Function block library

ControlTechnology_3

for PLCnext Engineer

Documentation for
PHOENIX CONTACT function blocks
PHOENIX CONTACT GmbH Co. KG
Flachsmarktstrasse 8
D-32825 Blomberg, Germany

This documentation is available in English only.

Table of Contents

- 1 Installation hint
- 2 General information
  - 2.1 Function blocks with “CYCLE” input
  - 2.2 Non-wiring of inputs and outputs
  - 2.3 Sampling times
  - 2.4 Ranges of values
- 3 Change notes
- 4 Function blocks
- 5 Scaling blocks
  - 5.1 SC_W_R
  - 5.2 SC_R_W
  - 5.3 SC_R_R
- 6 Function blocks for processing of analog values
  - 6.1 LTR
  - 6.2 A2_OF_A3
  - 6.3 LIMITVAL
  - 6.4 LIMITROC
  - 6.5 ALARM_2Q
  - 6.6 ALARM_4Q
- 7 Function blocks for processing of binary values
  - 7.1 B2_OF_B3
- 8 Function blocks for driving actuators
  - 8.1 TWIN_DRIVE
  - 8.2 THREE
  - 8.3 REV_LOCK
- 9 Function blocks for the non-linear signal processing
  - 9.1 POLG_N
  - 9.2 POLN_N
  - 9.3 DEADBAND
- 10 Function blocks for dynamic signal processing
  - 10.1 INT_C
  - 10.2 LAG1ST
  - 10.3 DYN
  - 10.4 HOLD
- 11 Controller blocks
  - 11.1 C_N
  - 11.2 PID_C
  - 11.3 PID_R
  - 11.4 THREE_C
  - 11.5 PID_ADA
  - 11.6 PID_STR

- 11.7 PID_MAN
- 11.8 PID_MODE
- 11.9 PID_PAR
- 11.10 PID_STAT
- 11.11 Startup information

- 12 Function blocks for parameterizing and special functions
  - 12.1 DRIVE_SIM
  - 12.2 SEND50
  - 12.3 RECV50
  - 12.4 ADA_PAR
  - 12.5 AG_PAR
  - 12.6 C_N_PAR
  - 12.7 MODE_PAR
  - 12.8 PAR_PAR
  - 12.9 POL_PAR

- 13 Appendix
  - 13.1 Conventions for identifiers
  - 13.2 User defined variables
  - 13.3 Data types

- 14 Support

1 Installation hint

If you did not specify a different directory during **library** installation all data in the MSI file will be unpacked to

c:\Users\Public\Documents\Phoenix Contact Libraries\PLCnext Engineer (former: PC Worx Engineer)

Please copy the library data to your PLCnext Engineer (former: PC Worx Engineer) working library directory.

If you did not specify a different directory during **PLCnext Engineer** installation the default PLCnext Engineer working library directory is

c:\Users\Public\Documents\PLCnext Engineer\Libraries (former: PC Worx Engineer\Libraries)

2 General information

Using this function block library a large number of automatic control engineering jobs can be realized. In the past, compact controllers, for example, were used for control tasks in many areas of industry. With the increasing use of controllers, even for smaller control tasks, there is often the desire to integrate the functionalities of these compact controllers into one controller. This function block library takes this into account through its universal configurability.

The following are the most important criteria considered during the development process:

1. Realization of all important basic control engineering functions,
2. scaleable functional scope due to modularization,
3. simplification by “omitting”, i.e. through non-wiring of inputs and outputs.
4. implemented functions to avoid operating and parameter setting errors.

2.1 Function blocks with “CYCLE” input

All function blocks with a CYCLE input (sampling time or cycle time) must not be executed in the default task. The cycle time of the default task changes permanently according to the utilization of the PLC. Therefore it is necessary to execute the corresponding function blocks in a “CYCLE”-task.

2.2 Non-wiring of inputs and outputs

With the exception of the inputs ENABLE and CYCLE it is not required to wire all inputs and outputs. This way program parts and functions can be tested and brought into operation step by step.

2.3 Sampling times

The type of the generated actuating signal is considerably influenced by the selection of the sampling times. Too large sampling times cause a strongly discontinuous mode of operation of the controller blocks or the dynamic signal processing blocks, respectively. However, this can also be desirable under certain circumstances. Very short sampling times (shorter than 5 ms) are not sensible with respect to the speed of the connected INTERBUS. In order not to load the PLC unnecessarily the sampling times should be selected just as short as required.

