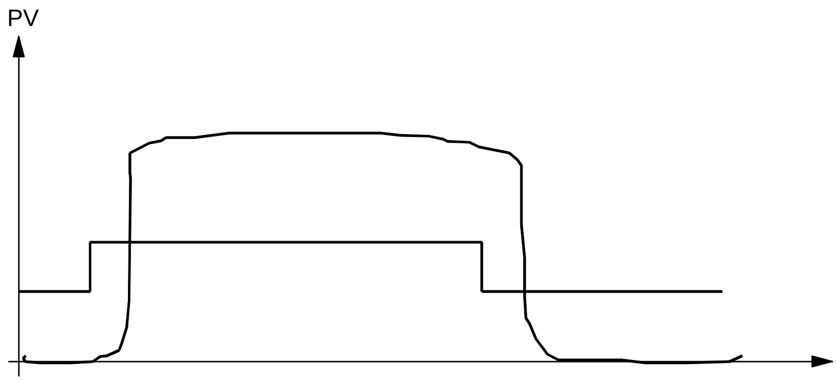
This image illustrates the behavior when noise/interference is too high:



The reaction of the process to the actuating pulse is insufficient relative to the level of noise/interference. The measurement should be filtered or step_ampl should be increased.

Bit 12 of the element diag

This image illustrates behavior when the actuating pulse is too long:



tmax specifies the frequency with which the measurement is taken, i.e. the value that is used to calculate the coefficients. tmax must be between 1 and 5 times the rise time of the repeated task.

Bit 13 of the element diag

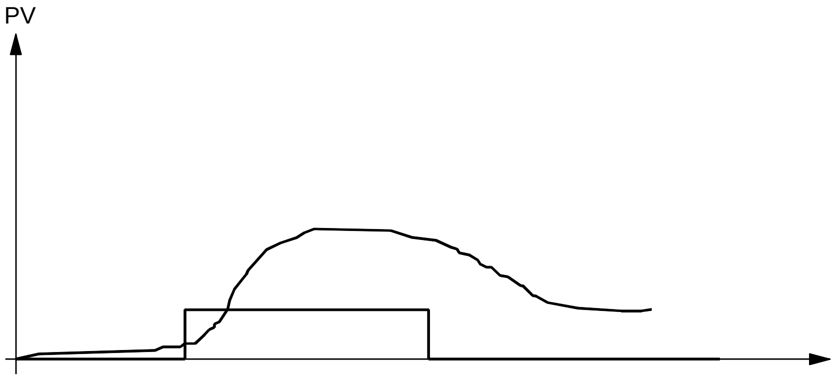
This bit is used when the reaction to an actuating pulse significantly exceeds (overshoots) the measured value (i.e. by more than 10%). The process does not conform to the models used by the algorithms.

Bit 14 of the element diag

This bit is used when the reaction to an actuating pulse leads to inversion of the reaction at the initial stage (i.e. undershoots by more than 10%). The process does not conform to the models used by the algorithms.

Bit 15 of the element diag

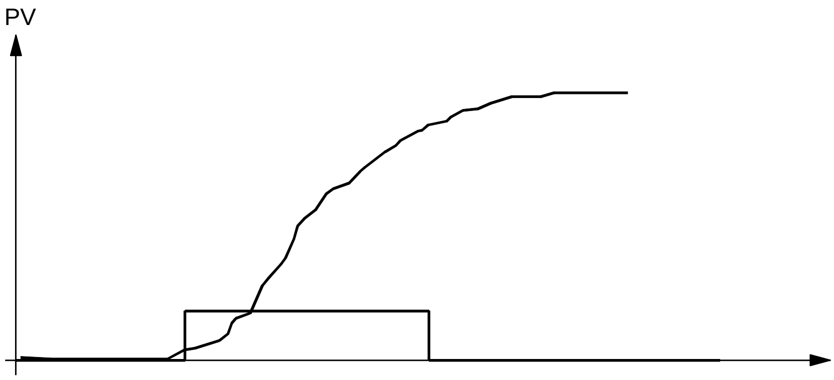
This image illustrates the behavior when the process is asymmetrical.



The reaction of the process is asymmetrical.
The last parameter set must be a compromise between the reactions at ascent and descent. Both cases concern average performance.
If the desired criterium is the length of the reaction on ascent, then the first parameter set must be taken into consideration. During the return phase (to the original manipulated variable) the automatic regulator setting is turned off. If the desired criteria is the length of descent, then a negative amplitude must be used.

Bit 16 of the element diag

This image illustrates the behavior during an integration process.



The process includes an integral component or tmax is too small and the process asymmetrical. The calculated coefficients must correlate to the process with the integral coefficient If this is not the case, the automatic regulator setting should be restarted, after tmax has been increased.

Runtime error

Status word

The status word bits have the following meaning:

Bit	Meaning
Bit 0 = 1	Error in a floating point value calculation
Bit 1 = 1	Invalid value recorded at one of the floating point inputs
Bit 2 = 1	Division by zero calculation when calculating in floating point values
Bit 3 = 1	Capacity overflow during calculation in floating point values
Bit 4 = 1	The parameter perf is outside the [0,1] range: in calculating the function block uses the value 0 or 1.
Bit 7 = 1	The thresholds (pv_inf and pv_sup) of the controller to be set are identical
Bit 8 = 1	The PARA_C output is not connected to the parameters of an autotunable controller
Bit 9 = 1	Autotuning failed
Bit 10 = 1	The last autotune was successful

Error message

This error is displayed when a non-floating point has been recorded at an input, when a problem occurs during a calculation with floating points or when the thresholds pv_inf and pv_sup of the controller are identical. In this case, all the outputs of the function block remain unchanged.

Warning

A warning is issued, if the parameter perf is outside the [0,1] range. In this case, the block can use either the value 0 or 1 for the purpose of calculations.

COMP_DB: Comparison

5

Overview

At a glance

This chapter describes the COMP_DB block.

What's in this chapter?

This chapter contains the following topics:

Brief description

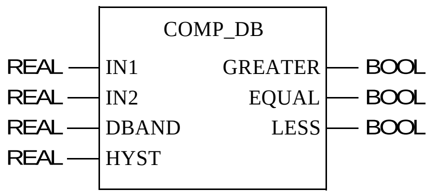
Function description

The COMP_DB function block enables two numerical values, IN1 and IN2 to be compared.
Depending on whether IN1 is greater, equal to or smaller than IN2, the appropriate output GREATER, EQUAL or LESS is set to 1 by the function block.
The function block takes any dead zone or hysteresis into account.
EN and ENO can be configured as additional parameters.

Representation

Symbol

Block representation



Parameter description

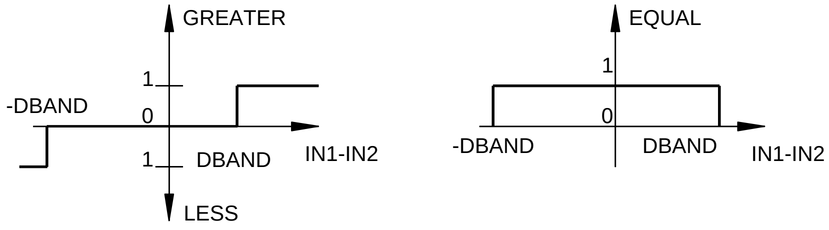
Block parameter description

Parameter	Data type	Meaning
IN1	REAL	Input No. 1
IN2	REAL	Input No. 2
DBAND	REAL	Dead zone
HYST	REAL	Hysteresis
GREATER	BOOL	Greater-than marker
EQUAL	BOOL	Equals marker
LESS	BOOL	Less-than marker

Detailed description

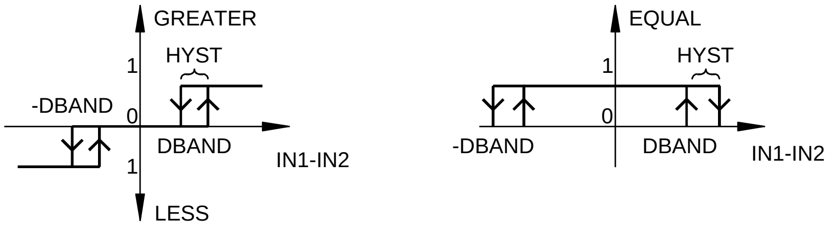
Dead zone

The D_BAND parameter enables a dead zone to be specified, within which deviation between IN1 and IN2 will be regarded as zero. If the deviation IN1 - IN2 remains within this zone, the EQUAL output is set to 1.
Dead zone specification



Hysteresis

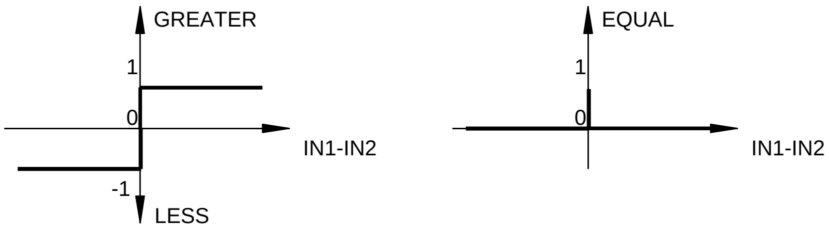
The HYST parameter enables a hysteresis effect to be generated, if the deviation between IN1 and IN2 decreases: starting from a situation where either the GREATER or LESS output has the value 1, the EQUAL output will only take the value 1 when the deviation IN1 – IN2 is less than DBAND – HYST.
Generating a hysteresis effect



DBAND = 0 and HYST = 0

In this case, the block behaves like a classic comparison function:
- If IN1 is always greater than IN2, then GREATER = 1
- When IN1 is equal to IN2, then EQUAL = 1
- If IN1 is less than IN2, then LESS = 1

Classic comparison function (DBAND = 0 and HYST = 0



Runtime error

Error message

This error appears if a non floating point value is recorded at an input or if there is a problem with a floating point calculation. In this case the outputs GREATER, EQUAL and LESS remain unchanged.

Warning

A warning message appears if:
- The DBAND parameter is negative: the function block then uses the value DBAND=0 for calculation.
- The HYST parameter is outside the [0, DBAND] range: the function block then uses the closest correct value, i.e. if HYST is less than 0 and DBAND, or when HYST is larger than DBAND.

COMP_PID: Complex PID controller

6

Overview

At a glance

This chapter describes the COMP_PID block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block represents a complex PID controller that in its design specifically includes cascade treatment. The control structure is displayed in the*Structure diagram, p. 56*.
EN and ENO can be configured as additional parameters.

Properties

The function block has the following properties:

- real PID controller with independent gain, ti, td setting
- Manual, halt, automatic, cascade, reset, manual value operating modes tracking
- Velocity limit for manual operation
- Adjustable manual manipulated value tracking
- Velocity limit for reference variable
- bumpless changeover between manual and automatic
- Manipulated variable limiting
- bumpless, individually connectable P, I and D components
- bumpless gain modification
- Choice of antiwindup reset and antiwindup halt
- Displacement of antiwindup limits compared to control limits
- Antiwindup measure with an active I component only
- definable delay of the D-component
- D component connectable to controlled variable PV or system deviation EER
- Dead zone with gain reduction
- external operating point (in P, PD and D operation)
- Choice of bump/bumpless manual/automatic switchover

Transfer function

The transfer function is:

$$G(s) = gain \times \left(\underbrace{1}_{YP} + \underbrace{\frac{1}{ti \times s}}_{YI} + \underbrace{\frac{td \times s}{1 + td\_lag \times s}}_{YD} \right)$$

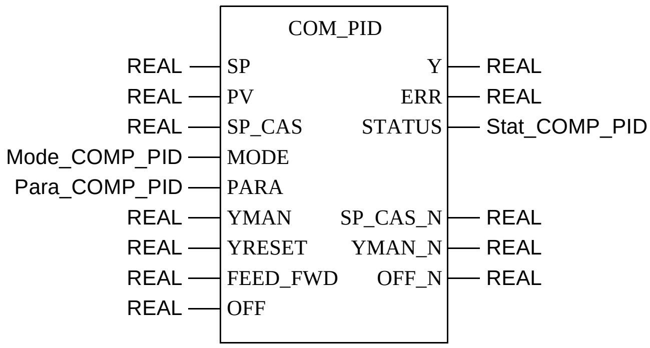
Explanation of the variables:

Variable	Meaning
YD	D component (only if en_d = 1)
YI	I component (only if en_i = 1)
YP	P component (only if en_p = 1)

Representation

Symbol

Block representation:



Parameter description COMP_PID

Block parameter description

Parameter	Data type	Meaning
SP	REAL	Reference variable
PV	REAL	Controlled variable
SP_CAS	REAL	Cascade reference variable
MODE	Mode_COMP_PID	Operating mode
PARA	Para_COMP_PID	Parameter
YMAN	REAL	Manually manipulated value
YRESET	REAL	Manipulated variable reset value
FEED_FWD	REAL	Disturbance input
OFF	REAL	Offset for P/PD operation
Y	REAL	Manipulated variable
ERR	REAL	System deviation
STATUS	Stat_COMP_PID	Output status
SP_CAS_N	REAL	Cascade reference variable
YMAN_N	REAL	Manually manipulated value
OFF_N	REAL	Offset for P/PD operation

Parameter description Mode_COMP_PID

Data structure description

Element	Data type	Meaning
r	BOOL	"1": Reset mode
man	BOOL	"1": Manual mode
halt	BOOL	"1": Halt mode
cascade	BOOL	"1": Cascade mode
en_p	BOOL	"1": P component in
en_i	BOOL	"1": I component in
en_d	BOOL	"1": D component
d_on_pv	BOOL	"1": D component on controlled variable "0": D component on system deviation
halt_aw	BOOL	"1": Antiwindup Halt "0": Antiwindup reset
bump	BOOL	"0": Bumpless operating mode switchover
ymanc	BOOL	"1": YMAN tracking

Parameter description Para_COMP_PID

Data structure description

Element	Data type	Meaning
gain	REAL	Proportional action coefficient (gain)
ti	TIME	Reset time
td	TIME	Rate time
td_lag	TIME	D component delay time
db	REAL	Dead zone
gain_red	REAL	Gain reduction in dead zone (db)
rate_sp	REAL	Setpoint velocity (SP) [1/s]
rate_man	REAL	Manually manipulated velocity value (YMAN) [1/s]
ymax	REAL	Upper threshold for Y
ymin	REAL	Lower threshold for Y
delt_aw	REAL	Limit expansion for antiwindup

Parameter description Stat_COMP_PID

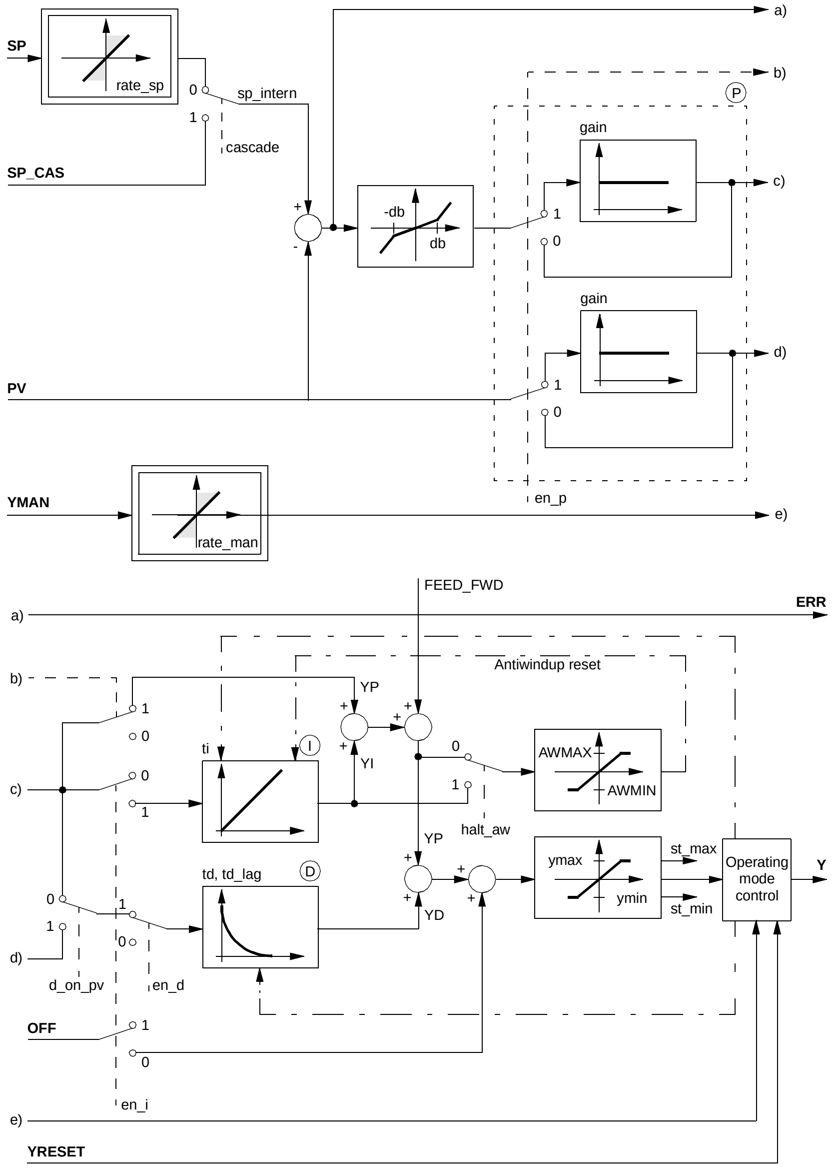
Data structure description

Element	Data type	Meaning
st_r	BOOL	"1": COMP_PID is in reset mode
st_man	BOOL	"1": COMP_PID is in manual mode
st_halt	BOOL	"1": COMP_PID is in halt mode
st_auto	BOOL	"1": COMP_PID is in automatic mode
st_cascade	BOOL	"1": COMP_PID is in cascade mode
st_max	BOOL	"1": Y ≥ Para_COMP_PID.ymax
st_min	BOOL	"1": Y ≤ Para_COMP_PID.ymin

Complex PID controller structure diagram

Structure diagram

The following is the structure diagram of the COMP_PID controller:



Parametering of the COMP_PID controller

Parametering

The COMP_PID control structure is displayed in the*Structure diagram, p. 56*.
The parametering of the function block is initially performed by the pure PID parameters, i.e. the proportional action coefficient gain, the reset time ti and the rate time td.
The D component is delayed by the time td_lag. The td/td_lag ratio is termed the differential gain, and is generally selected between 3 and 10. The D component can either be based upon the system deviation ERR (d_on_pv = "0") or the controlled variable PV (d_on_pv = "1"). Should the D component be determined by the controlled variable PV, then the D component will not be able to cause jumps when reference variable fluctuations (changes in input SP) take place. Generally, the D component only affects disturbances and process variances.

Note: The EFB has 3 I/O parameters (SP_CAS, OFF, YMAN) that are updated by the cascade mode function itself. To use the block in cascade mode, you have to establish the connection between these inputs and the appropriate outputs (SP_CAS_N, OFF_N, YMAN_N) through variables.

Control direction reversal

A reversed behavior of the controller can be achieved by reversing the sign of gain. Given a positive disturbance value, a positive/negative gain brings about a rise/fall of the manipulated variable. A negative value at gain causes the manipulated variable to drop when there is a positive deviation.

Forming the system deviation

In cascade mode, the ERR system deviation is formed by SP_CAS and PV:
- sp_intern = SP_CAS
- ERR = sp_intern - PV

The system deviation in automatic mode is formed by sp_intern and PV, whereby sp_intern is set to the value of parameter SP via a velocity limiter. The internal reference variable sp_intern is driven in ramp-type fashion toward the SP parameter value using the velocity specified in parameter rate_sp (unit 1/s).
The amount will be evaluated by parameter rate_sp. The function of the velocity limiter for SP is disabled if rate_sp = 0. SP is transferred directly to sp_intern.
System deviation is determined by the condition of parameter cascade when in reset, manual and halt modes.
If cascade = 1, sp_intern is set to the PV parameter value and ERR goes to 0.
If cascade = 0 and the setting is bumpless operation (bump = 0), sp_intern is set to the SP parameter value. Otherwise (bump = 1), sp_intern is also set to the PV parameter value.

Gain reduction for small system deviation values

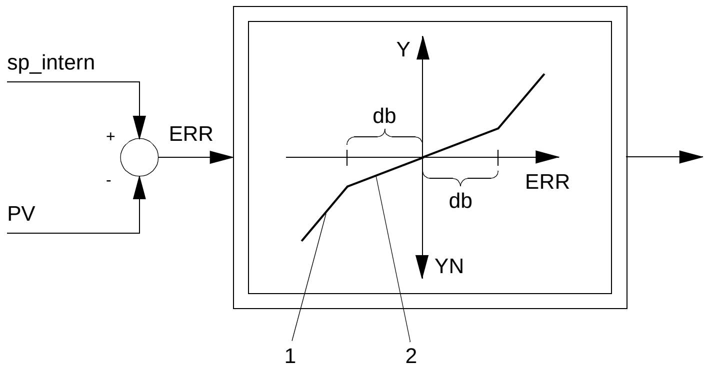
Parameter db determines the size of a dead zone in which the proportional action coefficient gain is not effective, but rather a proportional action coefficient reduced by the parameter gain_red. The parameter db has an effect on the system deviation ERR = SP - PV in the form shown in the illustration *Representation of the dead zone, p. 58*. Unnecessary actuator loads caused by small controlled variable disturbances or measurement noise can be reduced by the dead zone.
Enter the db parameter as positive.
Enter values between 0 and 1 for gain_red.

Tracking of manual value YMAN

When manual tracking mode is enabled (ymanc = 1), the input YMAN is tracked to the manipulated variable value Y when in automatic and cascade modes, this means: YMAN = Y. If manual tracking mode is disabled (ymanc = 0), the YMAN value remains unchanged.

Representation of the dead zone

Dead zone:



1 Gradient 1

2 Gradient gain_red

Manipulated variable limiting

The limits ymax and ymin retain the manipulated variable within the prescribed range. Hence, ymin ≤ Y ≤ ymax. .
The elements qmax and qmin signal that the manipulated variable has reached a limit, and thus been capped:

- st_max = 1 if Y ≥ ymax
- st_min = 1 if Y ≤ ymin

For limiting the manipulated variable, the upper limit ymax should be greater than the lower limit ymin.

Antiwindup for COMP_PID

Definition

The antiwindup measure ensures that the I component does not grow too much causing the controller to lock if it has been limited at a control limit too long.
Antiwindup measures are only performed for an active I component of the controller.
Limits for the antiwindup measure are by default the manipulated variables of the controller (delt_aw = 0). The parameter delt_aw can be used to either increase (delt_aw > 0) or decrease (delt_aw < 0) the limits with regard to the control limits (ymax, ymin).
Therefore, the limits used for the antiwindup measure are:

- AWMAX = ymax + delt_aw
- AWMIN = ymin - delt_aw.

Through displacement of the antiwindup limits in relation to the control limits (in particular with very noisy signals), the manipulated variable Y can be stopped from repeatedly 'jumping away' from the control limit (D component effect to disturbances) and subsequently returning to the limiting position (I component effect with system deviation ERR ≠ 0). If the control limits are to be simultaneously effective for the antiwindup measure, select the parameter delt_aw = 0.
By utilizing negative delt_aw values, antiwindup limits can be kept smaller than control limits (useful for antiwindup halt).

Antiwindup reset (halt_aw = 0)

Antiwindup measures disregard D component values to avoid being falsely triggered by D component peaks. The antiwindup-reset measure corrects the I component such that: AWMIN ≤ YP + FEED_FWD + YI ≤ AWMAX.

Antiwindup halt (halt_aw = 1)

The antiwindup measure only considers the I component. When antiwindup halt and I component are enabled, the antiwindup halt measure corrects the I component such that: AWMIN ≤ YP + FEED_FWD + YI ≤ AWMAX.
The parameters rate_sp and rate_man represent velocity limiters for the manual values SP and YMAN (see also function block VLIM). A 0 value disables the functionality of the corresponding velocity limiter (rate_sp = 0 or rate_man = 0, respectively). The SP and YMAN values are then utilized without delay.

Controller type selection for COMP_PID

Controller types

There are four different control types, which are selected via the parameters en_p, en_i and en_d.

Controller type	en_p	en_i	en_d
P controller	1	0	0
PI controller	1	1	0
PD controller	1	0	1
PID controller	1	1	1
I controller	0	1	0

The I-component can also be disabled with ti = 0.
The D contribution can also be disabled with td = 0.

OFF parameter influence

If the I contribution is enabled (en_i = 1), the manipulated variable Y is determined from the summation of the contributions YP, YI, YD, and FEED_FWD. Offset is not included in the calculation when the I contribution is enabled.
However, if the I component is disabled (EN_I = 0), the manipulated variable Y is formed from the summation of the components YP, YD, FEED_FWD, and the offset OFF.