In practice often an approximation is used. Here, the sampling time CYCLE is set in a way, that it approximately corresponds to maximally one tenth of the equivalent time constant T(ETC).

CYCLE <= 1/ 10 * T(ETC)

As a result the behaviour of the implemented software controller corresponds to a conventional analog controller. T(ETC) (alternative time constant) is determined by evaluating the initial transient of the controlled system after the input of a set point step.

2.4 Ranges of values

It is recommended to use generally only per cent quantities for the input quantities of the function blocks PID_C or PID_R. All internal presettingsare based on a range of values between 0 and 100 per cent. As a result the process quantities can be directly read in per cent. For the conversion into other ranges of values or into the corresponding peripheral formats, the scaling blocks SC_R_R, SC_W_R or SC_R_W are available.

The user can also apply own ranges of values. Doing so, however, it must be ensured that possible set point and actual value limitations must be parameterized.

3 Change notes

Library version	Library build	PLCnext Engineer version	Change notes	Supported PLCs
3	20200206	>= 2020.0 LTS	Released for 2020.0 LTS	AXC F 1152 (1151412) AXC F 2152 (2404267)
3	20191001	2019.0 LTS 2019.3 2019.6 2019.9	Adapted to 2019.9	AXC F 2152 (2404267)
2	20190913	2019.0 LTS 2019.3 2019.6	Adapted to 2019.6	AXC F 2152 (2404267)
1	20190604	2019.0 LTS	Converted from PC Worx 6	AXC F 2152 (2404267)

4 Function blocks

Function block	Description	Version	Supported articles	License
SC_W_R	Scaling of the analog input value and type conversion from WORD to REAL.	1	-	none
SC_R_W	Scaling of the analog output value and type conversion from REAL to WORD.	1	-	none
SC_R_R	Scaling of any quanity of the type REAL.	1	-	none
LTR	Linear transformation.	1	-	none
A2_OF_A3	Analog value selection 2 out of 3.	1	-	none
LIMITVAL	Amplitude limiter.	1	-	none
LIMITROC	Rate of change limiter.	1	-	none
ALARM_2Q	Limit value indicator with 2 alarm limits.	1	-	none
ALARM_4Q	Limit value indicator with 2 warning and 2 alarm limits.	1	-	none
B2_OF_B3	Binary value selection 2 out of 3.	1	-	none
TWIN_DRIVE	Simultaneous driving of two actuators.	1	-	none
THREE	Three-position control element (two-position control element).	1	-	none
REV_LOCK	Reversing interlock.	1	-	none
POLG_N	Polygonal line.	1	-	none
POLN_N	Polynomial.	1	-	none
DEADBAND	Dead band (without hysteresis).	1	-	none
INT_C	Integrator.	1	-	none
LAG1ST	PT1 element / PT1 filter.	1	-	none
DYN	Dynamic element.	1	-	none
HOLD	Holding element.	1	-	none
C_N	Keying controller of the n-th order.	1	-	none
PID_C	Continuous PID-type controller.	1	-	none
PID_R	Continuous PID-type controller (with reduced performance range).	1	-	none
THREE_C	Three-position controller attachment for PID_C and PID_R.	1	-	none
PID_ADA	PID-type controller attachment module for controlled adaption.	1	-	none
PID_STR	PID-type controller attachment module for controlled adaption.	1	-	none
PID_MAN	PID-type controller attachment module for the input of manual manipulated values.	1	-	none
PID_MODE	PID-type controller attachment module for the input of the MODE control commands.	1	-	none
PID_PAR	PID-type controller attachment module for the input of parameterizing values.	1	-	none
PID_STAT	PID-type controller attachment module for the output of status information.	1	-	none
DRIVE_SIM	Simulation of a servo motor.	1	-	none
SEND50	Storage of 50 successive values.	1	-	none
RCV50	Display of the 50 determined values in FBD.	1	-	none

ADA_PAR	Clear parameter transfer in FBD.	1	-	none
AG_PAR	Clear parameter transfer in FBD.	1	-	none
C_N_PAR	Simplified parameter setting for the function block C_N.	1	-	none
POL_PAR	Simplified parameter setting for the function block POLN_N.	1	-	none
MODE_PAR	Simplified parameter setting for the function block PID_MODE.	1	-	none
PAR_PAR	Simplified parameter setting for the function block PID_PAR.	1	-	none

5 Scaling blocks

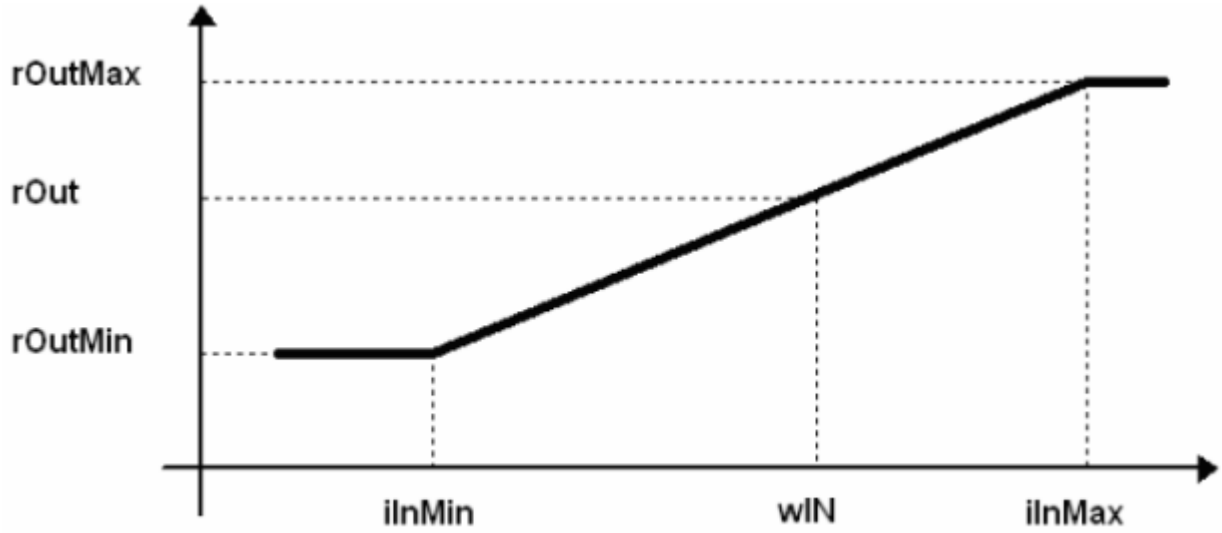
The following scaling blocks are available:

Function block	Description	Version	Supported articles	License
SC_W_R	Scaling of the analog input value and type conversion from WORD to REAL.	1	-	none
SC_R_W	Scaling of the analog output value and type conversion from REAL to WORD.	1	-	none
SC_R_R	Scaling of any quanity of the type REAL.	1	-	none

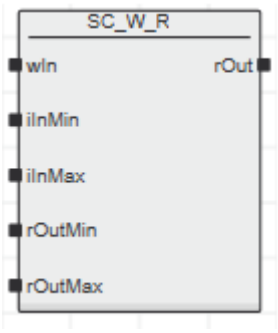
5.1 SC_W_R

This function block is used for scaling analog input values. For this purpose the peripheral values of an analog-to-digital converter are converted from the data format WORD to the floating point format REAL. If the values applied to the input are greater than IN_MX or lower than IN_MN, they are limited to the set maximum and minimum values.

Note: The entered limit values are not monitored.



5.1.1 Function block call



5.1.2 Input parameters

Name	Type	Description
wIn	WORD	Input value (e.g. from analog-to-digital-converter).
iInMin	INT	Input value limitation (minimum value).
iInMax	INT	Input value limitation (maximum value).
rOutMin	REAL	Output value limitation (minimum value).
rOutMax	REAL	Output value limitation (maximum value).