Note: The OFF parameter is only designed for P, D, or PD controllers.

Bumpless operating mode switchover

Method of switching over

Bumpless on/off switching of the various components (P, I, D) is implemented.

Bumpless switching with enabled I component

If the P component is connected/disconnected, the internal I component will be corrected by the P component. This way, the connection/disconnection of the P contribution is bumpless even if the system deviation is not 0.
If the D component is disconnected, the internal I component takes over the remaining D component. If the D component is connected, it is set to 0.

Bumpless switching for disconnected D component

Bumpless switching for a disconnected D component is only implemented if parameter bump = 0. In this case, the OFF parameter is used to achieve the bumpless switchover.
If the P component is connected/disconnected, the value in the OFF parameter is corrected by the P component. This way, the connection/disconnection of the P component is bumpless even if the system deviation is not 0.
If the D component is disconnected, the remaining D component is added to the OFF parameter value. If the D component is connected, it is set to 0 (OFF remains unchanged).

Bumpless I component switching

Bumpless I component disconnection is only performed if parameter bump = 0. In this case, the OFF parameter as well as the internal I component (YI) are used to make the bumpless switchover possible.

Bumpless switchover from a PI(D) to a P(D) controller

The principle consideration for bumpless switching from a PI(D) to P(D) controller is based on the assumption that the PI(D) controller has reached a static condition. In this case, the process is in an idle state. The I component has a specific value in this case. To allow a bumpless switch to P(D) operation now, the I contribution of the PI(D) controller would have to serve as the PD controller operating point (offset), thus allowing the switch to take place without equalization processes (new transient condition) taking place. Based on the above consideration, bumpless I component disconnection is implemented in such a way that the OFF parameter retrieves its value.
Value of the manipulated variable Y depending on en_i:

If…	Then…
en_i = 1	Y = YP + YI + YD + FEED_FWD
en_i = 0	Y = YP + OFF + YD + FEED_FWD

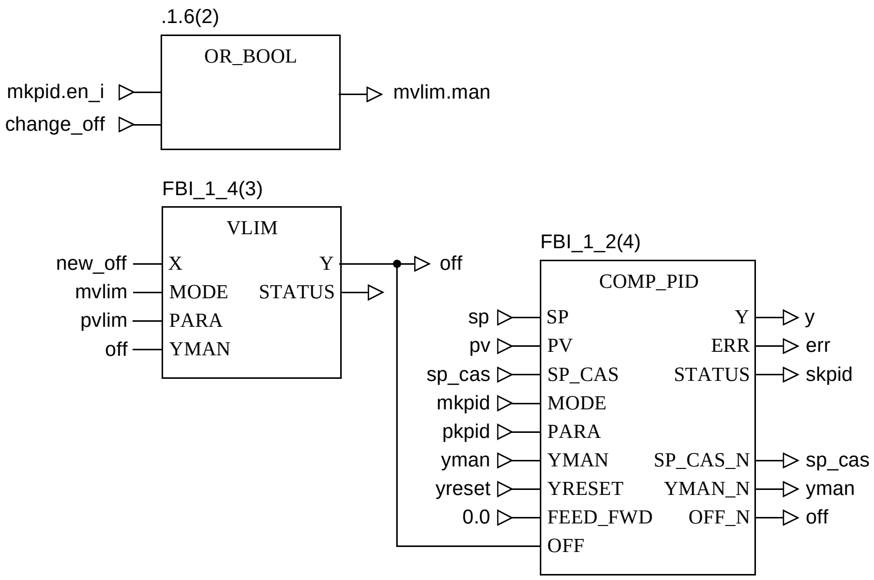
Starting up the I component

I component enabling is based on an analog consideration. The internal I component is set to the OFF parameter value. This allows the I component to be connected without giving rise to equalization processes.

Note: If the OFF parameter is calculated by a previous function block (EFB or DFB output, e.g. MOVE), the corrections for bumpless switching become ineffective (at the latest, when this function block is edited).

Example of a bumpless switchover of the D component

In order to achieve the bumpless P(D) controller switchover as well as OFF parameter modification by the user program, the following example can serve as a starting point.



In this example, the OFF parameter is set to the new_off variable value via a velocity limiter VLIM in ramp form using the velocity provided in pvlim.rate.

Note on the example

In this example, it is important to note the use of the OFF variable at the YMAN input of the VLIM as well as at the Y output of the VLIM, and the link of the output from VLIM to the OFF input of COMP_PID. The link between the Y output from VLIM and the OFF input from COMP_PID causes the VLIM function block to be processed prior to the COMP_PID function block (this is a prerequisite for proper operation). As long as the manual mode (mvlim.man = 1) is enabled in the VLIM, the manual value of the VLIM function block is transferred to the COMP_PID OFF parameter. The COMP_PID function block is now able to modify the content of the variable for bumpless handling. In the next cycle, this modified value is now available at the YMAN input of the VLIM function block. At an appropriate time, the manual mode in the VLIM function block can be disabled, and the function block drives up the value of the OFF variable from its current value to that of new_off. In the example above, manual mode enabling is controlled in the function block OR_BOOL. As long as COMP_PID has enabled the I component (mkpid.en_i = 1), the VLIM function block remains in manual mode.

Note: If mkpid.en_i = 1, the OFF parameter from COMP_ID will not be included in the calculation of the COMP_PID output.

In the above example, the OR_BOOL function block requires a second condition in order to change off to new_off: The variable change_off must be 1.

Bumpless alteration of gain

Modification of the proportional action coefficient gain is bumpless. As in the connection/disconnection of operating modes, this requires an internal correction to be carried out.
If the I component is enabled (en_i = 1 and ti > 0), the internal I component will be corrected by the expected P component jump which is caused by the gain modification.
If the I component is disconnected, the value in the OFF parameter will be corrected by the expected P component jump, provided the parameter bump = 0. If bump = 1, OFF is not modified and a P(D) controller gain variation leads to equalization processes.

Selecting the operating mode of the COMP_PID

Operating modes

There are five operating modes selectable through reset, man, halt, and cascade.

Operating mode	r	man	halt	cascade
Reset	1	1 or 0	1 or 0	1 or 0
Manual	0	1	1 or 0	1 or 0
Halt	0	0	1	1 or 0
Cascade	0	0	0	1
Automatic	0	0	0	0

Automatic and cascade modes

In automatic mode, the manipulated variable Y is determined through the discrete PID closed-loop control algorithm subject to controlled variable X and reference variable SP.
In cascade mode, the manipulated variable Y is determined through the discrete PID closed-loop control algorithm subject to controlled variable X and reference variable SP_CAS.
The distinction between these two operating modes, automatic and cascade, is only external in their different use of the reference variable SP. SP_CAS refers to cascade, SP to all other operating modes (with velocity limit). The SP_CAS variable is an input in cascade mode only, in all other modes it is an output. In SP_CAS, the X variable is returned to the master controller when in the modes reset, manual, halt or automatic as well as during startup, permitting bumpless switching from, for instance, fixed setpoint control to cascade control.
In both operating modes, the manipulated variable Y is limited by ymax and ymin. The control limits for the antiwindup measure can be extended using the parameter delt_aw.

Manual mode

In manual mode, the manual manipulated value YMAN is transferred to the manipulated variable Y with a velocity limiter. The manipulated variable Y is set to the YMAN parameter value in ramp form using the velocity (unit 1/s) rate set in the parameter rate_man.
The amount is evaluated by the parameter rate_man. +If rate_man = 0, the velocity limiter function for YMAN is disconnected. YMAN is transferred directly to the manipulated variable. The manipulated variable is limited by ymax and ymin. Internal variables will be manipulated in such a manner that the controller changeover from manual to automatic (with I component enabled) can be bumpless. The antiwindup measure is designed just like in automatic mode.
In this operating mode the D component is automatically set to 0.

Reset mode

In Reset mode, the reset value YRESET is transferred directly to the manipulated variable Y. The manipulated variable is limited by ymax and ymin. Internal variables will be manipulated in such a manner that the controller changeover from manual to automatic (with I component enabled) can be bumpless. The antiwindup measure is performed just like in automatic mode.

Halt mode

In halt mode, the control output remains as is, i.e. the function block does not change the manipulated variable Y. Internal variables will be manipulated in such a manner that the controller can be driven smoothly from it's current position. Manipulated variable limits and antiwindup measures are as those in automatic mode. Halt mode is also useful in allowing an external operator device to adjust control output Y, whereby the controller's internal components are given the chance to continuously react to the external influence.
In this operating mode the D component is automatically set to 0.

Non-bumpless operation (bump = 0)

The definition of non-bumpless operation is when the controller exhibits a jump during operating mode switchover (e.g. manual to automatic) due to the P component in the manipulated variable Y. Depending on the controller's area of utilization, it might be useful for the controller to make a jump-type correction of the manipulated variable when switching over, for instance from manual to automatic, provided the system deviation is not equal to 0.
The jump height corresponds to the P component of the controller and is:

YP = ERR x gain

Bumpless operation (bump = 1)

The definition of bumpless operation is, the controller does not produce a discontinuity in the manipulated variable Y during an operating mode switchover. That is, it should continue at exactly the same location where it was positioned last. In this operating mode, the internal I component is corrected by the P contribution. If no I component is enabled, bumpless operation is achieved by tracing the operating point OFF such that the controller can continue during operating mode change without a bump in spite of system deviation being not equal to 0.

Detailed formulas

Explanation of formula variables

Meaning of the variables in the following formulas:

Variable	Meaning
dt	Time differential between the current cycle and the previous cycle
ERR	The current internally formed System deviation
$ERR_{(new)}$	System deviation value from the current sampling step
$ERR_{(old)}$	System deviation value from the previous sampling step
FEED_FWD	Disturbance (only in P, D or PD controllers)
OFF	Offset
$PV_{(new)}$	Value of controlled variable from the current sampling step
$PV_{(old)}$	Value of controlled variable from the previous sampling step
Y	current output (halt mode) or YMAN (manual mode)
YD	D component (only if en_d = 1)
YI	I component (only if en_i = 1)
YP	P component (only if en_p = 1)

Manipulated variable

The manipulated variable consists of various terms which are dependent on the operating mode:

$$Y = YP + YI + YD + OFF + FEED\_FWD$$

After summation of the components manipulated variable limiting takes place, so that:

$$ymin \leq Y \leq ymax$$

Overview of the calculation of the control components

The following is an overview on the different calculations of the control components in relation to the elements en_-, en_I and en_d:
- P component YP for manual, halt, automatic and cascade modes
- I component YI for automatic mode
- I component YI for manual and halt modes
- D component YD for automatic and cascade mode
- D component YD for manual and halt modes

P component YP for all operating mode

YP for manual, halt, automatic and cascade modes is determined as follows:
For en_p = 1 the following applies:

$$YP = gain \times ERR$$

For en_p = 0 the following applies:

$$YP = 0$$

I component YI for automatic mode

YI for automatic mode is determined as follows:
For en_i = 1 the following applies:

$$YI_{(new)} = YI_{(old)} + gain \times \frac{dt}{ti} \times \frac{ERR_{(new)} + ERR_{(old)}}{2}$$

For en_i = 0 the following applies:

$$YI = 0$$

The I component is formed according to the trapezoid rule.

I component YI for manual and halt modes

YI for manual, halt and automatic modes is determined as follows:
For en_i = 1 the following applies:

$$YI = Y - YP - FEED\_FWD$$

For en_i = 0 the following applies:

$$YI = 0$$

D component YD for automatic and cascade mode

YD for automatic mode and cascade is determined as follows:
For en_d = 1 and d_on_pv = 0 the following applies:

$$YD_{(new)} = \frac{YD_{(old)} \times td\_lag + td \times gain \times (ERR_{(new)} - ERR_{(old)})}{dt + dt\_lag}$$

For en_d = 1 and d_on_pv = 1 the following applies:

$$YD_{(new)} = \frac{YD_{(old)} \times td\_lag + td \times gain \times (PV_{(old)} - PV_{(new)})}{dt + dt\_lag}$$

For en_d = 0 the following applies:

$$YD = 0$$

D component YD for manual and halt modes

YD for manual, halt and automatic modes is determined as follows:

$$YD = 0$$

Runtime error

Error message

An Error message appears, if
- an unauthorized floating point number is placed at the input PV
- gain_red > 1 or gain_red < 0 is
- db < 0 is
- or ymax < is ymin

DEADTIME: Deadtime device

7

Overview

At a glance

This chapter describes the DEADTIME block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

With this function block an input signal is delayed by a time, the so-called deadtime. The function block delays the signal X by the deadtime T_DELAY before it appears again at Y.
The function block utilizes a 128 element delay buffer to hold a sequence of X values, i.e. during the T_DELAY time 128 discrete X values are detained. The buffer is used in such a way that it corresponds with the operating mode.
The value of Output Y remains unchanged after cold and warm system starts. The internal values are set to the value of X.
After a change of deadtime T_DELAY or a cold or warm system start, the output READY goes to "0". This means: that the buffer is empty and not ready.
The function block has the following operating mode:

- Manual
- Halt
- Automatic.

EN and ENO can be projected as additional parameters.

> **Note:** The delay time continues to run even if the block is disabled via the EN parameter, because the block calculates its time differences according to the system clock.

Formula

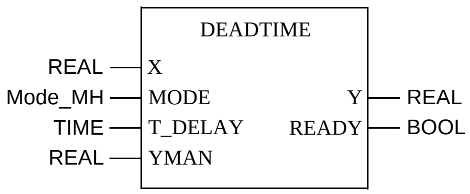
The transfer function is:

$$G(s) = e^{-s \times T\_DELAY}$$

Representation

Symbol

Representation of the block



DEADTIME parameter description

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input value
MODE	Mode_MH	Operating mode
T_DELAY	TIME	Deadtime
YMAN	REAL	Manual manipulated value
Y	REAL	Output
READY	BOOL	"1" = internal buffer is full "0" = internal buffer is not full (e.g. after warm/cold start or modification of deadtime)

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual mode
halt	BOOL	"1" =Halt mode

Operating mode

Selecting the operating modes

There are three operating modes, which are available via the man and halt parameter inputs:

Operating mode	man	halt
Automatic	0	0
Manual	1	0 or 1
Halt	0	1

Automatic operating mode

In the automatic mode, the function block operates according to the following rules:

If…	Then…
Scan time > $\frac{T\_Delay}{128}$	the current X value is transferred to the buffer, and the oldest X value in the buffer is placed on the output Y. If the scan time is more than T_DELAY / 128, resolution is less than 128 causing a systematic error, i.e. some X-values are double-stored (see the following Example).
Scan time < $\frac{T\_Delay}{128}$	not all X values can be stored in the buffer. In this case the X value is not saved in some cycles. After completion of T_DELAY, output Y may correspondingly remain unchanged in two (or more) consecutive cycles.

Example of automatic mode

In the example the following values are accepted:
Cycle time = 100 ms
T_DELAY = 10 s
tin = T_DELAY / 128 = 78 ms
As the reading time tin is shorter than the cycle time, each X value is transferred to the buffer. On the fourth execution of the function block (after 400 ms) the X value is saved twice rather than once (as 3 x 78 = 312 and 4 x 78 = 390).

Manual mode

In manual mode the manual value YMAN is consistently transferred to the control output Y. The internal buffer is charged with the manual value YMAN. The buffer is marked as charged (READY =1).

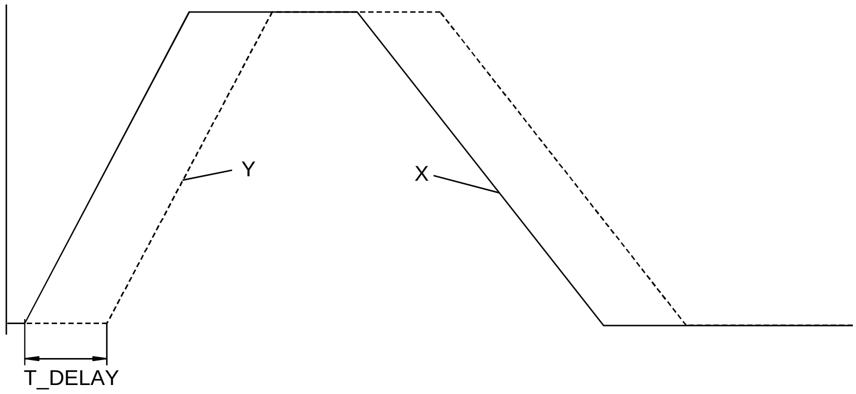
Halt mode

The output Y is held at the last calculated value in Halt mode. The output will no longer be changed, but can be overwritten by the user. The internal buffer still continues to operate as in automatic mode.

Example for behavior of the function block

Example

The following diagram shows an example for behavior of the function block. Input X follows a ramp function from one value to a new value. Delayed by the deadtime T delay, X values appear at Y.
DEADTIME function block diagram



Runtime error

Error message

An Error message, appears when an invalid floating point number lies at input YMAN or X.

DELAY: Deadtime device

8

Overview

At a glance

This chapter describes the DELAY block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

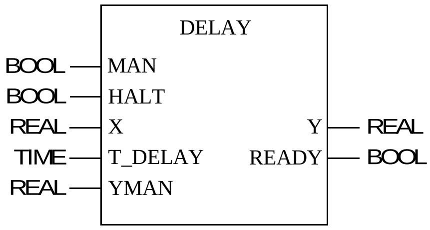
With this function block the input signal is delayed by a deadtime.
The function block delays the signal X by the deadtime T_DELAY before it appears again at Y.
The function block incorporates a delay buffer for 128 elements (X-values), meaning that during the time span T_DELAY 128 X-values can be stored. The buffer is used in accordance with the various operating mode.
The value of Output Y remains unchanged after cold and warm system starts. The internal values are set to the value of X.
After a change of deadtime T_DELAY or a cold or warm system start, the output READY goes to "0". This means: that the buffer is not ready because it is empty.
The function block has the following operating mode: Manual, halt and automatic mode.
EN and ENO can be projected as additional parameters.

Note: The delay time continues to run even if the block is disabled via the EN parameter, because the block calculates its time differences according to the system clock.

Representation

Symbol

Representation of the block



Parameter description

Block parameter description

Parameter	Data type	Meaning
MAN	BOOL	"1" = Manual mode
HALT	BOOL	"1" =Halt operating mode
X	REAL	Input value
T_DELAY	TIME	Deadtime
YMAN	REAL	Manual manipulated value
Y	REAL	Output
READY	BOOL	"1" = internal buffer is full "0" = internal buffer is not full (e.g. after warm/cold start or modification of deadtime)

Operating mode

Selecting the operating modes

There are three operating modes, which are selected via the inputs MAN and HALT.

Operating mode	MAN	HALT
Automatic	0	0
Manual	1	0 or 1
Halt	0	1

Automatic operating mode

In the automatic mode, the function block operates according to the following rules:

If	Then
Scan time > $\frac{T\_Delay}{128}$	the current X value is transferred to the buffer, and the oldest X value in the buffer is placed on the output Y. If a cycle time is greater than T_DELAY / a resolution of less than 128 will result, causing a systematic error leading to double storage of some X values. (see the following Example).
Scan time < $\frac{T\_Delay}{128}$	not all X values can be stored in the buffer. In this case the X value is not saved in some cycles, and Y remains unchanged in these cycles.

Example of automatic mode

In the example the following values are accepted:
Cycle time = 100 ms
T_DELAY = 10 s
tin = T_DELAY / 128 = 78 ms
As the reading time tin is shorter than the cycle time, each X value is transferred to the buffer. On the fourth execution of the function block (after 400 ms) the X value is saved twice rather than once (as 3 x 78 = 312 and 4 x 78 = 390).

Manual mode

In manual mode the manual value YMAN is consistently transferred to the control output Y. The internal buffer is charged with the manual value YMAN. The buffer is marked as charged (READY =1).

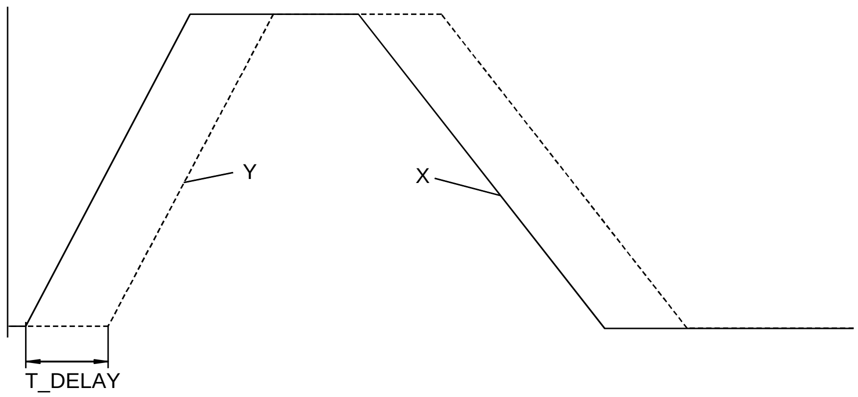
Halt mode

The output Y is held at the last calculated value in Halt mode. The output will no longer be changed, but can be overwritten by the user. The internal buffer still continues to operate as in automatic mode.

Example of the behavior of the function block

Example

The following diagram shows an example of the behavior of the function block. Input X follows a ramp function from one value to a new value. Delayed by the Deadtime T delay, X values appear at Y.
Diagram of the DELAY function block



DERIV: Differentiator with smoothing

9

Overview

At a glance

This chapter describes the DERIV block.

What's in this chapter?

This chapter contains the following topics:

Brief description

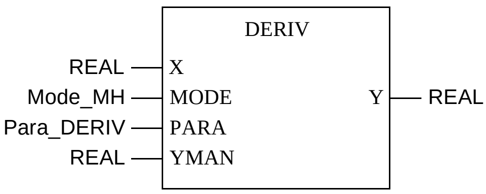
Function description

The function block is a differential element with a delayed output Y respecting the delay time constant lag.
The function block contains the following operating mode: Manual, halt and automatic mode.
EN and ENO can be projected as additional parameters.

Representation

Symbol

Representation of the block



Parameter description DERIV

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input variable
MODE	Mode_MH	Operating Modes
PARA	Para_DERIV	Parameter
YMAN	REAL	Manual manipulated value
Y	REAL	Output derivative unit with smoothing

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual mode
halt	BOOL	"1" =Halt operating mode

Parameter description Para_DERIV

Data structure description

Element	Data type	Meaning
gain	REAL	Gain of the differentiation
lag	TIME	Delayed time constants

Formulas

Transmission function

The transfer function for Y is:

$$G(s) = gain \times \frac{s \times lag}{1 + s \times lag}$$

Calculation formula for Y

The calculation formula for Y is:

$$Y = \frac{lag}{dt + lag} \times (Y_{(old)} + gain \times (X_{(new)} - X_{(old)}))$$

Special case: lag = 0

This amounts to pure differentiation without a 1st order time limiter.
In this situation the transfer function is:

$$G(s) = gain \times s$$

The formula of calculation is:

$$Y = gain \times \frac{X_{(new)} - X_{(old)}}{dt}$$

Meaning of the sizes

The meaning of the formula sizes is asfollows:

size	Meaning
$X_{(new)}$	the input X value for the current cycle
$X_{(old)}$	the input X value from the previous cycle
$Y_{(old)}$	the output Y value from the previous cycle
dt	is the time differential between the current cycle and the previous cycle

Detailed description

Parametering

The parameter assignments of the function block are effected by the determination of gain, the differentiator and the time constant lag, by which the output Y is delayed. For very short sampling times and an input X unit step (input X jumps from 0 to 1.0), the output Y jumps to the value gain (in theory _ in reality somewhat smaller, due to the sampling time not being infinitely small), and then returns to 0 with the delay time constant lag.

Operating mode

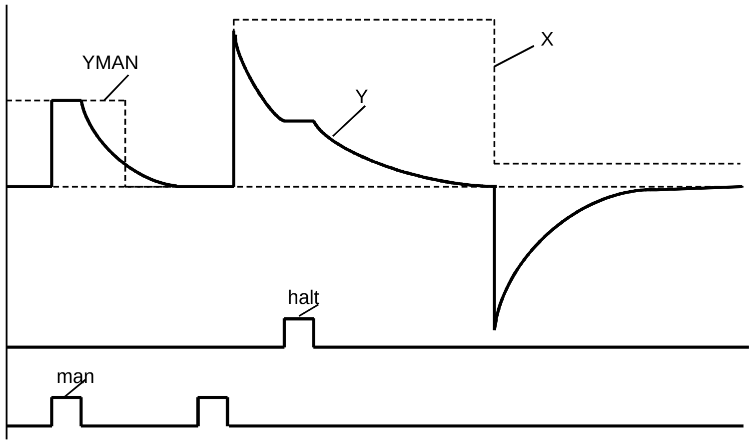
There are three operating modes selectable via the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual	1	0 or 1	The input YMAN will be transferred directly to the output Y.
Halt	0	1	The output Y will be held at the last calculated value. The output remains at this value, but can still be overwritten by the user.

Example for the function block

DERIV example

The following example shows the step response of the DERIV function block.
Jump response with gain = 1 and lag = 10 s



Runtime error

Error message

An Error message, appears when an invalid floating point number lies at input YMAN or X.

DTIME: Delay

10

Overview

At a glance

This chapter describes the DTIME block.

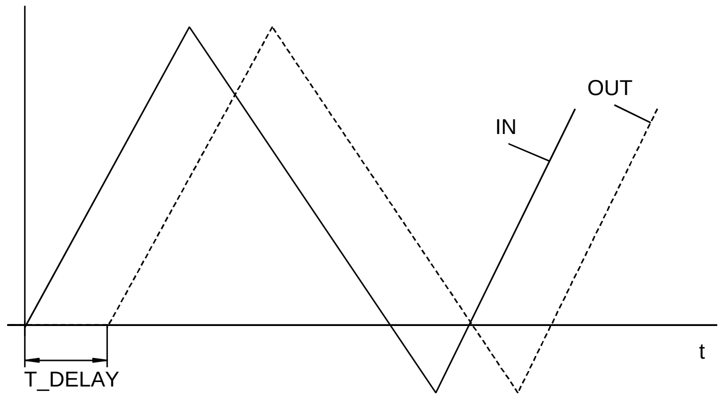
What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The function blockDTIME generates a delay when numerical input variables are transferred. The numerical output variable OUT generates the same behavior as the numerical input variable when the delay T_DELAY, which can vary, is included.
Behavior of the DTIME function block:



EN and ENO can be projected as additional parameters.

Formula

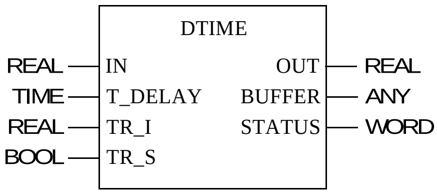
This function block implements the following transfer function :

$$G_{(p)} = e^{-p.T\_DELAY}$$

Representation

Symbol

Representation of the block



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Digital value to be delayed
T_DELAY	TIME	Desired delay
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization command
OUT	REAL	Delayed output
BUFFER	ANY*)	Memory for the purpose of storing delayed values.
STATUS	WORD	Status word

*) It is essential for this to be linked to a variable (see"*Parametering, p. 85*").

Parametering

Saving the input values (BUFFER output)

The BUFFER output must be linked to a variable (generally of the Buffer_DTIME) type. The values to be delayed are contained in these variables. Each time the function block is executed a new value is saved for the IN input.
The size of the variable linked to the BUFFER output determines the number of values which can be saved and therefore also the allowable maximum delay value.

$$T\_DELAY_{maximum} = n \times T\_Period$$

The following applies here

Formula size	Meaning
n	Number of real values which the BUFFER can contain.
T_PERIOD	Sampling interval of the function block

Note: As soon as a variable has been linked to the BUFFER output, it can only be replaced by a variable of the same type. To replace it with a greater variable, which would enable a higher delay value to be reached for example, the function block must be deleted and a new one put in place.

Data type of the buffer output

The BUFFER output is of the ANY type. This means any variable type can be assigned to it. It is generally an advantage to use a variable of the Buffer_DTIME type at first. This also involves a table containing up to 100 real values. With this variable type it is possible to attain a delay which corresponds to 100 times the sampling interval of the DTIME function block.

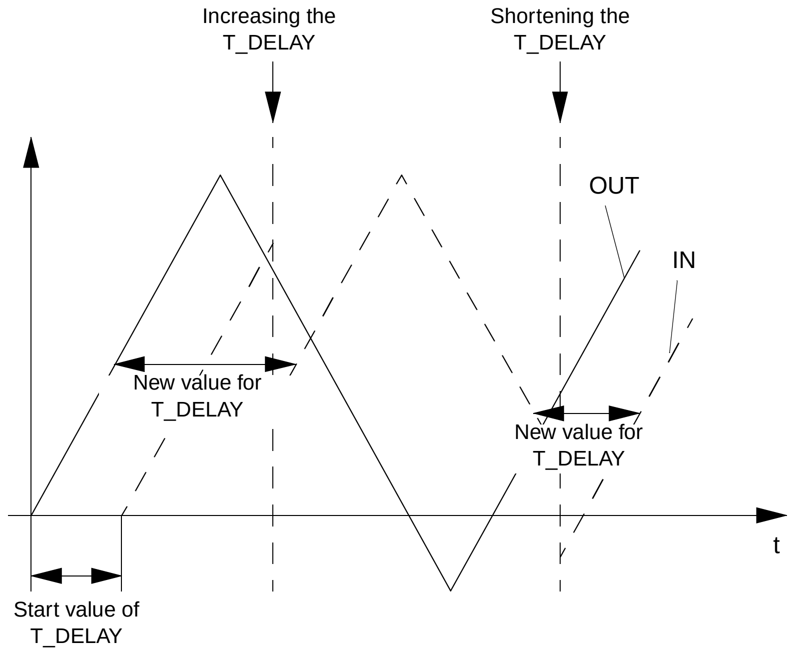
Procedure for large delay times

To attain delay values which are equivalent to over 100 times the sampling interval of the function block, a larger variable must be assigned to the BUFFER parameter:

Step	Action
1	Define a new derived data type, e.g. a table with 200 floating point values
2	Declare a variable of this type and link it to the BUFFER parameter of the DTIME function block.
3	In this case the maximum delay corresponds to 200 times the sampling interval of the function block

Dynamic modification of the T_DELAY delay

It is possible to raise or lower the T_DELAY delay time while the program is running. As long as the re-adjusted delay time is compatible with the size of the BUFFER output, the new delay is effective immediately.
Presentation of the dynamic modification of the T_DELAY delay



If the T_DELAY value is too great in relation to the BUFFER size, it is no longer possible to save enough input values to attain the delay desired. In this case the delay remains at the longest time possible (bit 8 of the status word then goes to 1 over).
To prevent this problem it is advisable to define the dimensions of the variable assigned to the BUFFER parameter so that a possible increase in the T_DELAY can be provided for.
When T_DELAY = 0, the OUT output always corresponds to the IN input.

Initialization and operating mode

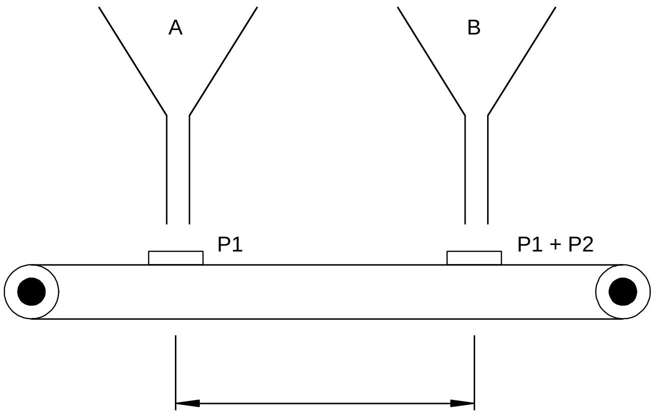
Initialization and operating mode

The first time the function block is executed (when loading the program or during online calls), all the values contained in the buffer are initialized with the value of TR_I. The OUT output retains this value for the duration of the T_DELAY. If the TR_I input is not attached, the value 0 serves to initialize the BUFFER output and the OUT output retains the value 0 during the T_DELAY.
In the tracking operating mode (TR_S = 1), the input TR_I is transferred to the OUT output and the BUFFER output is also initialized with the value of TR_I. After returning to normal operating mode, the output retains this value for the duration of T_DELAY, as was the case with the first cycle.

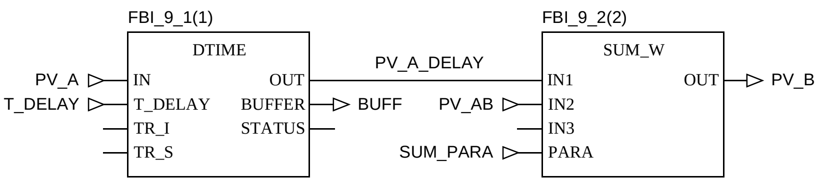
Example for measuring a rate of flow

Measuring a rate of flow

The DTIME function block can be used for example to model a process delay, whose uses include a design to measure flow rates or the number of revolutions of propulsion systems.
In the following example two products, A and B, are poured into a container one after the other and mixed. First, the container is placed under the dosing device for product A, to give the amount P1. Then it is moved on a conveyor belt to the dosing device for product B to give the amount P2. The time interval between the two dosing devices is 20 s.
Measuring flow rates



The product amount P2 is regulated, but the weight in the container is P1+P2. P1 should be removed. The amount P2 corresponds to the amount measured minus the amount P1 dosed 20 s beforehand.
Measuring the servo loop at P2 corresponds to the following illustration:



Values of the data structure elements of the SUM_PARA variables:

Element of SUM_PARA	value
SUM_PARA.K1	1
SUM_PARA.K2	1

Runtime error

Status word

In the status word the following messages are displayed:

Bit	Meaning
Bit 0 = 1	Error in a calculation with floating point values
Bit 1 = 1	Invalid value recorded at one of the floating point value inputs
Bit 2 = 1	Division by zero with calculation in floating point values
Bit 3 = 1	Capacity overflow with calculation in floating point values
Bit 8 = 1	T_DELAY exceeds the maximum value that can be represented on the BUFFER output.

Error message

This error appears if a non floating point value is inputted or if there is a problem with a floating point calculation. In this case the outputs OUT and BUFFER remain unchanged.

Alert

There will be an alert if a T_DELAY exceeds the maximum possible value. In this case the function block uses the maximum value. If an outgoing value is required, which is above the default value, only the BUFFER output needs to be linked to a larger variable.

FGEN: Function generator

11

Overview

At a glance

This chapter describes the FGEN block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

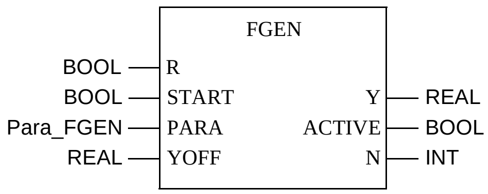
TheFunction block FGEN represents a function generator. It generates a signal form at output Y which is defined in the data structure Para_FGEN. The function block can be cascaded, i.e. if several of these EFBs are used, various signal forms can be created and laid over one another.
The following 8 different signal forms can be generated:
- Jump function
- Ramp function
- Delta function
- Saw-tooth function
- Square wave function
- Trapezoid function
- Sine function
- Random Number

As additional parameters, EN and ENO can be projected.

Representation

Symbol

Block representation



Parameter description FGEN

Block parameter description

Parameter	Data type	Meaning
R	BOOL	"1": Reset
START	BOOL	1": Start function generator
PARA	Para_FGEN	Parameter
YOFF	REAL	Output Y offset
Y	REAL	Function generator output
ACTIVE	BOOL	ACTIVE = 1: Function generator is active
N	INT	Number of intervals since start

Parameter description Para_FGEN

Data structure description

Element	Data type	Meaning
func_no	INT	Generator function choice (1-8)
amplitude	REAL	Function amplitude
halfperiod	TIME	Half cycle duration
t_off	TIME	Idle time constant
t_rise	TIME	Rise time constant
t_acc	TIME	Smoothing time
unipolar	BOOL	"1 "= Signal unipolar "0 "= Signal bipolar

Parametering

Reset

Parameter R stands for RESET. If this parameter is set (R = 1), all running functions will be immediately terminated and output Y goes to the value of parameter YOFF (offset). Simultaneously the cycle counter N is also reset to 0 and ACTIVE returns to "0".

Starting the function generator.

The parameter START (START = 1) starts the function defined with the data structure. Output N is incremented with the beginning of each new cycle. If the parameter START returns to "0", the active cycle of the selected function runs to completion. As long as a function runs, the output ACTIVE is 1. If the period ends the output ACTIVE is reset to 0.

Offset

Waveforms produced by the function generator have an amplitude with the value of parameter "amplitude", i.e. values range from "amplitude" to -"amplitude" for bipolar operation (unipolar = "0") resp. from 0 to "amplitude" in unipolar operation (unipolar = "1"). Waveform values can be shifted away from the zero reference point through the parameter YOFF.

Note: Should the output of another function generator be applied to parameter YOFF, the waveforms produced by both function generators are overlaid.

Rise time t_rise

Rise time t_rise is used only by the functions "ramp" and "trapezoid". In the "saw-tooth" function rise time is determined by halfperiod - t_off. Rise time is 0.5 * (halfperiod - t_off) for the "delta" function.

Function selection

Selection

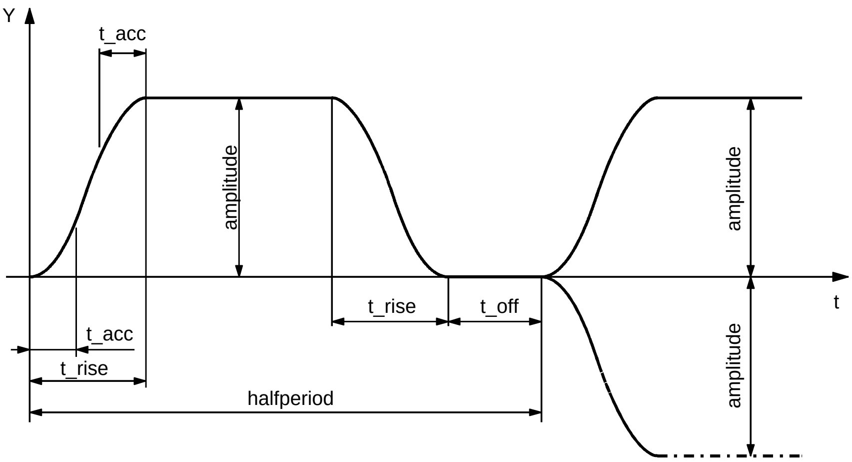
There are a total of 8 functions which can be produced by the function generator. Function selection is made through func_no. At a function change the last selected running function still proceeds to completion.
The following function numbers are allowed:

func_no	Function
1	Jump
2	Ramp
3	Saw-tooth
4	Delta
5	Square
6	Trapezoid
7	Sine
8	Random Number

Function definition

Definition

The function is defined completely in the data structure Para_FGEN. First of all the waveform must be determined (refer to *Function selection, p. 94*).
Trapezoid (Delta, Saw-tooth, Square) unipolar/bipolar is selected as the basic type for the definition.

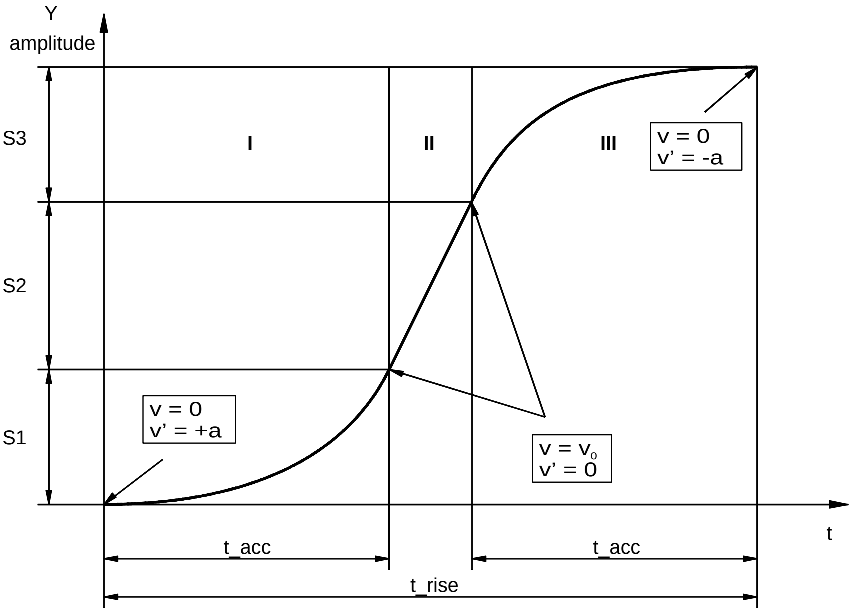


Function amplitude is determined in the parameter amplitude. It should be noted that this declaration applies to unipolar operation. Amplitude in bipolar operation is doubled and consists of amplitude and -amplitude.
The parameter halfperiod defines the half cycle duration.
Parameter t_off defines an idle time. A half cycle of the function is then output within the time halfperiod - t_off.
For the trapezoid function definition the rise time t_rise is also required. This is the time in which the signal should accelerate from 0 to amplitude. This time is also taken for the descent from amplitude back to 0.

"Smoothing" a function

If a function in ramp form is to rise or decline, the transitions are first of all always made in a sharp crease. The gradient is not constant in this case. "Smoothing" is used to achieve a soft rise and descent, i.e. the ramp turns into an S-curve.

"Smoothing" a function



This is then divided into three sections. Section I "accelerates" directly from 0. Section II is traversed with the velocity attained at the end of section I. In section III, the acceleration from section I is used to brake, and thus approach the terminal point softly. The size of the section is user-definable. They are defined by specifying t_acc and t_rise.

The acceleration involved is calculated by the following formulas:

$$\text{amplitude} = S1 + S2 + S3$$

with

$$S3 = S1 = \frac{a}{2} \times t\_acc^2$$

and

$$S2 = a \times t\_acc \times (t\_rise - 2 \times t\_acc)$$

It then follows that:

$$a = \frac{\text{amplitude}}{t\_acc \times t\_rise - t\_acc^2}$$

Note: Smoothing is used only by the functions "Ramp", "Saw-Tooth", "Delta" and "trapezoid". "Jump", "Square" and "Sine" are not "smoothable" functions.

Individual Parameter Usage

Parameter use within the various functions.

Function	amplitude	halfperiod	t_off	t_rise	t_acc	unipolar
Jump	X					
Ramp	X			X	X	
Saw-tooth	X	X	X	halfperiod - t_acc	X	X
Delta	X	X	X	(halfperiod - t_acc)/2	X	X
Square	X	X	X			X
Trapezoid	X	X	X	X	X	X
Sine	X	X	X			X
Random number	X					X

Function diagrams can be found in the section *Diagrams of the individual functions, p. 98*.

Unipolar operation

The unipolar parameter defines whether the selected function should be output as a unipolar or bipolar function. Particular attention should be paid to the fact that in unipolar operation a cycle is still characterized by 2 "unipolar" half waves.

Altering function parameters

During a currently executing cycle, all function parameters may be altered. However, any alterations made will not take effect until the cycle has completed. Should, for example, the idle time t_off be altered during the running cycle, it does not apply until the start of the next cycle.

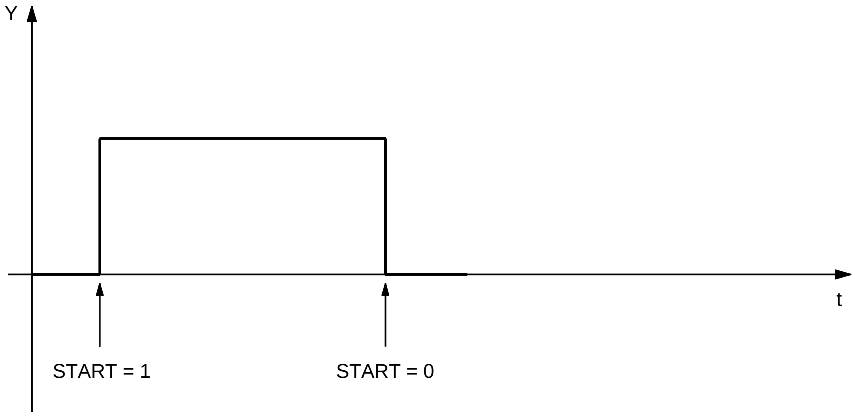
Altering a function

If the parameter func_no is changed during a currently executing cycle, it will also not take effect until the cycle has completed with the previously selected function. The new function is then started. This resets the cycle counter N, which indicates the period number, to 0.

Diagrams of the individual functions

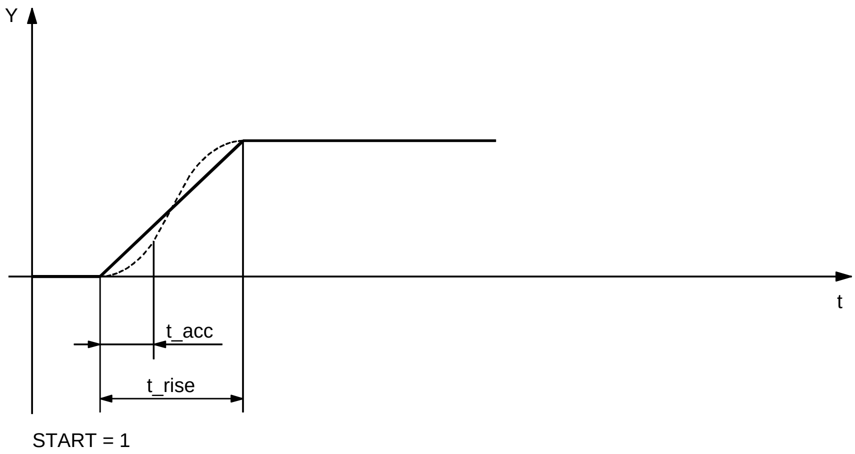
Jump function

Representation of the Jump function



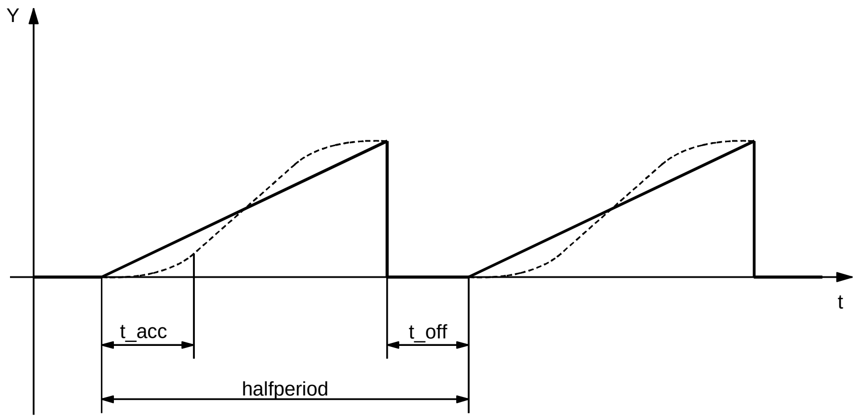
Ramp function

Representation of the Ramp function



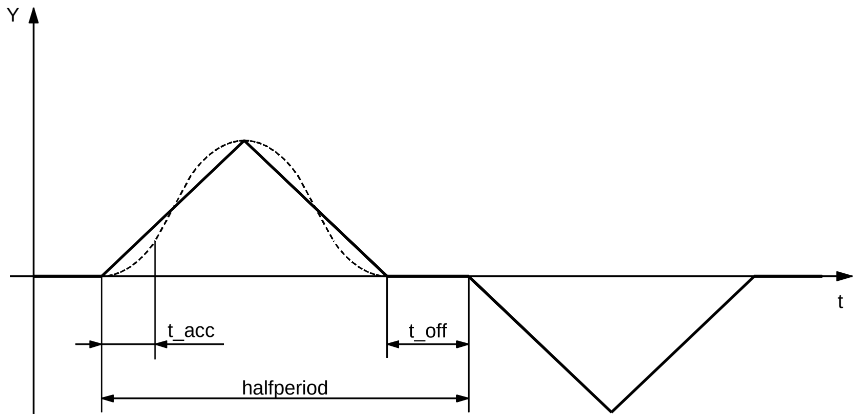
Saw-tooth function

Representation of the Saw-tooth function



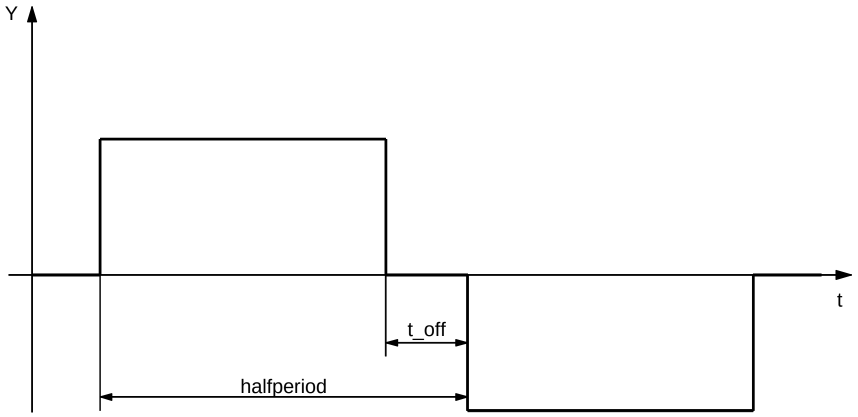
Delta function

Representation of the Delta function



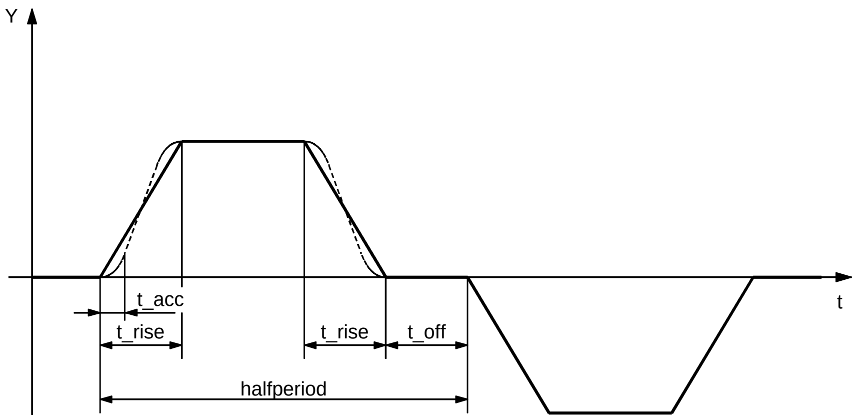
Square wave function

Representation of the Square wave function



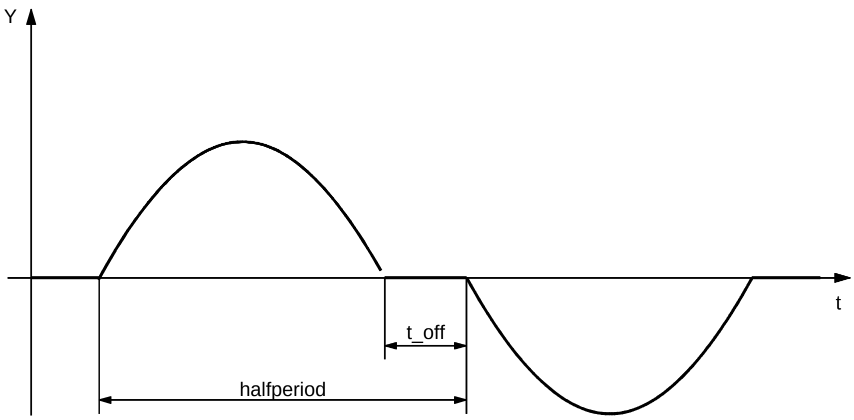
Trapezoid function

Representation of the Trapezoid function



Sine function

Representation of the Sine function



Special cases

Jump function

On the "Jump" function the output goes to
the value Y = OFF if START = 0
and
the value Y = OFF + amplitude if START = 1
set
The time specifications (t_off, t_rise, t_acc) do not play a role in this function.
Output N is incremented for every new 0 →1 transition of input START.
There is no bipolar mode for this function, i.e. the unipolar parameter value is disregarded.

Ramp function

In the "Ramp" function output Y ramps upward from value YOFF to YOFF + amplitude. While START is unchanged at 1, output Y remains at the value YOFF + amplitude. Output Y jumps back to value YOFF should START be taken back to 0. Run up is determined by the times t_rise and t_acc. The time needed for run up from Y = YOFF to Y = YOFF + amplitude is specified by t_rise. "Smoothing" can be influenced by t_acc.
Output N is incremented for every new 0 →1 transition of input START.
There is no bipolar mode for this function, i.e. the unipolar parameter value is disregarded.

Random number

In the "Random number" function output Y is set to a number resulting "by chance" between
YOFF ≤ Y ≤ YOFF + amplitude, in unipolar operation
and
YOFF - amplitude ≤ Y ≤ YOFF + amplitude, when the operation is bipolar
.
The time specifications (t_off, t_rise, t_acc) do not play a role in this function.
Output N is incremented for every new 0 →1 transition of input START.

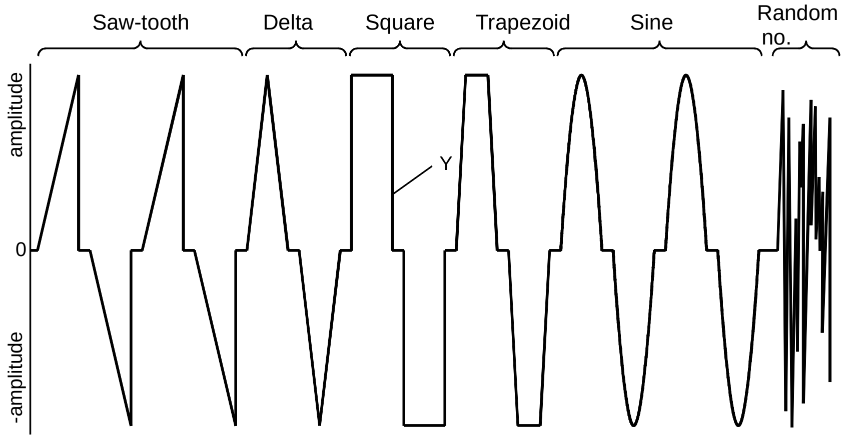
Timing diagrams

Bipolar operation

The following parameter specifications represent the various functions in bipolar operation:

Parameter	Specification
amplitude	1
halfperiod	10
t_off	2
t_rise	2
t_acc	0
unipolar	0

Bipolar operation

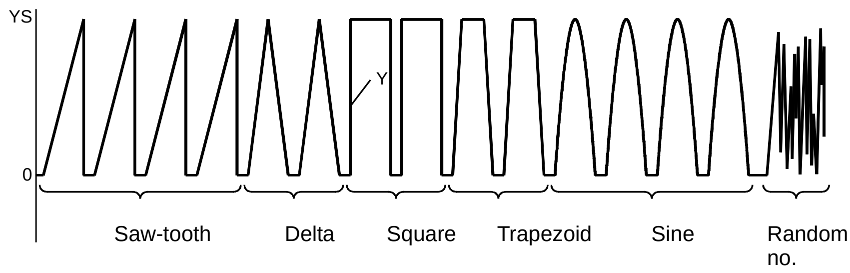


Unipolar operation

The following parameter specifications represent the various functions in unipolar operation:

Parameter	Specification
amplitude	1
halfperiod	10
t_off	2
t_rise	2
t_acc	0
unipolar	1

Unipolar operation

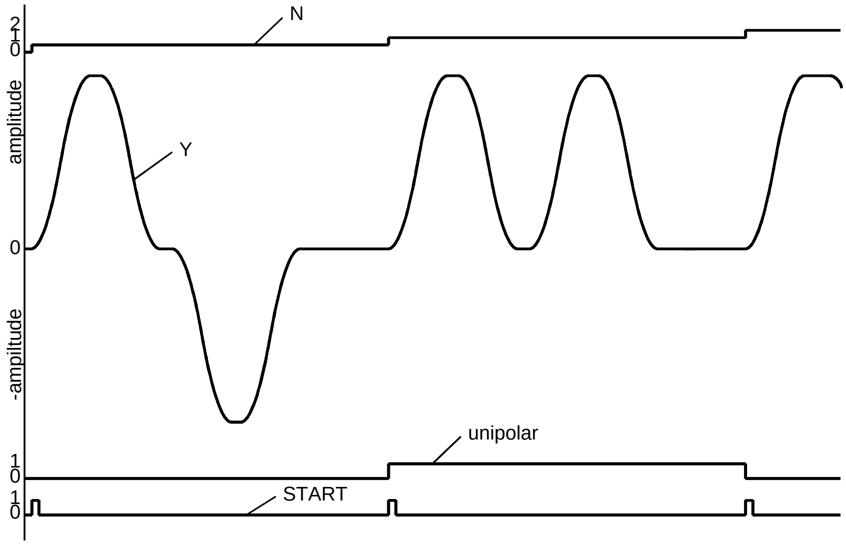


Trapezoid function

The following parameter specification represents the trapezoid function:

Parameter	Specification
amplitude	1
halfperiod	10
t_off	1
t_rise	4
t_acc	1.5

Trapezoid function



INTEG: Integrator with limit

12

Overview

At a glance

This chapter describes the INTEG block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block replicates a limited integrator.
The function block has the following properties:
- Operating modes, Manual, Stop, Automatic
- Manipulated variable limiting in automatic mode

As additional parameters, EN and ENO can be projected.

Formula

The transfer function is:

$$G(s) = \frac{gain}{s}$$

The formula of calculation is:

$$Y = Y_{(old)} + gain \times dt \times \frac{X_{(new)} + X_{(old)}}{2}$$

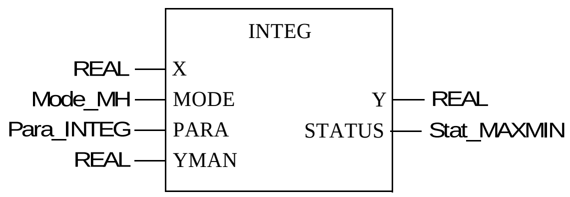
Meaning of the sizes

Size	Meaning
$X_{(old)}$	Value of input X from the previous cycle
$Y_{(old)}$	Value of output Y from the previous cycle
dt	Time difference between current and previous cycle

Representation

Symbol

Block representation



Description of the INTEG parameter

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input variable
MODE	Mode_MH	Operating modes
PARA	Para_INTEG	Parameter
YMAN	REAL	Manually manipulated value
Y	REAL	Output
STATUS	Stat_MAXMIN	Output status

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual operating mode
halt	BOOL	"1" =Halt operating mode

Parameter description Para_INTEG

Data structure description

Element	Data type	Meaning
gain	REAL	Integral gain (units/second)
ymax	REAL	Upper limit
ymin	REAL	Lower limit

Parameter description Stat_MAXMIN

Data structure description

Element	Data type	Meaning
qmin	BOOL	"1" = Y has reached lower limit
qmax	BOOL	"1" = Y has reached upper limit

Detailed description

Parametering

The parameter assignments of the function block are satisfied by the determination of gain, the integral gain and the limiting values ymax und ymin for output Y.
The values ymax and ymin limit the upper and lower values of the output. So that means ymin ≤ Y ≤ ymax
If the threshold value is reached or the output signal is limited this will be indicated by qmax and qmin.
- qmax = 1 if Y ≥ ymax
- qmin = 1 when Y ≤ ymin

Operating mode

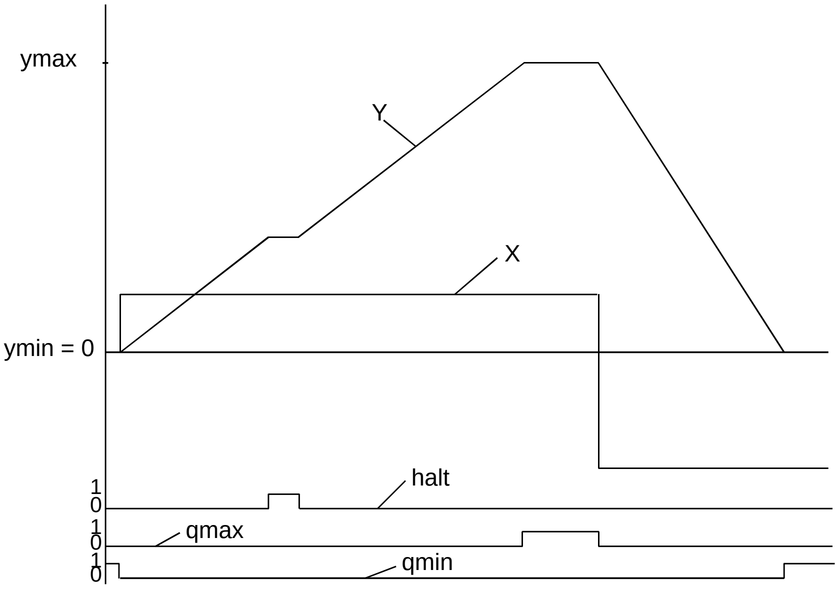
There are three operating mode selectable through the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual mode	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y. The control output is, however, limited by ymax and ymin.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can, however, be overwritten by the user.

Example

The input signal is integrated via the time. The output follows jumps of the input X value in a ramp function of like polarity. Limiting of output Y within ymax and ymin with the appropriate signals at qmax and qmin can also be clearly seen.

Representation of the integrator jump response



Runtime error

Error message

There is an Error message, if
- an unauthorized floating point number is placed at the input YMAN or X,
- ymax < is ymin

INTEGRATOR: Integrator with limit

13

Overview

At a glance

This chapter describes the INTEGRATOR block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block replicates a limited integrator.
The function block has the following properties:

- Tracking and automatic modes
- Manipulated variable limiting in automatic mode

EN and ENO can be configured as additional parameters.

Formulas

The transfer function is:

$$G(s) = \frac{GAIN}{s}$$

The formula for the output OUT is:

$$OUT = OUT_{(old))} + GAIN \times dt \times \frac{IN_{(new)} + IN_{(old)}}{2}$$

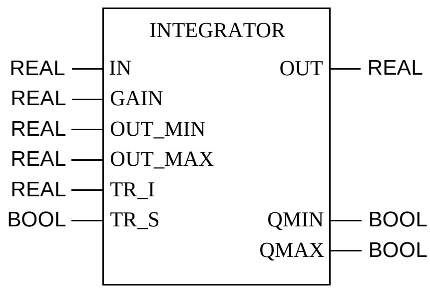
Meaning of variables

Variable	Meaning
$IN_{(new)}$	current value of input IN
$IN_{(old)}$	Value of the input IN from the previous cycle
$OUT_{(old)}$	Value of the output OUT from the previous cycle
dt	Time difference between the current cycle and the previous cycle

Display

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Input variable
GAIN	REAL	Integral gain
OUT_MIN	REAL	Lower output limit
OUT_MAX	REAL	Upper output limit
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization type "1" = Tracking mode "0" = Automatic mode
OUT	REAL	Output
QMIN	BOOL	"1" = Output OUT has reached lower limit
QMAX	BOOL	"1" = Output OUT has reached upper limit

Detailed description

Parametering

Parameter assignment for the function block is accomplished by specifying the integration gain GAIN and the limiting values OUT_MAX and OUT_MIN for the output OUT.
The limits OUT_MAX and OUT_MIN retain the output within the prescribed range. So that means OUT_MIN ≤ OUT ≤ OUT_MAX.
The markers QMAX and QMIN are signalling that the limits or a limitation of the output signal have/has been reached.

- QMAX = 1 if OUT ≥ OUT_MAX
- QMIN = 1 if OUT ≤ OUT_MIN

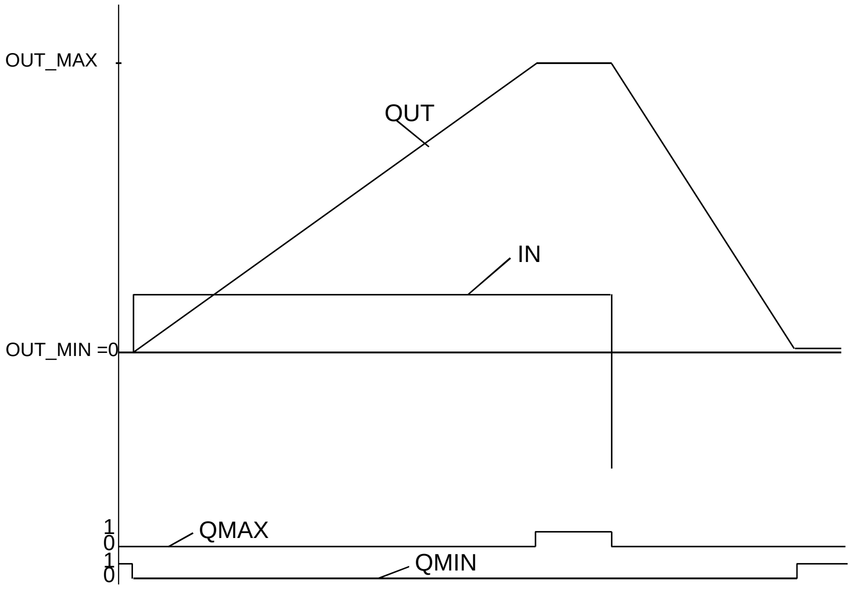
Operating mode

There are two operating mode selectable through the TR_S parameter input.

Operating mode	TR_S	Meaning
Automatic	0	The Function block operates as described in "Parametering".
Tracking	1	The tracking value TR_I is transferred permanently to the output OUT. The control output is, however, limited by OUT_MAX and OUT_MIN.

Example

The input signal is integrated using the time. In the event of a transition at the input IN, the output will rise (if the IN values are positive) or fall off (if the IN values are negative) along a ramp function. OUT will always be between OUTMAX and OUT_MIN; if OUT is equal to OUT_MAX or OUT_MIN, it will be so indicated in QMAX or QMIN.
It displays the integrator jump response:.



Runtime error

Error message

If OUT_MAX < OUT_MIN an Error message is generated.

INTEGRATOR1: Integrator with limit

14

Overview

At a glance

This chapter describes the INTEGRATOR1 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block replicates a limited integrator.
The function block has the following properties:
- Manual, halt and automatic modes
- Manipulated variable limiting in automatic mode

EN and ENO can be configured as additional parameters.

Formulas

The transfer function is:

$$G(s) = \frac{GAIN}{s}$$

The formula for the output Y is:

$$Y = Y_{(old))} + GAIN \times dt \times \frac{X_{(new)} + X_{(old)}}{2}$$

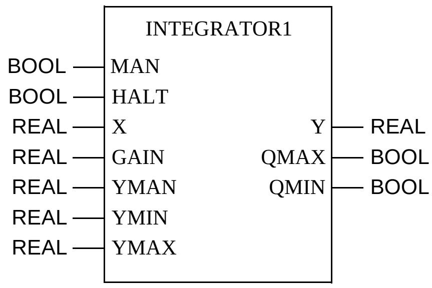
Meaning of variables

Variable	Meaning
$X_{(old)}$	Value of the input X from the previous cycle
$Y_{(old)}$	Value of the output Y from the previous cycle
dt	Time difference between current and previous cycle

Display

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
MAN	BOOL	"1" = Hand mode
HALT	BOOL	"1" = Halt mode
X	REAL	Input variable
GAIN	REAL	Integral gain
YMAX	REAL	Upper output limit
YMIN	REAL	Lower output limit
YMAN	REAL	Manual manipulated value
Y	REAL	Output
QMAX	BOOL	"1" = Output Y has reached upper limit
QMIN	BOOL	"1" = Output Y has reached lower limit

Detailed description

Parametering

The parametering of the function block is accomplished by specifying the integral gain GAIN and the limiting values YMAX and YMIN for the output Y.
The limits YMAX and YMIN retain the output within the prescribed range. Hence, YMIN ≤ Y ≤ YMAX.
The outputs QMAX and QMIN signal that the output has reached a limit, and thus been capped.

- QMAX = 1 if Y ≥ YMAX
- QMIN = 1 if Y ≤ YMIN

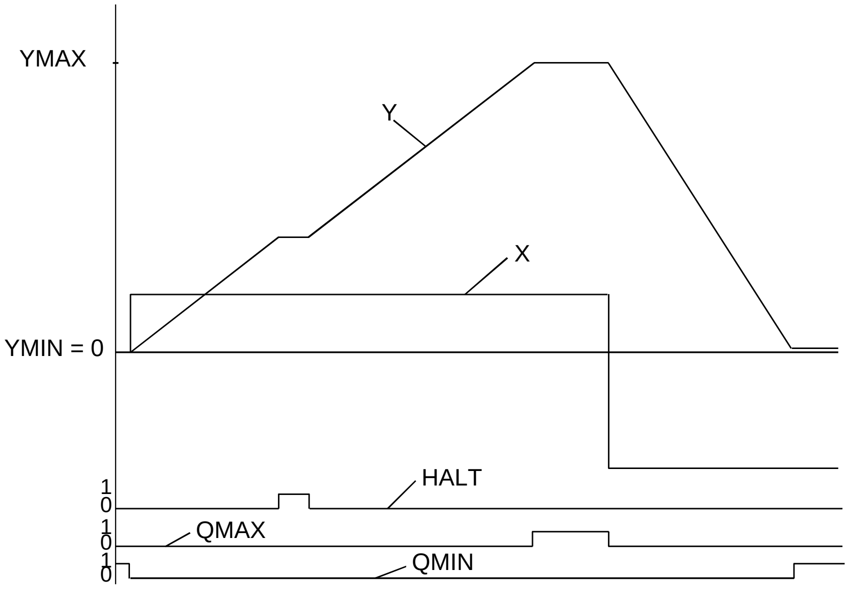
Operating mode

There are three operating mode selectable through the inputs MAN and HALT:

Operating mode	MAN	HALT	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual	1	0 or 1	The manual value YMAN will be transferred directly to the output Y. The control output is, however, limited by YMAX and YMIN.
Halt	0	1	The output Y will be set at the last calculated value. The output remains at this value, but can still be overwritten by the user.

Example

The input signal is integrated via the time. The output follows jumps of the input X value in a ramp function of like polarity. Limiting of output Y within YMAX and YMIN with the appropriate signals at QMAX and QMIN can also be clearly seen. Representation of the integrator jump response:.



Runtime error

Error message

If YMAN < YMIN an Error message is generated.

K_SQRT: Square root

15

Overview

At a glance

This chapter describes the K_SQRT block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

This Function block calculates the weighted square root of a numerical value. A division can be defined under which the function block issues the value zero.
Taking the square root typically serves to linearize a flow measurement using a throttle device.
EN and ENO can be configured as additional parameters.

Formula

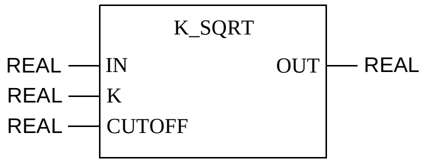
The function block performs the following calculation:

Calculation	Condition
$OUT = K\sqrt{IN}$	$IN \geq CUTOFF$
$OUT = 0$	$IN < 0$ or $IN < CUTOFF$

Presentation

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Numerical value to process
K	REAL	Weighting coefficient
CUTOFF	REAL	Division
OUT	REAL	Result of the calculation

Runtime error

Error message

An error is displayed if a non floating point value is recorded at input or if there is a problem with floating point calculation. In this case the output OUT remains unchanged.

Warning

A warning is given if the CUTOFF input is negative. The function block then uses the value 0 for calculation.

LAG: Time lag device: 1st order

16

Overview

At a glance

This chapter describes the LAG block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block represents a first order delay (low pass)
The function block contains the following operating mode:
- Manual
- Halt
- Automatic

EN and ENO can be projected as additional parameters.

Equation

The transmission function says:

$$G(s) = gain \times \frac{gain}{1 + s \times lag}$$

The calculation equation says:

$$Y = Y_{(old)} + \frac{dt}{lag + dt} \times \left(gain \times \frac{X_{(old)} + X_{(new)}}{2} - Y_{(old)}\right)$$

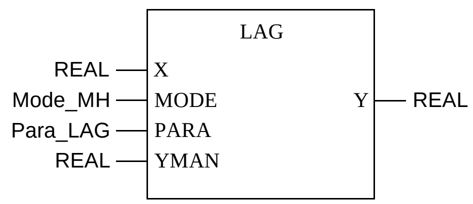
Meaning of the sizes

Size	Meaning
$X_{(old)}$	Value of output X from the previous cycle
$Y_{(old)}$	Value of the output Y from the previous cycle
dt	Time difference between current and previous cycle

Presentation

Symbol

Block display



Parameter description LAG

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input value
MODE	Mode_MH	Operating mode
PARA	Para_LAG	Parameter
YMAN	REAL	Manual manipulation
Y	REAL	Output

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Operating mode Hand
halt	BOOL	"1" = Halt mode

Parameter description Para_LAG

Data structure description

Element	Data type	Meaning
gain	REAL	Gain factor
lag	TIME	Delayed time constants

Detailed description

Parametering

The parametering of the Function block is achieved through specification of the boost factor gain as well as the parametering of the delayed time constants lag. The unit jump at input X (jump at input X of 0 to 1.0) succeeds the output Y with delay. Along an e-function

$$\exp(-t/lag)$$

it will approximate the value $gain \times X$.

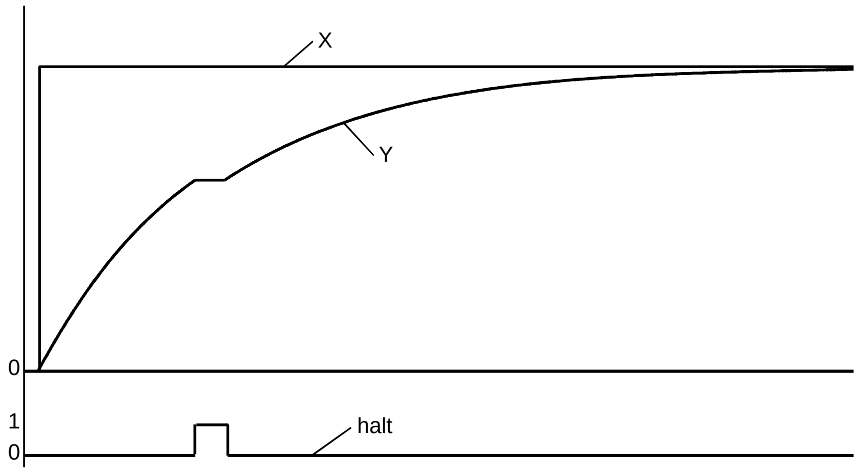
Operating mode

There are three operating modes selectable through the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual mode	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y.
Halt	0	1	The output Y will be set at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

Example

The diagram shows an example of the jump response of the function block. Input X jumps to a new value that output Y approaches exponentially.
Function block LAG jump response with gain = 1



LAG1: Time lag device: 1st order

17

Overview

At a glance

This chapter describes the LAG1 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block represents a first order delay.
The function block contains the following operating mode:
- Manual mode
- Halt
- Automatic

EN and ENO can be projected as additional parameters.

Equation

The transmission function says:

$$G(s) = \text{gain} \times \frac{1}{1 + s \times \text{lag}}$$

The calculation equation says:

$$Y = Y_{(old)} + \frac{dt}{LAG + dt} \times \left(\text{gain} \times \frac{X_{(old)} + X_{(new)}}{2} - Y_{(old)}\right)$$

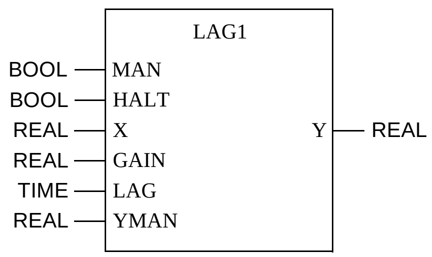
Meaning of the sizes

Size	Meaning
$X_{(old)}$	Value of output X from the previous cycle
$Y_{(old)}$	Value of the output Y from the previous cycle
dt	Time difference between current and previous cycle

Presentation

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
MAN	BOOL	"1" = Operating mode Hand
HALT	BOOL	"1" = Halt mode
X	REAL	Input value
GAIN	REAL	Gain factor
LAG	TIME	Delayed time constants
YMAN	REAL	Manual manipulation
Y	REAL	Output

Detailed description

Parametering

The parametering of the Function block is achieved through specification of the boost factor GAIN as well as the parametering of the delayed time constants LAG. The unit jump at input X (jump at input X of 0 to 1.0) succeeds the output Y delay. Along an e-function

$\exp(-t/(LAG))$

it will approximate the value $GAIN \times X$.

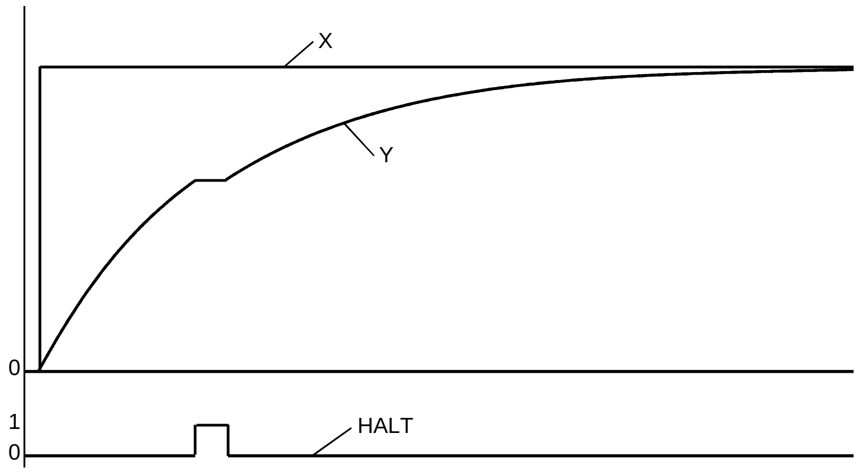
Operating mode

There are three operating mode, which are selected via the elements MAN and HALT:

Operating mode	MAN	HALT	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual mode	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can, however, be overwritten by the user.

Example

The diagram shows an example of the jump response of the PLAG device: Input X jumps to a new value that output Y approaches exponentially.
Function block LAG1 jump response with GAIN = 1



LAG2: Time lag device: 2nd order

18

Overview

At a glance

This chapter describes the LAG2 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block LAG2 represents a second order with delay.
The function block contains the following operating mode:
- Manual mode
- Halt
- Automatic

EN and ENO can be projected as additional parameters.

Equation

The transmission function says:

$$G(s) = gain \times \frac{1}{1 + s \times 2 \times \frac{dmp}{freq} + \left(\frac{s}{freq}\right)^2}$$

The calculation equation is as follows:

$$Y_{(new)} = A \times B$$

where

$$A = \frac{gain \times X \times (freq \times dt)^2 + Y_{(old)}}{1 + 2 \times dmp \times freq \times dt + (freq \times dt)^2}$$

and

$$B = \frac{(2 \times dmp \times freq \times dt \times 2) - Y_{(old2)}}{1 + 2 \times dmp \times freq \times dt + (freq \times dt)^2}$$

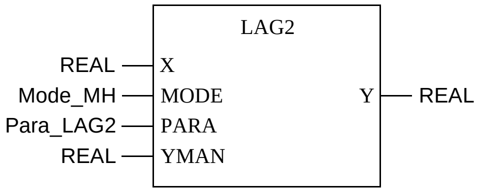
Meaning of the sizes

Size	Meaning
$Y_{(old)}$	Value of the output Y from the previous cycle
$Y_{(old2)}$	Value of the output Y from the cycle preceding the previous
dt	Time difference between current and previous call

Presentation

Symbol

Block display



parameter description LAG2

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input value
MODE	Mode_MH	Operating mode
PARA	Para_LAG2	Parameter
YMAN	REAL	Manual manipulated value for output
Y	REAL	Output

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Operating mode Hand
halt	BOOL	"1" = Halt mode

Parameter description Para_LAG2

Data structure description

Element	Data type	Meaning
gain	REAL	Gain factor
dmp	REAL	Dampening
freq	REAL	Natural frequency

Detailed description

Parametering

The parameter assignments of the function block are satisfied by the determination of gain, the gain and the values for dampening dmp, and natural frequency freq. Dampening dmp and natural frequency freq must have positive values.
Output Y follows input X jumps in a dampened oscillation. The period of undampened oscillation is T = 1/freq. For dampening values dmp < 1 reference is made to a dampened oscillation. For dampening values ≥ 1 reference is made to non-resonant behavior (i.e. without oscillation); in this case the output follows the input in the same way as with 2 LAG function blocks, which are switched in series.

Operating mode

There are three operating mode selectable through the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	The function block operates as described in "Parametering".
Manual mode	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

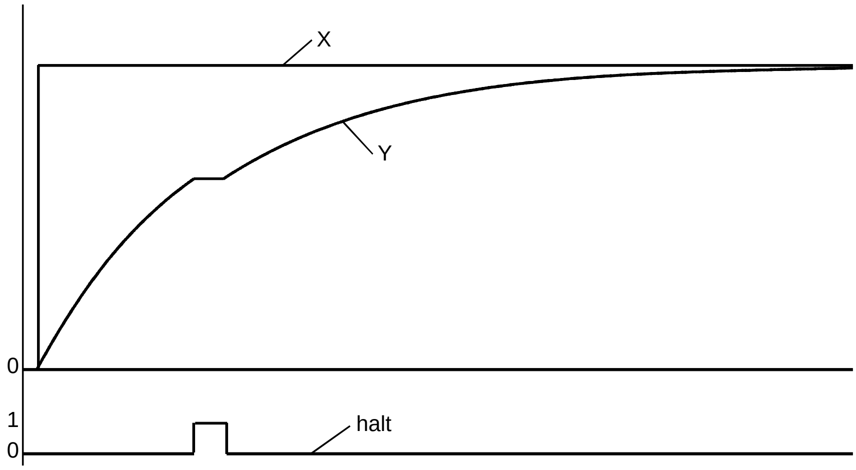
Timing diagrams

Overview

The following diagrams show examples of the LAG2 device's jump response with varying parameters.

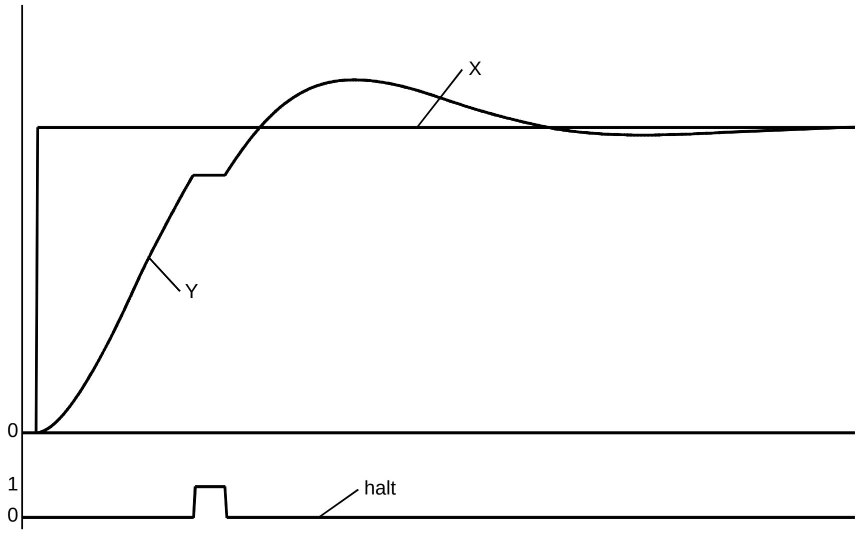
Dampening dmp = 1

For a dampening of dmp = 1 the output Y follows input X with a non-resonant action.



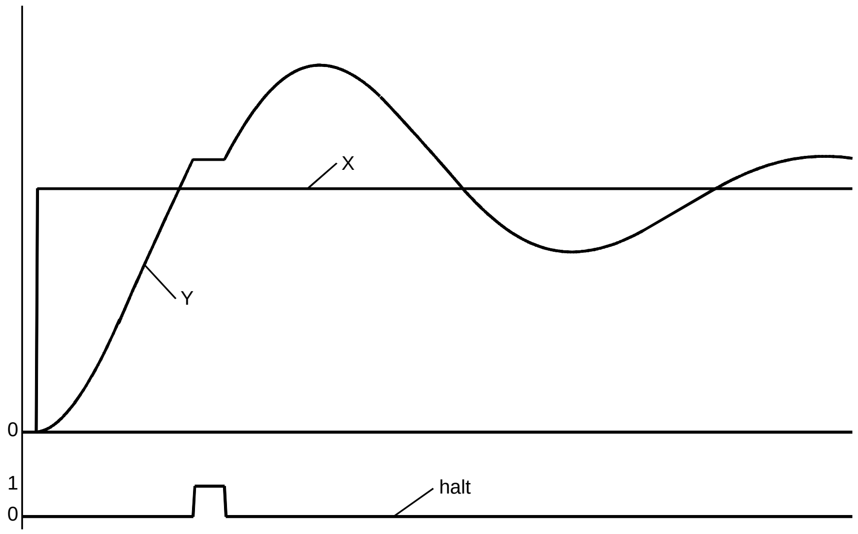
Dampening dmp = 0.5

For a dampening of dmp = 0.5 the output Y follows input X in a dampened periodic manner.



Dampening dmp = 0.2

For a dampening of dmp = 0.2 it is clear that the jump response is considerably less dampened.



LAG_FILTER: Time lag device: 1st order

19

Overview

At a glance

This chapter describes the LAG_Filter block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block represents a first order delay.
The function block contains the following operating mode:
- Tracking
- Automatic

EN and ENO can be projected as additional parameters.

Equation

The transmission function says:

$$G(s) = GAIN \times \frac{1}{1 + s \times LAG}$$

The calculation equation says:

$$OUT = OUT_{(old)} + \frac{dt}{LAG + dt} \times \left(GAIN \times \frac{IN_{(old)} + IN_{(new)}}{2} - OUT_{(old)}\right)$$

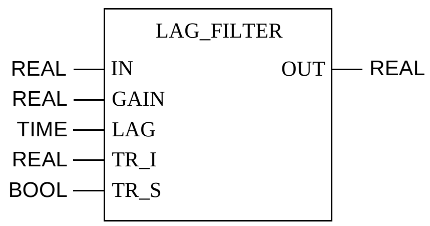
Meaning of the sizes

Size	Meaning
$IN_{(old)}$	Value of the input IN from the previous cycle
$OUT_{(old)}$	Value of the output OUT from the previous cycle
dt	Time difference between current and previous cycle

Representation

Symbol

Representation of the block



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Input value
GAIN	REAL	Gain factor
LAG	TIME	Delayed time constants
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization type "1" = Operating mode Tracking "0" = Halt mode
OUT	REAL	Output

Detailed description

Parametering

The parametering of the Function block is achieved through specification of the boost factor GAIN as well as the parametering of the delayed time constants LAG. The unit step at the input IN (jump at the input IN from 0 to 1.0) is followed by the output OUT with a lag time. Along an e-function

$\exp(-t/LAG)$

it will approximate the value $GAIN \times X$.

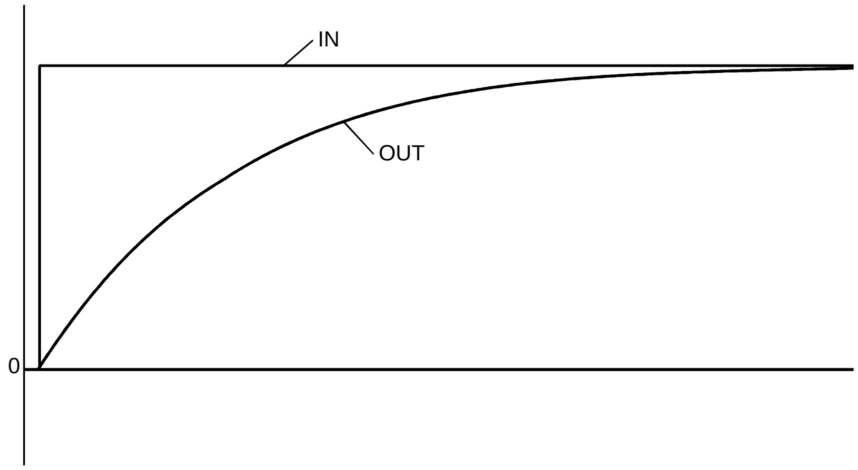
Operating mode

There are two operating mode, which can be selected via the input TR_S:

Operating mode	TR_S	Meaning
Automatic	0	The function block operates as described in "Parametering".
Tracking	1	The tracking value TR_I is transmitted permanently to the output OUT.

Example

The diagram shows an example of the jump response of the LAG_FILTER function block. The input IN jumps to a new value and the output OUT follows the input IN along an e-function.
Jump response of the function block LAG_FILTER when GAIN = 1



LDLG: PD device with smoothing

20

Overview

At a glance

This chapter describes the LDLG block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block serves as a PD outline with subsequent smoothing.
The function block has the following properties:

- Definable delay of the D-component
- Tracking and automatic modes

EN and ENO can be projected as additional parameters.

Formula

The transfer function is:

$$G(s) = GAIN \times \frac{1 + s \times LEAD}{1 + s \times LAG}$$

The formula of calculation is:

$$OUT = \frac{LAG \times OUT_{(old)} + GAIN \times ((LEAD + dt) \times IN - LEAD \times IN_{(old)})}{LAG + dt}$$

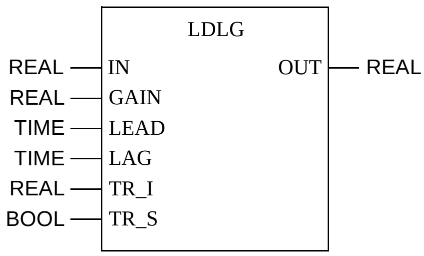
Meaning of the sizes

size	Meaning
$IN_{(old)}$	Value of the input IN from the previous cycle
$OUT_{(old)}$	Value of the output OUT from the previous cycle
dt	is the time differential between the current cycle and the previous cycle

Representation

Symbol

Representation of the block



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Input
GAIN	REAL	Gain factor
LEAD	TIME	Dirivative time constant
LAG	TIME	Delayed time constants
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization type "1" = Operating mode Tracking "0" = Halt mode
OUT	REAL	Output

Detailed description

Parametering

The parametering of the Function block appears through specification of the boost factors GAIN as well as the parametering of the Derivative time constants LEAD and the delayed time constants LAG.
For very small sample times and the unit jump to input IN (jump at line-in IN from 0 to 1.0) output OUT will jump to the value $\mathrm{GAIN} \times \mathrm{LEAD} / \mathrm{LAG}$ (theoretical value - actual slightly smaller, due to the not infinitely small sample times), using the time constant LAG to approximate the value $\mathrm{GAIN} \times 1.0$ closer.

Operating mode

There are two operating mode, which can be selected via the input TR_S:

Operating mode	TR_S	Meaning
Automatic	0	The function block operates as described in "Parametering".
Tracking	1	The tracking value TR_I is transmitted permanently to the output OUT.

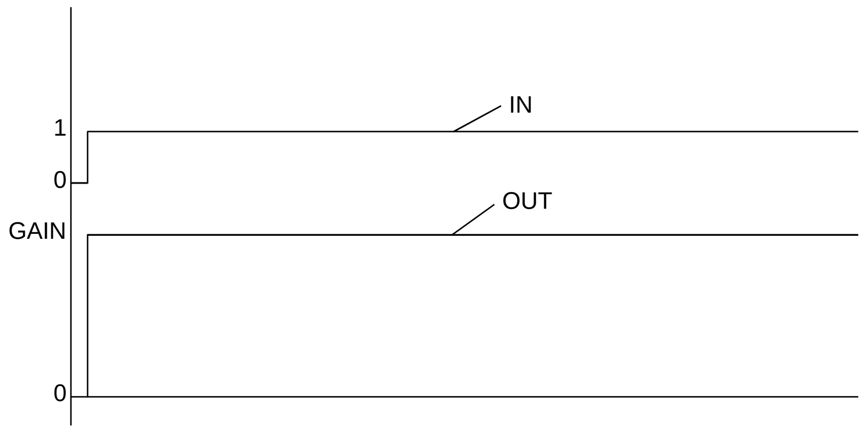
Examples of function block LDLG

Example-overview

The following examples are presented in the following diagrams:
- LEAD = LAG
- LEAD/LAG = 0,5, GAIN = 1
- LEAD/LAG = 2, GAIN = 1

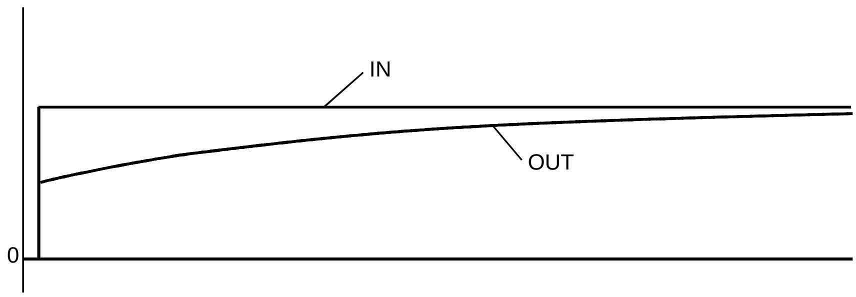
LEAD = LAG

The function block behaves like a pure multiplication block with the multiplier GAIN.
Function block LDLG with LEAD = LAG



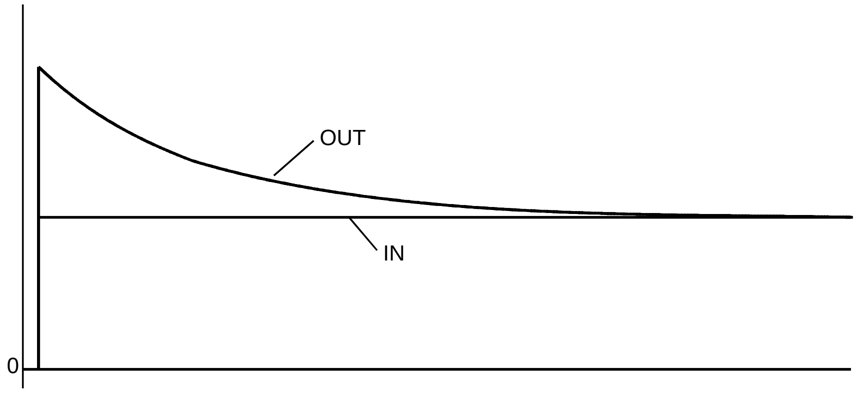
LEAD/LAG = 0,5, GAIN = 1

In this case the output OUT will jump to half the accumulated value in order to then transition to the upper range value (GAIN * IN) with the lag time constant LAG.
Function block LDLG with LEAD/LAG = 0,5 and GAIN = 1



LEAD/LAG = 2, GAIN = 1

In this case the output OUT will jump to twice the accumulated value in order to then transition to the end value (GAIN * IN) with the lag time constant LAG.
Function block LDLG with LEAD/LAG = 2 and GAIN = 1



LEAD: Differentiator with smoothing

21

Overview

At a glance

This chapter describes the LEAD block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The function block is a differentiator element with an output OUT delayed by the lag time constant LAG.
The function block contains the following operating modes:
- Tracking
- Automatic

EN and ENO can be projected as additional parameters.

Formula

The transfer function for OUT is:

$$G(s) = GAIN \times \frac{s}{1 + s \times LAG}$$

The formula of calculation is:

$$OUT = \frac{LAG}{dt + LAG} \times (OUT_{(old)} + GAIN \times (IN_{(new)} - IN_{(old)}))$$

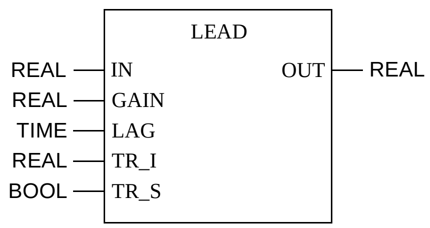
Meaning of the sizes

size	Meaning
$IN_{(new)}$	Value of the input IN from the current cycle
$IN_{(old)}$	Value of the input IN from the previous cycle
$OUT_{(old)}$	Value of the output OUT from the previous cycle
dt	is the time differential between the current cycle and the previous cycle

Representation

Symbol

Block representation



Parameter description

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Input value
GAIN	REAL	Gain of the differentiation
LAG	TIME	Delay time constants
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization type "1" = Tracking mode "0" = Automatic mode
OUT	REAL	Output derivative unit with smoothing

Detailed description

Parametering

Parameter assignment for this function block is accomplished by selecting the GAIN of the derivative unit and the lag time constant LAG by which the output OUT will be delayed.
For very short scan times, after a unit step at the input IN (jump at input IN from 0 to 1.0), the output OUT will jump to the value of GAIN (theoretical value - in reality somewhat smaller due to the fact that the scan time is not infinitely short), to then return to 0 with the time constant LAG.

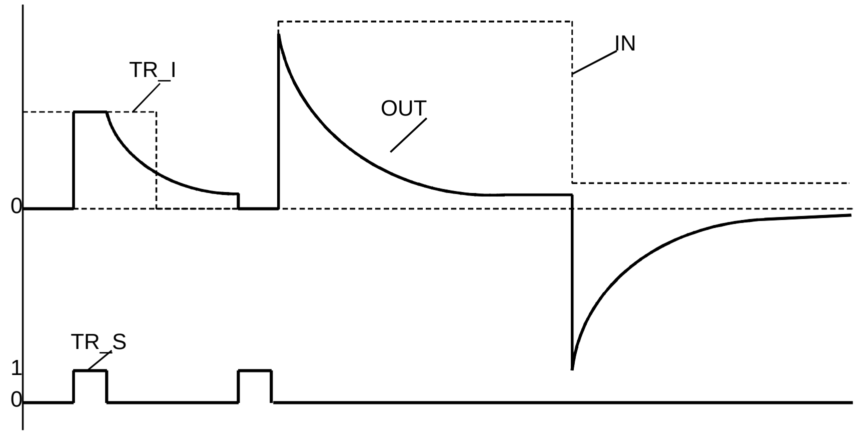
Operating mode

There are two operating mode selectable using the input TR_S:

Operating mode	TR_S	Meaning
Automatic	0	The function block operates as described in "Parametering".
Tracking	1	The tracking value TR_I is transferred directly to the output OUT.

Example

Representation of the LEAD function block jump response with GAIN = 1 and LAG = 10s:



LEAD_LAG: PD device with smoothing

22

Overview

At a glance

This chapter describes the LEAD_LAG block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block implements a PD element with following low-pass filter.
The function block has the following properties:
- Definable delay of the D-component
- Manual, halt and automatic modes

EN and ENO can be configured as additional parameters.

Formula

The transfer function is:

$$G(s) = gain \times \frac{1 + s \times lead}{1 + s \times lag}$$

The calculation formula is:

$$Y = \frac{lag \times Y_{(old)} + gain \times ((lead + dt) \times X - lead \times X_{(old)})}{lag + dt}$$

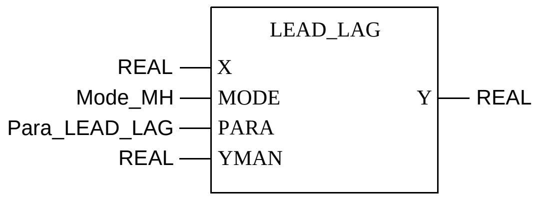
Meaning of the variables

Variable	Meaning
$X_{(old)}$	Value of input X from the previous cycle
$Y_{(old)}$	Value of output Y from the previous cycle
dt	Time difference between current and previous cycle

Representation

Symbol

Block representation



Parameter description LEAD_LAG

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input
MODE	Mode_MH	Operating mode
PARA	Para_LEAD_LAG	Parameter
YMAN	REAL	Manual value manipulated value
Y	REAL	Output

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual mode
halt	BOOL	"1" =Halt mode

Parameter description Para_LEAD_LAG

Data structure description

Element	Data type	Meaning
gain	REAL	Gain factor
lead	TIME	Derivative time constant
lag	TIME	Delay time constants

Detail description

Parametering

The parametering of the Function block is achieved through specification of the boost factor gain as well as the parametering of the Derivative time constant lead and the delayed time constants lag.
For very small sample times and the unit jump at input X (jump at input X from 0 to 1.0) output Y will jump to the value $\mathrm{gain} \times \mathrm{lead} / \mathrm{lag}$ (theoretical value - actual slightly smaller, due to the not infinitely small sample times), using the time constant lag to approximate the value $\mathrm{gain} \times 1.0$

Operating mode

There are three operating mode, which are selected via the elements man and halt:

Operating mode	man	halt	Meaning
Automatic	0	0	The Function block will be handled, as described in "Parametering".
Hand	1	0 or 1	The hand value YMAN will be transmitted permanently to the output Y.
Halt	0	1	The output Y will be set at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

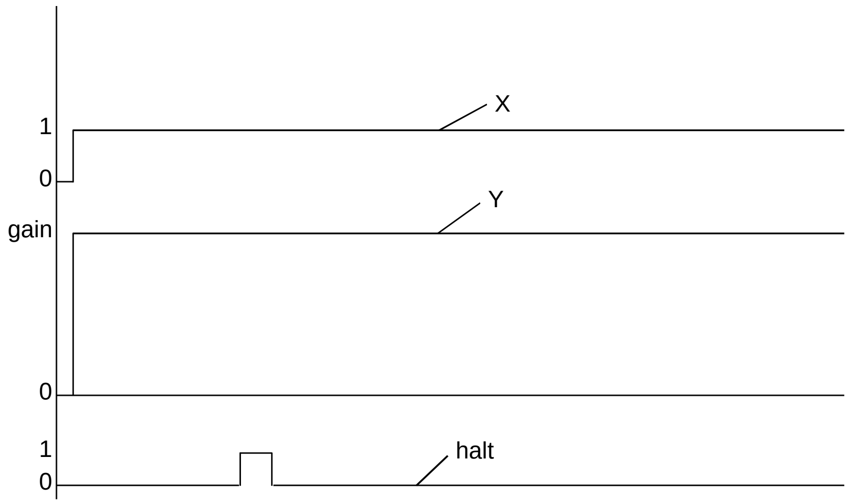
Examples of function blocks LEAD_LAG

Example-overview

The following examples are presented in the following diagrams:
- lead = lag
- lead=lag * 0.5, gain = 1
- lead/lag = 2, gain = 1

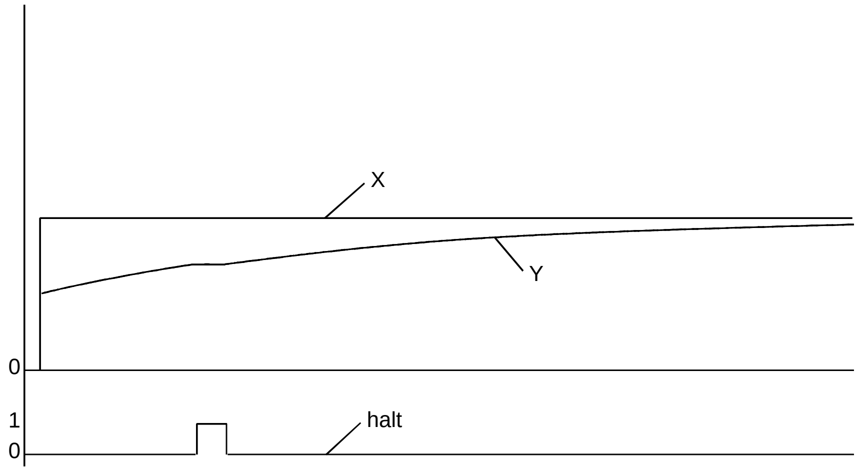
lead = lag

The function blocks behave like a pure multiplication block with the multiplier gain.
Function blockLEAD_LAG with lead = lag



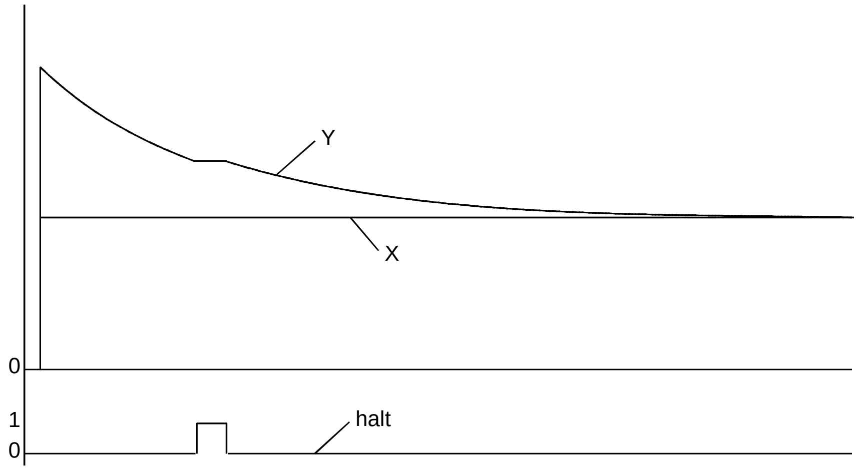
lead=lag * 0.5, gain = 1

The output Y jumps in this case to half the end value in order to run into the end value with the delayed time constant lag (gain * X)
Function block LEAD_LAG with lead/lag = 0.5 and gain = 1



lead/lag = 2, gain = 1

The output Y jumps in this case to double the end value in order to run into the end value with the delayed time constant lag (gain * X)
Function block LEAD_LAG with lead/lag = 2 and gain = 1



Runtime error

Error message

An Error message, appears when an invalid floating point number lies at input YMAN or X.

LEAD_LAG1: PD device with smoothing

23

Overview

At a glance

This chapter describes the LEAD_LAG1 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block serves as a PD outline with subsequent smoothing.
The function block contains the following properties:
- Definable delay of the D-component
- Operating mode, hand, halt, automatic

EN and ENO can be projected as additional parameters.

equation

The transmission function says:

$$G(s) = GAIN \times \frac{1 + s \times LEAD}{1 + s \times LAG}$$

The calculation equation says:

$$Y = \frac{LAG \times Y_{(old)} + GAIN \times ((LEAD + dt) \times X - LEAD \times X_{(old)})}{LAG + dt}$$

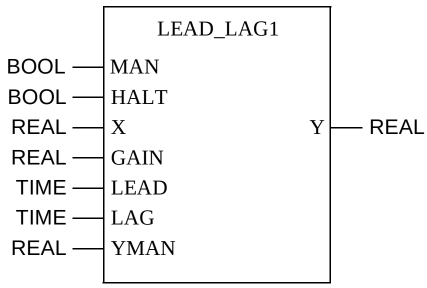
Meaning of the sizes

Size	Meaning
$X_{(old)}$	Value of output Y from the previous cycle
$Y_{(old)}$	Value of the input X from the previous cycle
dt	Time difference between current and previous cycle

Display

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
MAN	BOOL	"1" = Operating mode Hand
HALT	BOOL	"1" = Halt mode
X	REAL	Input
GAIN	REAL	Gain factor
LEAD	TIME	Derivative time constants
LAG	TIME	Delayed time constants
YMAN	REAL	Manual value-rank value
Y	REAL	Output

Detailed description

Parametering

The parametering of the Function block appears through specification of the boost factors GAIN as well as the parametering of the Derivative time constants LEAD and the delayed time constants LAG.
For very small sample times and the unit jump to input X (jump at line-in X from 0 to 1.0) output Y will jump to the value $\mathrm{GAIN} \times \mathrm{LEAD} / \mathrm{LAG}$ (theoretical value - actual slightly smaller due to the, not infinitely small sample times), using the time constant LAG to approximate the value $\mathrm{GAIN} \times 1.0$ closer.

Operating mode

There are three operating mode, which are selected via the elements MAN and HALT:

Operating mode	MAN	HALT	Meaning
Automatic	0	0	The Function block will be handled as "Parametering" describes.
Hand	1	0 or 1	The hand value YMAN will be transmitted fixed to the output Y.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

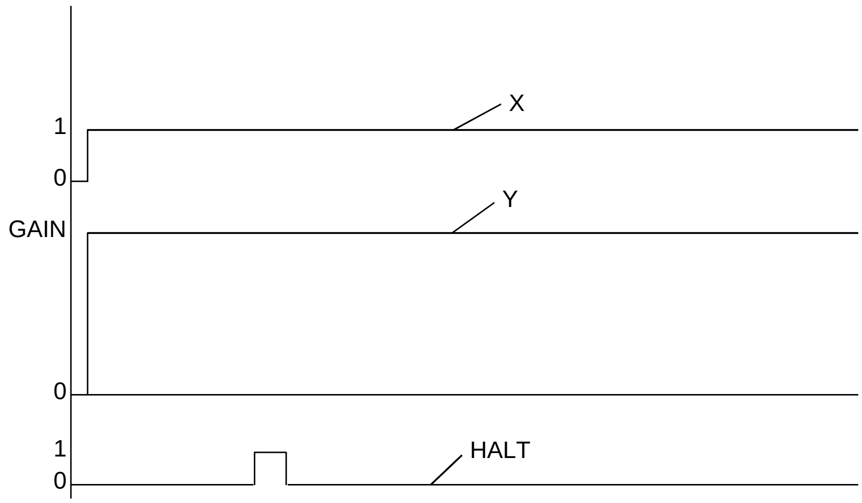
Examples of function blocks LEAD_LAG1

Example-overview

The following examples are presented in the following diagrams:
- LEAD = LAG
- LEAD=LAG * 0.5, GAIN = 1
- LEAD/LAG = 2, GAIN = 1

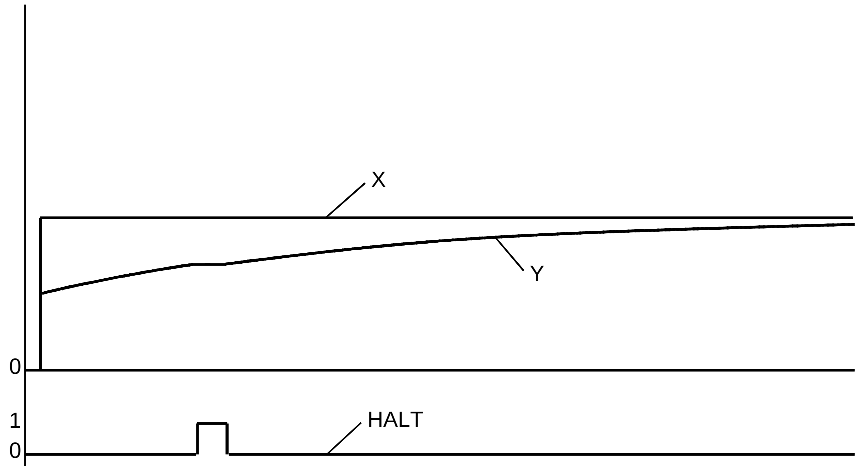
LEAD = LAG

The function block behaves like a pure multiplication block with the multiplier GAIN.
Function blockLEAD_LAG1 with LEAD = LAG



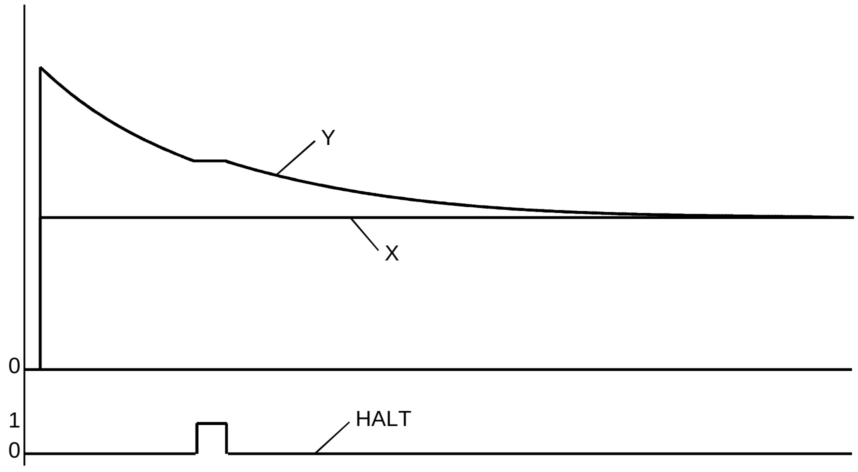
LEAD=LAG * 0.5, GAIN = 1

The output Y jumps in this case to half the end value in order to run into the end value with the delayed time constant lag (GAIN * X)
Function block LEAD_LAG1 with LEAD/LAG = 0.5 and GAIN = 1



LEAD/LAG = 2, GAIN = 1

The output Y jumps in this case to double the end value in order to run into the end value with the delayed time constant LAG (GAIN * X).
Function block LEAD_LAG1 with LEAD/LAG = 2 and GAIN = 1



LIMV: Velocity limiter: 1st order

24

Overview

At a glance

This chapter describes the LIMV block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block realizes a velocity limiter 1. Order with limiting of the manipulated variable.
The gradient of the input size X is limited to a specified value RATE. Further the output Y will be limited through YMAX and YMIN. This allows the function block to adjust signals to the technologically limited pace and limits from controlling elements.
EN and ENO can be projected as additional parameters.

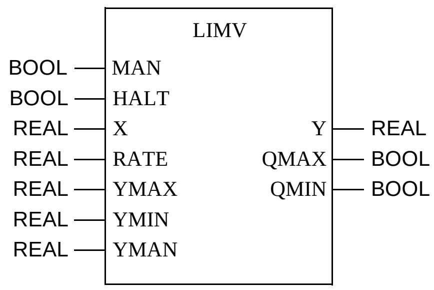
Properties

The function block contains the following properties:
- Operating mode, Hand, Halt, Automatic
- Manipulated variable limiting in automatic action

Display

Symbol

Block display



Parameter description

Block parameter description

Parameter	Data type	Meaning
MAN	BOOL	"1" = Operating mode Hand
HALT	BOOL	"1" = Halt mode
X	REAL	Input
RATE	REAL	Maximum upper limit (maximum x')
YMAX	REAL	Upper limit
YMIN	REAL	Lower limit
YMAN	REAL	Manual manipulated value
Y	REAL	Output
QMAX	BOOL	"1" = Output Y has reached upper limit
QMIN	BOOL	"1" = Output Y has reached lower limit

Detailed description

Parametering

The parametering of the function block appears through specification of the maximum upper speed RATE as well as the limits YMAX and YMIN for output Y. The maximum upper speed specifies to which value the output can change within one second.
The amount will be resolved from the parameter RATE. Ist RATE = 0, then Y = X.
The limits YMAX and YMIN limit the upper output as well as the lower output. So that means YMIN ≤ Y ≤ YMAX.
Reaching the bound value, i.e. a limit of the output signals will be shown at both the outputs, QMAX and QMIN:
- QMAX = 1 if Y ≥ YMAX
- QMIN = 1 if Y ≤ YMIN

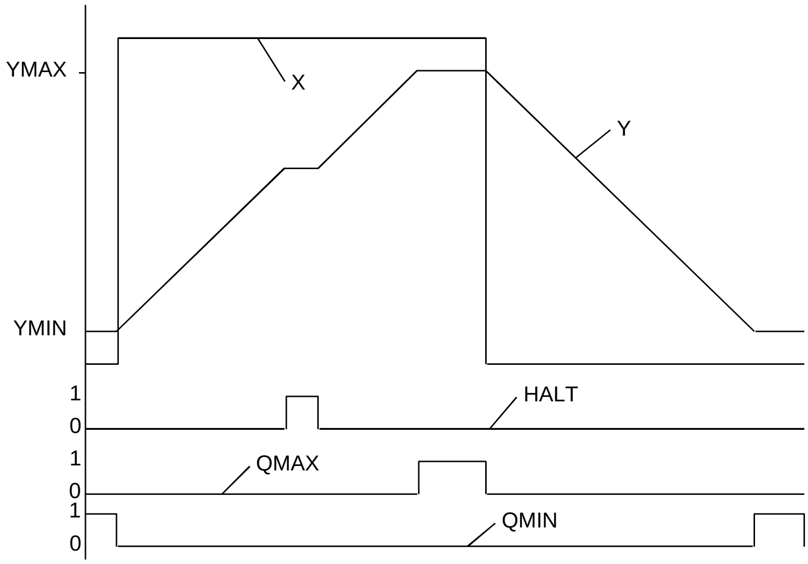
Operating mode

There are three operating mode, which are selected via the elements MAN and HALT:

Operating mode	MAN	HALT	Meaning
Automatic	0	0	The current value for Y will be constantly calculated and spent.
Hand	1	0 or 1	The manual value YMAN will be transmitted fixed to the output Y. The control output is, however, limited through YMAX and YMIN.
Halt	0	1	The output Y will be held at the last calculated value. The output will no longer be changed, but can be overwritten by the user.

Example

The function block follows the jump to input X with maximum change in speed. Output Y remains at a standstill in Halt mode, in order to subsequently move on from the rank at which it has stopped. It is also clear to see the limits of output Y through YMAX and YMIN with the relevant messages QMAX and QMIN.
Dynamic behavior of LIMV



Runtime error

Error message

With YMAN < YMIN an Error message appears

MFLOW: mass flow block

25

Overview

At a glance

This chapter describes the MFLOW block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block MFLOW calculates the mass flow of a gas in a throttle device due to the differential pressure and the temperature and pressure conditions of the gas.
The measure of the differential pressure can be replaced by the speed of the medium or with another measure with pressure and temperature compensation.
EN and ENO can be projected as additional parameters.

Equation

The full equation (i.e. with en_sqrt = 1, en_pres = 1 and en_temp =1) says as follows:

$$OUT = k \times \sqrt{\frac{IN \times PA}{TA}}$$

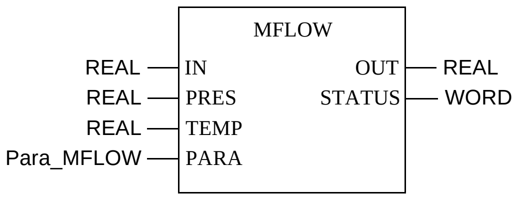
Meaning of the sizes

Size	Meaning
SV	Gas pressure in absolute units
TA	Absolute gas temperature in Kelvin

Representation

Symbol

Block representation



Parameter description MFLOW

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Input
PRES	REAL	Absolute or relative gas pressure
TEMP	REAL	Gas temperature printed out in °C or °F
PARA	Para_MFLOW	Parameter
OUT	REAL	Value of the mass flow, with temperature and pressure correction
STATUS	WORD	Status word

Parameter description Para_MFLOW

Data structure description

Element	Data type	Meaning
k	REAL	Calculating constants (see *Calculation of the constant k, p. 168*)
en_pres	BOOL	"1": Activate the pressure correction
pr_pa	BOOL	"1": PRES is an absolute pressure "0": PRES is a relative pressure
pu	REAL	Value, which in the used pressure unit 1 displays atmosphere
en_temp	BOOL	"1": Activate the temperature correction
tc_tf	BOOL	"1": TEMP will be printed out in Degree Fahrenheit "0": TEMP will be printed out in Degree Celsius
en_sqrt	BOOL	"1": Calculation with Square Root

Detailed description

Calculation of the constant k

The constant k can be calculated because of a work point reference, with which the mass flow (MF_REF), the differential pressure (IN_REF), the absolute pressure (P_REF) and the absolute temperature (T_REF) are recognized.
When the input IN is **a Differential pressure** the equation says as follows:

$$k = MF\_REF \times \sqrt{\frac{T\_REF}{P\_REF \times IN\_REF}}$$

When the input IN is **no Differential pressure** the equation says as follows:

$$k = MF\_REF$$

Specification of the calculation

With the calculation, a simple multiplication is entered: $OUT = k \times IN$. In order to achieve pressure or temperature compensation, the parameters en_pres or en_temp must be set to 1. The square route is also only active when en_sqrt = 1. When one of the parameters en_sqrt, en_pres, en_temp remains at 0, the calculation of the constant k must be adjusted to correspond (Delete the square route, replace from P_REF or T_REF through 1)

Temperature unit

The temperature TEMP can be printed out in Degree Celsius or Degree Fahrenheit, depending on the value of the parameter tc_tf :

tc_tf	Temperature unit from TEMP
0	Degree Celsius Calculation of the absolute temperature TA: $TA(°K) = TEMP + 273$
1	Degree Fahrenheit Calculation of the absolute temperature TA: $TA(°K) = \frac{5}{9} \times (TEMP - 32) + 273$

Pressure unit

The pressure PRES can be printed out in any unit, as absolute or relative pressure, according to the value of the parameter pr_pa.

pr_pa	Pressure unit from PRES
0	Relative pressure Parameter pu in the used unit 1 atmosphere, must conform Calculation of absolute pressure: PA = PRES + pu
1	Absolute pressure: PA = PRES

Runtime error

Status word

The bits of the status words have the following meaning:

Bit	Meaning
Bit 0 = 1	Error in a calculation in floating point values
Bit 1 = 1	Recording of an invalid value of a floating point value input
Bit 2 = 1	Division by zero with calculation in floating point values
Bit 3 = 1	Capacity overflow with calculation in floating point values
Bit 4 = 1	One of the following sizes is negative: IN, pu, PA, TA. For calculation, the function block uses the value 0.

Error message

In the following cases an Error will be recorded:
- At one of the floating point inputs an invalid value will be recorded
- Division by zero with calculation in floating point values
- Capacity overflow with calculation in floating point values

The output OUT will not be altered.

Warning

A warning is given if the parameter pu is negative, in this case with the calculation the block can use the value 0 in place of the defective value pu.

MS: Manual control of an output

26

Overview

At a glance

This chapter describes the MS block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

This Function block serves as the control of a numerical output, which can be switched off via the function block PWM1 (See *PWM1: Pulse width modulation, p. 341*) controlled analog output, server motor or controlling element. The control can appear via server dialog or direct via the SPS-Software.
In general a control-function block serves as the control of a digital output. The MS-block should then be used, if the control output should be uncoupled from the control of the analog output.
EN and ENO can be projected as additional parameters.

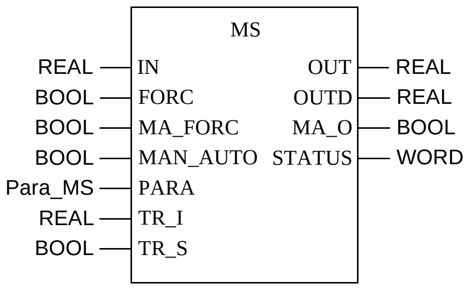
Application possibilities

The function block will mainly be used with the following applications:
- For the control of an analog output, which is not controlled via a servo loop (open loop).
- Servo loops, with which the control output and the user controlled output have inserted a processing operation.
- With scanning of the output controlled controller, if the scanning period exceeds 1 to 2 seconds.
- With control of a server motor: the function block MS is in this case the controller block in order to insert the server motor.

Representation

Symbol

Block representation



Parameter description MS

Block parameter description

Parameter	Data type	Meaning
IN	REAL	Manipulated variable used in automatic mode
FORC	BOOL	"1": The mode manual/automatic will be entered via MA_FORC "0": The mode manual/automatic will be entered via MAN_AUTO
MA_FORC	BOOL	Mode manual/automatic (if FORC = 1) "1": Automatic operating mode "0": Manual mode
MAN_AUTO	BOOL	Mode manual/automatic (if FORC = 0) "1": Automatic operating mode "0": Manual mode
PARA	Para_MS	Parameter
TR_I	REAL	Initialization input
TR_S	BOOL	Initialization command
OUT	REAL	Absolute output
OUTD	REAL	Incremental output: Difference between the present output and the output of the previous execution
MA_O	BOOL	Current mode of the function block (0: Manual, 1: Automatic)
STATUS	WORD	Status word

Parameter description Para_MS

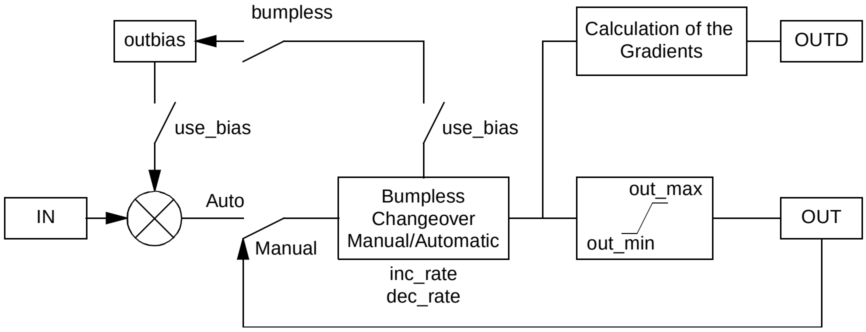
Data structure description

Element	Data type	Meaning
out_min	REAL	lower limit value of the output
out_max	REAL	upper limit value of the output
inc_rate	REAL	Increasing ramp at the changeover manual/ automatic (units per second)
dec_rate	REAL	Decreasing ramp at the changeover manual/ automatic (units per second)
outbias	REAL	Value of the bias
use_bias	BOOL	"1": Enable the bias
bumpless	BOOL	"1": Settings of the bias with changeover manual/ automatic (bumpless)

Detailed description

Structure diagram

In the following diagram the structure of the function block is displayed:



Setting of the mode selection

The mode selection can be set depending on input FORC either via the SPS program or via a server dialog (surveillance device):

Input FORC	Set the operating mode
0	Setting through the input MAN_AUTO (via operating device): MAN_AUTO= 1: Automatic mode MAN_AUTO= 0: Manual mode In this case the input MA_FORC is ineffective.
1	Setting through the input MA_FORC (via SPS-program): MA_FORC = 1: Automatic mode MA_FORC = 0: Manual mode In this case the input MAN_AUTO is ineffective.

The output MA_O always indicates the current operating mode of the function block.

Characteristics of the output OUT

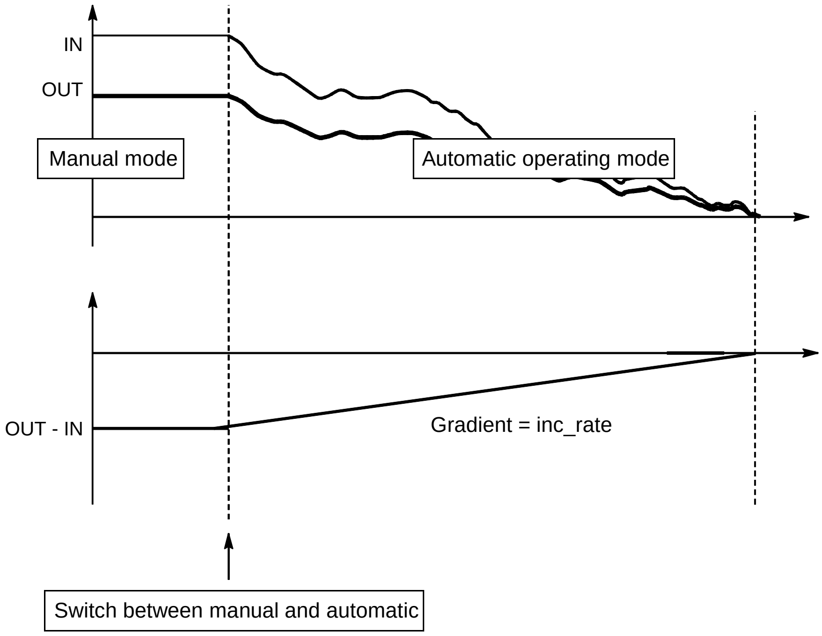
The following characteristics apply to the output OUT:
- Automatic mode: The output OUT is a copy of the input IN.
  In this operating mode, the output OUT can be assigned an OUTBIAS value (set _bias to 1). OUT calculates as follows: OUT = IN + outbias.
- Manual mode: The function block does not set the output, the server can directly change the value that is the connected variable at the output OUT.
- The output OUT is principally limited to an area between out_min and out_max. When the value calculated by the function block (or entered by the server in manual mode) exceeds one of these limit values, the value of OUT will be cut (to out_min or out_max). The incremental output OUTD on the other hand, never takes this cut into consideration.

Switch between manual and automatic

The switch manual/automatic at output appears bumpless, as the value of IN is not suddenly led to the output.
The output OUT gets closer to input IN ramps with positive (inc_rate) or negative increase (dec_rate):

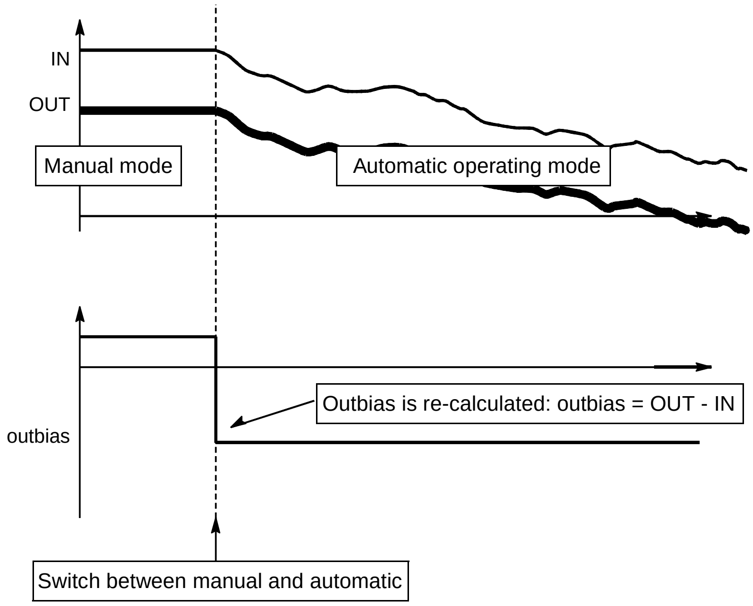
- inc_rate applies when IN is larger than OUT at the time of the changeover
- dec_rate applies when IN is smaller than OUT at the time of the changeover

bumpless changeover



The bumpless changeover can be annulled with the increasing ramp, when inc_rate is set to 0. Just as with dec_rate = 0 the changeover is with decreasing ramp with bumps. In both cases the input IN will travel immediately to output OUT when changed over to automatic mode.
When the parameter outbias (use_bias = 1) is used, a bumpless changeover manual/automatic can be achieved without change of the output, when the parameter is set to 1. In this case the parameter outbias will be recalculated by the block to compensate the difference between the input IN and the output OUT.

Bumpless changeover with the parameter Outbias

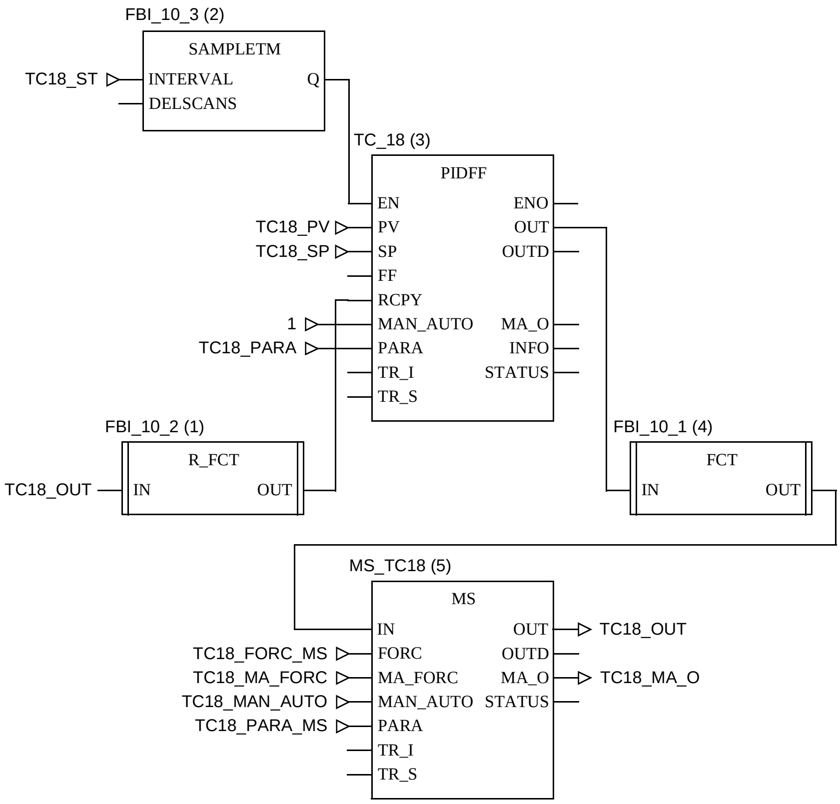


The bumpless changeover manual/automatic is advisable when the input of the function block is not connected to any controller or to a controller output without integral component.

Example

Example

In this example the output of the control block and the output controlled by the server will insert a processing operation (through the DFB FCT).
In order to guarantee a bumpless changeover between the modes manual/automatic, the reversed processing operation (R_FCT) will be assigned to the output of the MS function block and the result led back to the control input RCPY, which remained in automatic mode (MAN_AUTO = 1).
Display of the function plans



Runtime error

Status word

The bits of the status words have the following meaning:

Bit	Meaning
Bit 0 = 1	Error in a calculation in floating point values
Bit 1 = 1	Invalid value recorded at one of the floating point value inputs
Bit 2 = 1	Division by zero with calculation in floating point values
Bit 3 = 1	Capacity overflow with calculation in floating point values
Bit 4 = 1	The following error will be shown: ● One of the following sizes is negative: inc_rate, dec_rate. For calculation, the function block uses the value 0. ● The parameter Outbias lies out of the area $[(out\_min - out\_max), (out\_max - out\_min)]$. In this case the function block uses a cut value: $(out\_min - out\_max)$ and/or. $(out\_max - out\_min)$.
Bit 5 = 1	The output OUT has reached the lower limit value out_min (see Note)
Bit 6 = 1	The output OUT has reached the upper limit value out_max (see Note)

Note

Note: In manual mode these bits stay at 1 for only one program cycle. When the user enters a value for OUT which exceeds one of the limit values, the function block sets the Bit 5 or 6 to 1 and cuts them from the user entered value. With the following execution of the function block the value of OUT no longer lies outside the area and the Bits 5 and 6 are set to 0 again.

Error message

An error appears if a non floating point value is inputted or if there is a problem with a floating point calculation. In this case the outputs OUT, OUTD and MA_O remain unchanged.

Warning

In the following cases a warning is given:
- The parameter inc_rate is negative: in this case the function block uses the value 0 in place of the faulty value from inc_rate.
- The parameter dec_rate is negative: in this case the function block uses the value 0 in place of the faulty value from dec_rate.
- The parameter outbias lies outside the area [(out_min –out_max), (out_max – out_min)]. In this case for calculating the value the function block uses (out_min – out_max) and/or (out_max – out_min).

MULDIV_W: Multiplication/ Division

27

Overview

At a glance

This chapter describes the MULDIV_W block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block MULDIV_W carries out a weighted multiplication/division from 3 numerical input variables.
EN and ENO can be projected as an additional parameter.

Equation

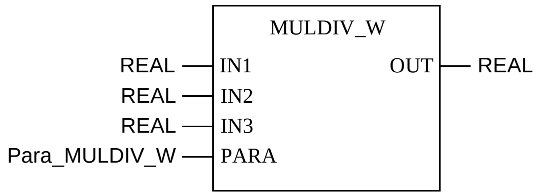
The equation says:

$$OUT = \frac{k \times (IN1 + c1) \times (IN2 + c2)}{IN3 + c3} + c4$$

Representation

Symbol

Block representation



Parameter description MULDIV_W

Block parameter description

Parameter	Data type	Meaning
IN1 to IN3	REAL	Numerical variables to be processed
PARA	Para_MULDIV_W	Parameter
OUT	REAL	Result of the calculation

Parameter description PARA_MULDIV_W

Data structure description

Element	Data type	Meaning
k, c1 to c4	REAL	Calculation coefficients

Runtime error

Error message

This error will be signaled if a non floating point value is inputted or if there is a problem with a floating point calculation. In general, the output OUT keeps its previous value, apart from with a division by 0, where the value corresponds to INF depending on which sign the counter uses.

PCON2: Two point controller

28

Overview

At a glance

This chapter describes the PCON2 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block forms a two-point controller, which maintains PID-similar behavior through two dynamic feedback paths.
EN and ENO can be projected as additional parameters.

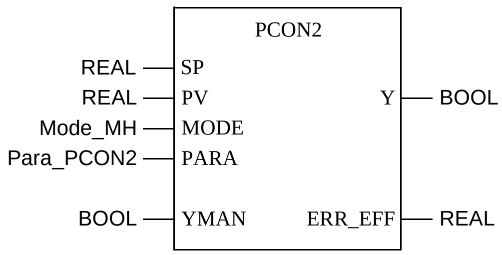
Properties

The function block contains the following properties:
- Operating mode, Manual, Halt, Automatic
- two internal feedback paths (delay 1. order)

Presentation

Symbol

Block display:



Parameter description PCON2

Block parameter description

Parameter	Data type	Meaning
SP	REAL	Setpoint input
PV	REAL	Process value input
MODE	Mode_MH	Operating mode
PARA	Para_PCON2	Parameter
YMAN	BOOL	"1" = Manual value for ERR_EFF
Y	BOOL	"1" = Output manipulated variable
ERR_EFF	REAL	Effective switch value

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual mode
halt	BOOL	"1" = Halt mode

Parameter description Para_PCON2

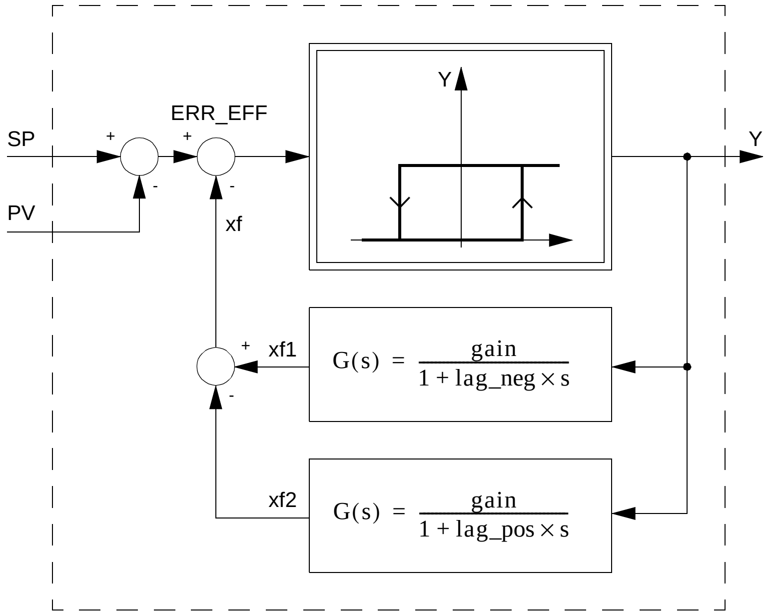
Data structure description

Element	Data type	Meaning
gain	REAL	Reset boost
lag_neg	TIME	Time constants of the quick reset
lag_pos	TIME	Time constants of the slow reset
hys	REAL	Hysteresis from two point switch
xf_man	REAL	Reset value of the reset in % (0 – 100)

Detailed description

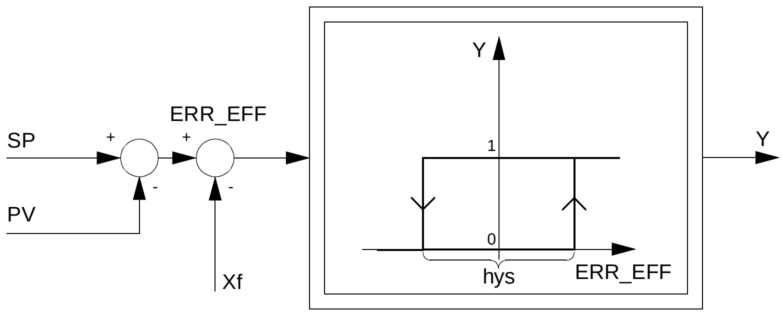
Structure of the controller

Structure of the two-point controller:



Principle of the two-point controller

The actual two-point controller will have 2 dynamic feedback paths (PT1-element) added. Through appropriate selection of the time constants of the reset-element, the two-point controller maintains dynamic behavior that corresponds to the behavior of a PID controller.



Reset

The revert- parameter set, made up of the revert boost gain and the revert time constant lag_neg and lag_pos, allows universal usage of the two point controller. The following table provides more exact information about it:

Revert	lag_neg	lag_pos
2-Point-Behavior (without revert)	= 0	= 0
negative revert	> 0	= 0
negative + positive revert	> 0	> lag_neg
Warning, regeneration (neg. feedback with lag_pos)	= 0	> 0
Warning, regeneration (pos. Feedback disabled)	> lag_pos	> 0

Select revert-boost gain is greater than zero!
Enter xf_man (meaning 0% to 100%) values between 0 and 100!

Hysteresis

The parameter hys indicates the connector hysteresis, i.e. the value that the effective switch value ERR_EFF outgoing from control point hys/2 must be reduced by, before the output Y is reset to"0". The dependence of the output Y depending of the effective switch value ERR_EFF and the Parameter hys, becomes clear in the picture *Principle of the two-point controller, p. 184* The value of the hys parameter is typically set to 1% of the maximum control area [max. (SP – PV].

Operating mode

There are three operating mode, which are selected via the elements man and halt:

Operating mode	man	halt	Meaning
Automatic	0	0	The Function block will be handled as described above.
Manual	1	1 or 0	The output Y are set to the value YMAN. xfl and xf2 are calculated using the following formula: xf1 = xf_man * gain /100 xf2 = xf_man * gain /100
Halt	0	1	The output Y will be held at the last value. xf1 and xf2 are set to gain * Y.

Runtime error

Warning

In the following cases a Warning will be given:

Causes	Behavior of the controller
lag_neg = 0 and lag_pos > 0	The controller works as if it had only a negative feedback lag_pos.
lag_pos < lag_neg > 0	The controller works as if it had only a negative feedback with the time constant lag_neg.
xf_man < 0 or xf_man > 100	The controller works without internal feedback paths.

PCON3: Three point controller

29

Overview

At a glance

This chapter describes the PCON3 block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block forms a three-point controller, which maintains PID-similar behavior through two dynamic feedback paths.
EN and ENO can be projected as additional parameters.

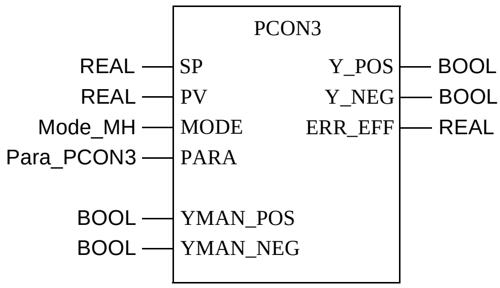
Properties

The function block PCON3 contains the following properties:
- Operating mode, Manual, Halt, Automatic
- two internal feedback paths (delay of the 1st order)

Presentation

Symbol

Block display:



Parameter description PCON3

Block parameter description

Parameter	Data type	Meaning
SP	REAL	Setpoint input
PV	REAL	Process value input
MODE	Mode_MH	Operating mode
PARA	Para_PCON3	Parameter
YMAN_POS	BOOL	Manual manipulation for Y_POS
YMAN_NEG	BOOL	Manual manipulation for Y_NEG
Y_POS	BOOL	"1" = positive manipulated variable at output ERR_EFF
Y_NEG	BOOL	"1" = negative manipulated variable at output ERR_EFF
ERR_EFF	REAL	Effective switch value

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1" = Manual mode
halt	BOOL	"1" = Halt mode

Parameter description Para_PCON3

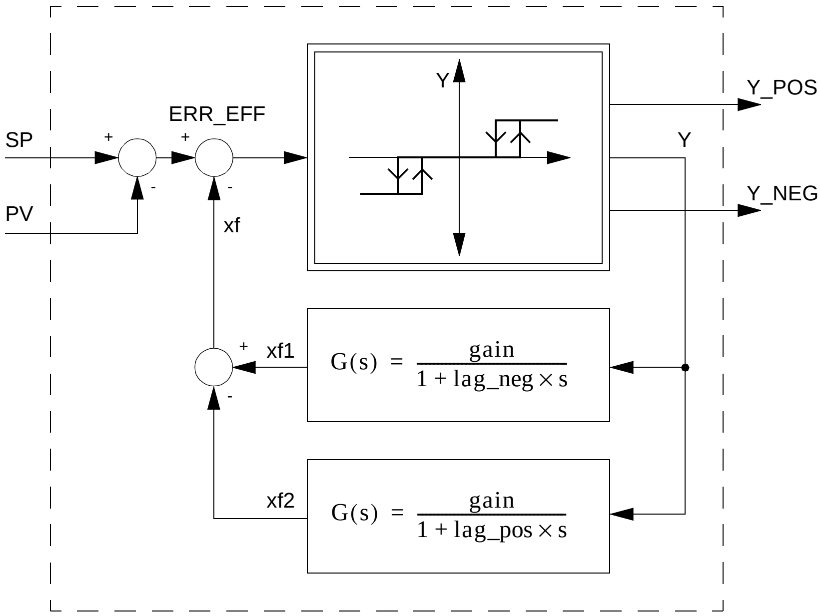
Data structure description

Element	Data type	Meaning
gain	REAL	Reset-boost (reset-parameter-sequence)
lag_neg	TIME	Time constant of the quick reset (reset-parameter-sequence)
lag_pos	TIME	Time constant of the slow reset (reset-parameter-sequence)
hys	REAL	Hysteresis from three point switch
db	REAL	Insensitivity zone
xf_man	REAL	Reset value of the reset in % (0 – 100)

Detail description

Structure of the controller

Structure of the three-point controller:

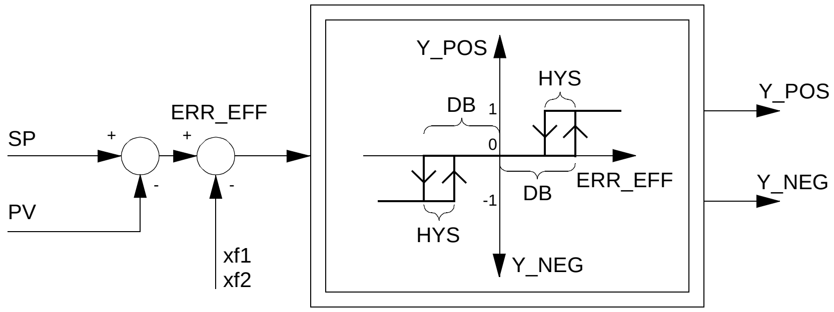


The following applies:

If…	Then…
Y = 1	Y_POS = 1 Y_NEG = 0
Y = 0	Y_POS = 0 Y_NEG = 0
Y = -1	Y_POS = 0 Y_NEG = 1

Principle of the three-point controller

The actual three-point controller will have 2 dynamic feedback paths (PT1-elements) added. Through appropriate selection of the time constants of the reset-element, the three-point controller maintains dynamic behavior that corresponds to the behavior of a PID controller.



Feedback

The function block has a parameter sequence for the internal feedback paths, comprised of the reset-boost gain and the reset time constant lag_neg and lag_pos. The following table provides more exact information about it:

Feedback	**lag_neg**	**lag_pos**
3-Point-Behavior (without revert)	= 0	= 0
negative revert	> 0	= 0
negative + positive revert	> 0	> lag_neg
Warning, regeneration (neg. feedback with lag_pos)	= 0	> 0
Warning, regeneration (pos. Feedback disabled)	> lag_pos	> 0

The parameter gain must be > 0
The amount will be resolved from the Hysterisis hys and the no-sensitivity zone db!
For xf_man (meaning -100 to 100%) values between -100 and 100 are to be entered!

No-sensitivity zone

The parameter db fixes the connection point for the outputs Y_POS and Y_NEG. If the effective switch value ERR_EFF is positive and is greater than db, then the output Y_POS will switch from "0" to "1". If the effective switch value ERR_EFF is negative and is smaller than db, then the output Y_NEG will switch from "0" to "1". The value of the db parameter is typically set to 1% of the maximum control area (max. SP – PV).

Hysteresis

The parameter hys indicates the connector-hysteresis, i.e. the value which the effective switch value ERR_EFF outgoing from control point db must be reduced by, before the output Y_POS (Y_NEG) is reset to "0". The connection between Y_POS and Y_NEG depending on effective switch value ERR_EFF and the parameters db and hys Is illustrated in the image *Principle of the three-point controller, p. 191*. The value of the hys parameter is typically set to 0.5% of the maximum control area (max. SP – PV).

Operating mode

There are three operating modes, which are selected via the elements man and halt:

Operating mode	man	halt	Meaning
Automatic	0	0	The Function block will be handled as described above.
Manual	1	0 or 1	The outputs Y_POS and Y_NEG are set to the values YMAN_POS and YMAN_NEG. In this case, the built in priority-logic – Y_NEG is dominant over Y_POS, which prohibits both outputs from being set simultaneously. xf1 and xf2 are calculated using the following formula: xf1 = xf_man * gain /100 xf2 = xf_man * gain /100
Halt	0	1	In Halt mode, both outputs Y_POS and Y_NEG will be held at the last value. xf1 and xf2 are set to gain * Y.

Runtime error

Error message

With hys > 2 * db, an Error Message appears.

Warning

In the following cases a Warning will be given:

Causes	Behavior of the controller
lag_neg = 0 and lag_pos > 0	The controller works as if it had only a negative feedback with the constant lag_pos.
lag_pos < lag_neg > 0	The controller works as if it had only a negative feedback with the time constant lag_neg.
xf_man < 0 or xf_man > 100	The controller works without internal feedback paths.

PD_or_PI: Structure changeover PD/PI controller

30

Overview

At a glance

This chapter describes the PD_or_PI block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Function description

The Function block PD_or_PI can work equally well as either PD-Controller or PI-Controller. Depending on the system deviation (SP – PV) and a specified switch value, trig_err will automatically perform a structural changeover from PD- to PI-Controller and vice-versa from PI- to PD-Controller.
This EFB is particularly suitable for starting control purposes. When the process is runup the controller reacts as a P(D) controller, whereby the controlled variable is to reach the adjusted reference variable value as fast as possible. Shortly before the given setpoint value is reached, the control algorithm is switched over and an I component makes sure that the remaining control deviation fades out.
EN and ENO can be projected as additional parameters.

Properties

The function block contains the following properties:
- PI controller with independent gain, ti-adjust
- PI controller with independent gain, td-adjust
- Limited manipulated variable in automatic mode
- Antiwindup reset in PI operation
- definable delay of the D-component
- Operating mode, Manual, Halt, Automatic
- smooth switch between manual and automatic
- Automatic bumpless changeover from PDPI operation

The PI controller transfer function

The PI controller transfer function is:

$$G(s) = gain\_i \times \left(1 + \frac{1}{ti \times s}\right)$$

The PD controller transfer function

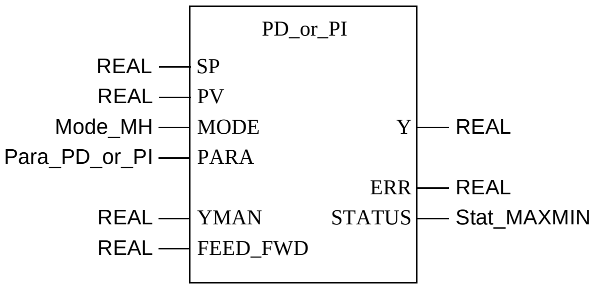
The PD controller transfer function is:

$$G(s) = gain\_d \times \left(1 + \frac{td \times s}{1 + td\_lag \times s}\right)$$

Presentation

Symbol

Block display:



Parameter description PD_or_PI

Block parameter description

Parameter	Data type	Meaning
SP	REAL	Setpoint input (reference variable)
PV	REAL	Process variable (controlled variable)
MODE	Mode_MH	Operating mode
PARA	Para_PD_or_PI	Parameter
YMAN	REAL	Manual manipulated variable
FEED_FWD	REAL	Disturbance variable
Y	REAL	Manipulated variable
ERR	REAL	System deviation
STATUS	Stat_MAXMIN	Output status

Parameter description Mode_MH

Data structure description

Element	Data type	Meaning
man	BOOL	"1": Manual mode
halt	BOOL	"1": Halt operating mode

Parameter description Para_PD_or_PI

Data structure description

Element	Data type	Meaning
trig_err	REAL	Changeover switching value for PDPI controller
gain_d	REAL	PD controller proportional action coefficient (gain)
td	TIME	PD controller rate time
td_lag	TIME	Delay of the PD controller rate time
gain_i	REAL	PI controller proportional action coefficient (gain)
ti	TIME	PI controller reset time
ymax	REAL	Upper limit
ymin	REAL	Lower limit

Parameter description Stat_MAXMIN

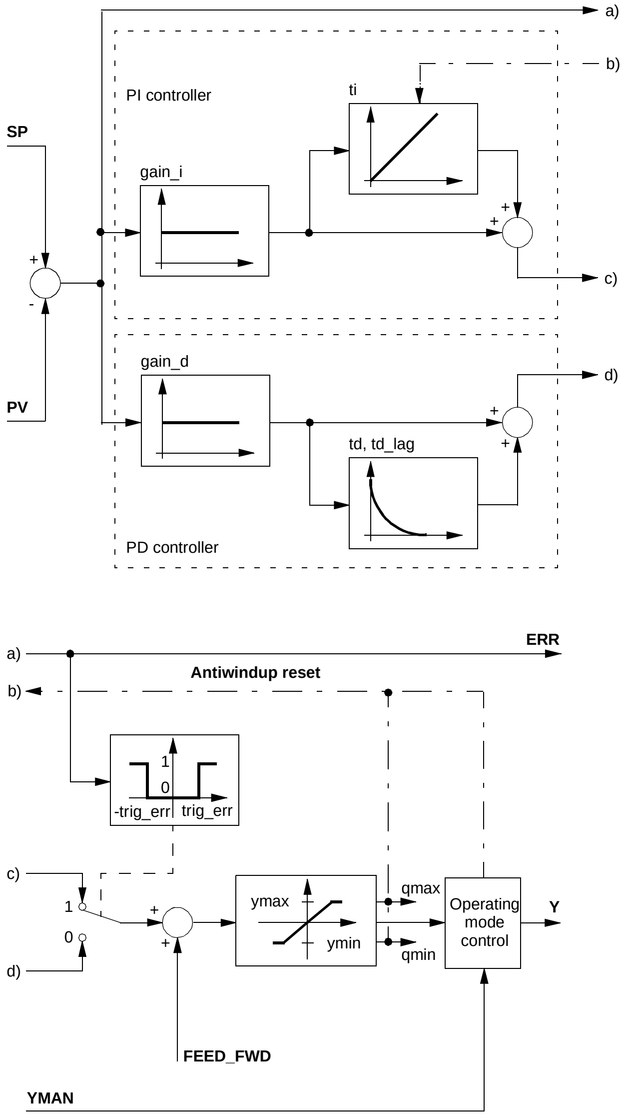
Data structure description

Element	Data type	Meaning
qmax	BOOL	"1" = Y reached upper limit
qmin	BOOL	"1" = Y reached lower limit

PD_or_PI function block structure diagram

Structure diagram

There follows now the structure diagram of the PD_or_PI block:



Detailed description

Determination of switching value

The parameterization of the function block begins with the determination of switching value trig_err. This parameter fixes the automatic changeover point of the function block from PDPI operation.
When the absolute value of system deviation ERR = SP - PV is smaller than the switching value trig_err, the controller switches automatically from PD operation into PI operation.
When the absolute value of system deviation is larger than the switching value trig_err, the controller switches automatically form PI operation into PD operation.
It then follows that:

- PD controller: ERR > trig_err
- PI controller: ERR ≤ trig_err

Each controller type is linked to a parameter set, which must be projected as well. The control algorithm changeover is practically a switch from one parameter set to the other. The changeover is bumpless.

PD controller

PD controller parameterization is accomplished by projection of the proportional action coefficient gain_d and rate time td.
For PD controller operation the D component is delayed by the time constant value td_lag. The td/td_lag ratio is termed the differential gain, and is generally selected between 3 and 10. The D component directly determined by the system deviation ERR, such that for reference variable fluctuations (variations at input SP) a jump attributed to the D component is produced.
The D component can be disabled by setting td = 0.

PI controller

PI controller parameterization is accomplished by projection of the proportional action coefficient gain_i and reset time ti.
In general during run-up with the PD algorithm, the proportional action coefficient is set considerably higher, than in the practically stationary operation in PI algorithm thereafter. This circumstance is conceded to by the designation of two independent proportional action coefficients.
The I component can be disabled by setting ti = 0.

Limiting of manipulated variable

The limits ymax and ymin retain the manipulated variable within the prescribed range.
It therefore holds that: ymin ≤ Y ≤ ymax
The outputs qmax and qmin signal that the manipulated variable has reached a limit, and thus been capped:

- QMAX = 1 if Y ≥ YMAX
- QMIN = 1 if Y ≤ YMIN

Upper limit ymax, limiting the manipulated variable, is to be set higher than lower limit ymin.

Antiwindup Reset

Should limiting of the manipulated variable take place while the PI control algorithm is active, the antiwindup reset should ensure that the I component "cannot go berserk". Antiwindup measures are taken only for I component values other than 0. Antiwindup limits are identical to those for the manipulated variable.
The antiwindup-reset measures correct the I component such that:
- YI ≥ ymin - gain_i * (SP - PV) - FEED_FWD
- YI ≤ ymax - gain_i * (SP - PV) - FEED_FWD

perating mode

There are three operating modes selectable via the man and halt parameter inputs:

Operating mode	man	halt	Meaning
Automatic	0	0	The manipulated variable output Y is determined through the discrete PI or PD closed-loop control algorithms, based on the controlled variable PV and reference variable SP. The manipulated variable is limited by ymax and ymin. The controller output limits also serve as limits for the antiwindup reset.
Manual	1	0 or 1	The manual manipulated value YMAN is passed on directly to the manipulated variable Y. The manipulated variable is limited by ymax and ymin. Internal variables will be so manipulated, that the controller changeover from manual to automatic can be bumpless.
Halt	0	1	The manipulated variable remains unchanged, the block does not influence the manipulated variable Y. Internal variables will be manipulated in such a manner that the controller can be driven smoothly from it's current position. Manipulated variable limits and antiwindup measures are as those in automatic mode Halt operating mode is also useful for allowing an external operator device to adjust the manipulated variable Y; the internal components in the controller are then tracked correctly.

Detailed formulas

Explanation of the formula sizes

Significance of the size in the following formulas:

Size	Meaning
dt	Present sample time
ERR	System deviation
$ERR_{(old)}$	System deviation value from the previous sampling step
FEED_FWD	Disturbance variable
Y	Current output (halt operating mode) or YMAN (manual mode)
YD	D component
$YD_{(old)}$	Value of the D-component from the previous sampling step
YI	I component
$YI_{(old)}$	Value of the I component from the previous sampling step
YP	P component

System deviation

The system deviation will be determined as follows:

$$ERR = SP - PV$$

Manipulated variable

The manipulated variable consists of different partial sizes which are dependent on the operating mode.

$$Y = YP + YI + YD + FEED\_FWD$$

After the summation of the components a manipulated variable limiting takes place at the output of the sub controller, which means:

$$ymin \le Y \le ymax$$

Overview to calculate the control components

Following this an overview on the different calculations of the control components in relation to the elements trig_err can be found:

Controller type	Controller components
PI-Controller (ERR ≤ trig_err)	YP and YD for manual, halt and automatic modes YI for automatic operating mode YI for manual and halt operating mode

Controller type	Controller components
PD-Controller (ERR > trig_err)	YP and YI for manual, halt and automatic modes YD for automatic mode YD for manual and halt operating mode

PI controller: YP and YD for all operating mode

YP and YD for manual, halt, automatic and cascade modes are located as follows:

$$YP = gain\_i \times ERR$$
$$YD = 0$$

PI controller: I component for automatic mode

YI for automatic mode is determined as follows (ti > 0):

$$YI = YI_{(old)} + gain\_i \times \frac{dt}{ti} \times \frac{ERR + ERR_{(old)}}{2}$$

The I-component is formed according to the trapezoid rule.

PI controller: I component YI for manual and halt modes

YI for manual and halt are located as follows

$$YI = Y - YP - FEED\_FWD$$

PD controller: YP and YI for all modes

YP and YI for manual, halt, and automatic modes are determined as follows

$$YP = gain\_d \times ERR$$
$$YI = 0$$

PD controller: D component for automatic mode

YD for automatic mode is determined as follows:

$$YD = \frac{YD_{(old)} \times td\_lag + td \times gain\_d \times (ERR - ERR_{(old)})}{dt + dt\_lag}$$

PD controller: D component for manual and halt operating mode

YD for manual, halt and automatic modes are determined as follows:

$$YD = 0$$

Runtime error

Error message

There is an Error message, if
- an unauthorized floating point number is placed at the input PV
- or ymax < is ymin

PDM: Pulse duration modulation

31

Overview

At a glance

This chapter describes the PDM block.

What's in this chapter?

This chapter contains the following topics:

Brief description

Using the block

Actuators are driven not only by analog quantities, but also through binary actuating signals. The conversion of analog values into binary output signals is achieved for example, through pulse width modulation (PWM) or pulse duration modulation (PDM).
The actuator adjusted average energy (actuator energy) should be in accord with the modulation block's analog input value (X).

Function description

The Function block PDM is used to convert analog values into digital output signals. In the function block PD a "1" signal of fixed persistence is output within a variable cycle time proportional to the analog value X. The adjusted average energy corresponds to the quotient of the fixed duty cycle t_on and the variable cycle period. In order that the adjusted average energy also corresponds to the analog input variable X, the following must apply:

$$T_{period} \sim \frac{1}{X}$$

EN and ENO can be configured as additional parameters.

General information about the actuator drive

In general, the binary actuator drive is performed by two Boolean signals Y_POS and Y_NEG. On a motor the output Y_POS corresponds to the signal "clockwise rotation" and the output Y_NEG the signal "counter-clockwise rotation". For an oven the outputs Y_POS and Y_NEG could be interpreted as corresponding to "heating" and "cooling".
Should the actuating drive in question be a motor, it is possible that to avoid overtravel for non-self-locking gearboxes, a brake pulse must be output after the engage signal.
In order to protect the power electronics, there must be a pause time t_pause after switching on t_on and before the brake pulse t_brake so as to avoid short circuits.

Formula

For correct operation the following rules should be observed:

$$t\_on + 2 \times t\_pause + t\_brake \geq \frac{pos\_}{neg\_} \times t\_min$$

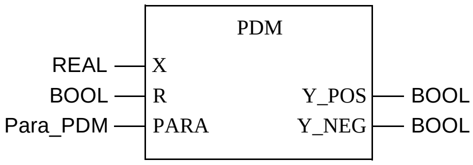
and

$$\frac{pos\_}{neg\_} \times t\_min < \frac{pos\_}{neg\_} \times t\_max$$

Representation

Symbol

Block representation



Parameter description PDM

Block parameter description

Parameter	Data type	Meaning
X	REAL	Input variable
R	BOOL	Reset mode
PARA	Para_PDM	Parameter
Y_POS	BOOL	Positive X value output
Y_NEG	BOOL	Negative X value output

Parameter description Para_PDM

Data structure description

Element	Data type	Meaning
t_on	TIME	Pulse duration (in s)
t_pause	TIME	Pause time (in s)
t_brake	TIME	Braking time (in s)
pos_up_x	REAL	Upper limit for positive X
pos_t_min	TIME	Minimum cycle time for Y_POS (where x = pos_up_x) (in s)
pos_lo_x	REAL	Lower limit for positive X
pos_t_max	TIME	Maximum cycle time for Y_POS (where x = pos_lo_x) (in s)
neg_up_x	REAL	Upper limit for negative X
neg_t_min	TIME	Minimum cycle time for Y_NEG (where x = -neg_up_x) (in s)
neg_lo_x	REAL	Lower limit for negative X
neg_t_max	TIME	Maximum cycle time for Y_NEG (where x = -neg_lo_x) (in s)

Detailed description

Block mode of operation

The pulse duration t_on determines the time span in which the output Y_POS resp. Y_NEG has "1" signal. For a positive input signal X the output Y_POS is set, on negative the output Y_NEG is set. Only one output can carry "1" signal. It is advisable to perform a freely definable pause time of t_pause = 10 or 20 ms between the actuating and brake pulses to protect the power electronics (hopefully preventing simultaneous firing of the antiparallel connected thyristors).
A possible brake pulse of duration time t_brake follows the output pulse duration after a pause time t_pause. Within the pause time both outputs carry "0" signal. During the braking time the output opposite that carrying the previous pulse goes to "1" signal. A pause time of t_pause = 20 ms (t_pause =0.02) corresponds to an interruption of the firing angle control for two half waves. That should guarantee a sufficiently large safety margin for the prevention of short-circuits resp. triggering of the suppressor circuitry as a consequence of antiparallel thyristors firing.
Thereafter follows a period in which both outputs carry "0" signal (wait timeout).

Period t_{period}

This wait timeout, together with the pulse, pause and brake times, all makeup a period t_{period} , which depending on lo_x and t_min, is calculated according to the following formulas:

Requirements	Equation	Explanation of formula variables
lo_x <> 0	$t_{period} = t_0 + \frac{K}{X}$	$K = (t\_max - t\_min) \times \frac{up\_x \times lo\_x}{up\_x - lo\_x}$ $t_0 = t\_max - \frac{K}{lo\_x}$
lo_x = 0 t_min > 0	$t_{period} = \frac{K}{X - X0}$	$X0 = \frac{t\_max \times lo\_x - t\_min \times up\_x}{t\_max - t\_min}$ $= t\_min \times (up\_x - X0)$
lo_x = 0 t_min = 0	$t_{period} = t\_max \times \left(1 - \frac{X}{up\_x}\right)$	

The following holds for all three cases:

Assuming	lo_x	up_x	t_min	t_max
$X \geq \lvert pos\_lo\_x \rvert$	pos_lo_x	pos_up_x	pos_t_min	pos_t_max
$X \geq -\lvert neg\_lo\_x \rvert$	neg_lo_x	neg_up_x	neg_t_min	neg_t_max

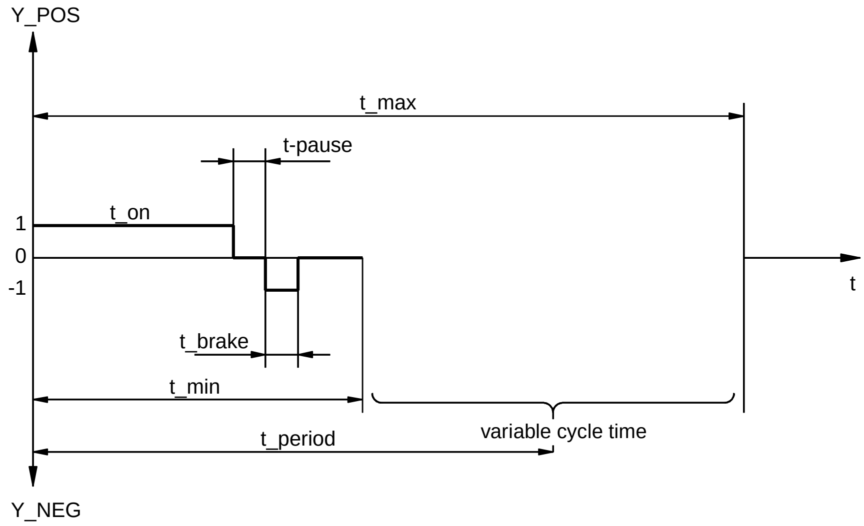
Note: From the parameters up_x (-pos/-neg) and lo_x (-pos/-neg) only the (absolute) value is evaluated.

Cycle time

The parameter t_min _ for every output there is a separate value _ gives the minimum period, i.e. the time span, which passes from the beginning of one actuating pulse until the start of the next. This time span appears when input X goes beyond value up_x _ this time there is a separate value for each sign.
The parameter t_max places an upper limit on the maximum period. Should the input cross below the value pos_lo_x or neg_lo_x, the actuating pulse output is terminated until the until the input exceeds the value pos_lo_x or neg_lo_x again. The values pos_lo_x and neg_lo_x define what is in principle a dead zone, in which the function block outputs are not activated.
The parameters (pos_t_min, pos_up_x) and (pos_t_max, pos_lo_x) apply for positive X input signals, whereby output Y_POS is set. In the same way the parameters (neg_t_min, neg_up_x) and (neg_t_max, neg_lo_x) are valid for negative X input signals. Output Y_NEG is set.

Time ratios display

An overview of the ratios between times is shown in the following diagram:

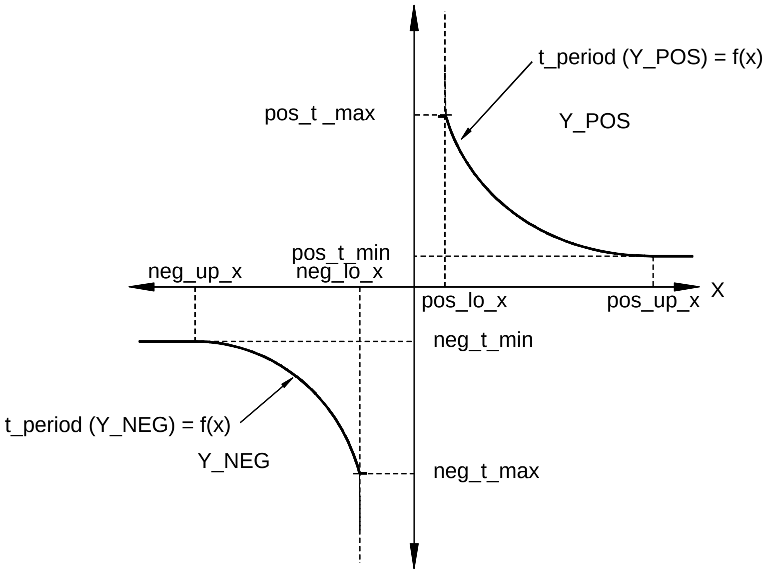


Time-span dependency

The time-span dependency from the input variable X, in which the output Y_POS (Y_NEG) carries 1-Signal, is displayed in the picture "*Output dependency on X, p. 209*" and the picture "*Output dependency on X (Special case), p. 209*".

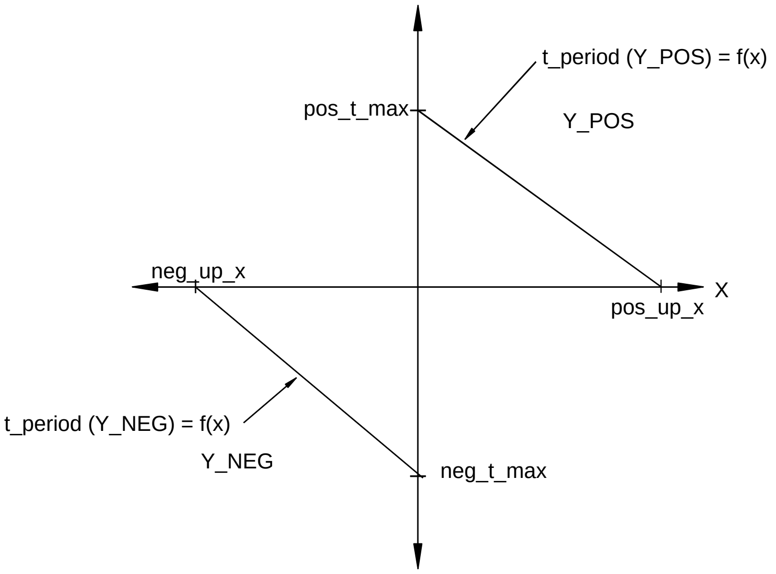
Output dependency on X

In the following picture the dependency of the output on X is shown:



Output dependency on X (Special case)

In the following picture the special case t_min = 0, lo_x = 0 is shown:



Operating mode

In reset mode R = "1", outputs Y_POS and Y_NEG are set to "0" signal. The internal time meters are also standardized, so that the function block begins the transfer to R=0 with the output of a new 1 signal on the associated output.

Boundary conditions

If the PDM function block is operated together with a PID controller, then the maximum period t_max should be so selected, that it corresponds to the PID controller's scan time. It is then guaranteed that every new actuating signal from the PID controller within the period time can be fully processed.
The PDM scan time should be in proportion with period vs. pulse time, Though this, the smallest possible actuating pulse is be determined.

The following ratio is recommended:
t_max/scan time (PDM) $\geq$ 10

Runtime error

Error message

An Error message appears, if
- $|up\_x| \leq |lo\_x|$
- $t\_max \leq t\_min$

Index

D

Q

R

S

T

V