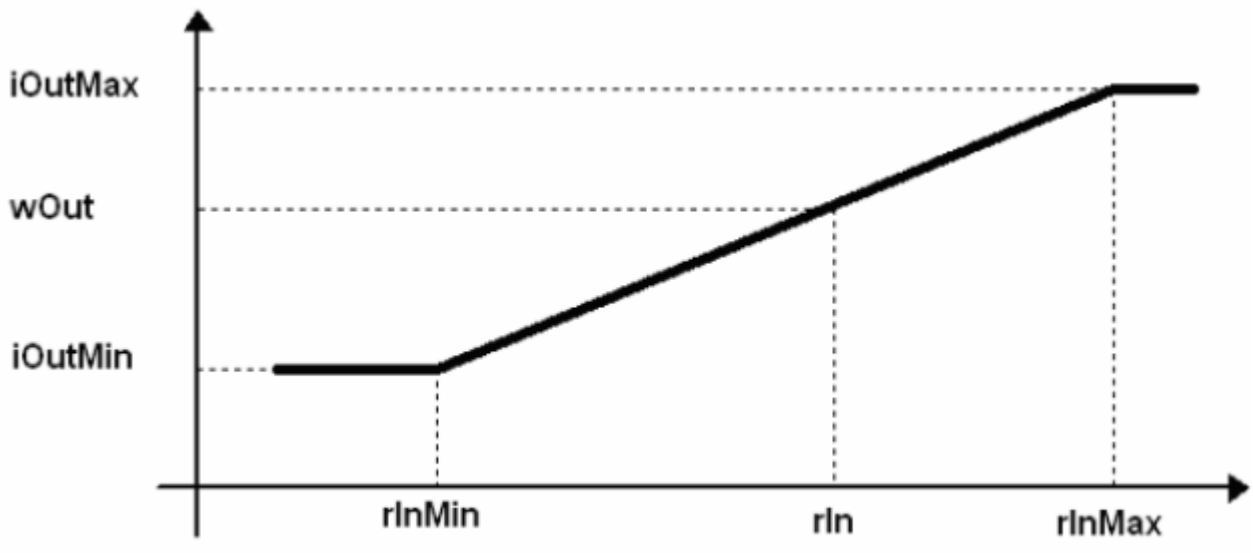
5.1.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

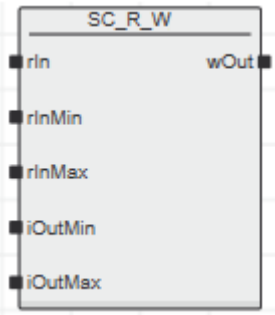
5.2 SC_R_W

This function block is used for scaling analog output values. For this purpose values of the type REAL are converted to the data format WORD of the digital-to-analog converter peripherals.

Note: The entered limit values are not monitored.



5.2.1 Function block call



5.2.2 Input parameters

Name	Type	Description
rIn	REAL	Input value.
rInMin	REAL	Input value limitation (minimum value).
rInMax	REAL	Input value limitation (maximum value).
iOutMin	INT	Output value limitation (minimum value).
iOutMax	INT	Output value limitation (maximum value).

5.2.3 Output parameters

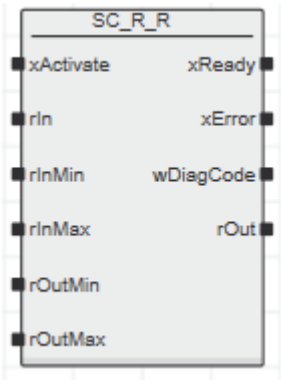
Name	Type	Description
wOut	WORD	Output value (e.g. from digital-to-analog-converter).

5.3 SC_R_R

This function block is used for scaling any quantities of the type REAL. The field of application for this function block is the conversion between different ranges of representation.

Note: The entered limit values are not monitored.

5.3.1 Function block call



5.3.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rIn	REAL	Input value.
rInMin	REAL	Input value limitation (minimum value).
rInMax	REAL	Input value limitation (maximum value).
rOutMin	REAL	Output value limitation (minimum value).
rOutMax	REAL	Output value limitation (maximum value).

5.3.3 Output parameters

Name	Type	Description
xReady	BOOL	FALSE: The function block is executing services. TRUE: The function block is ready to execute services.
xError	BOOL	TRUE: An error has occurred. For details refer to wDiagCode and wAddDiagCode.
wDiagCode	WORD	Diagnosis code. Refer to diagnostic table.
rOut	REAL	Output value.

5.3.4 Diagnosis

DiagCode	Description
16#0000	Function block is deactivated
16#8000	Function block is in regular operation
16#C101	rOutMax <= rOutMin
16#C103	rIn does not fit within the specified rInMin and rInMax

6 Function blocks for processing of analog values

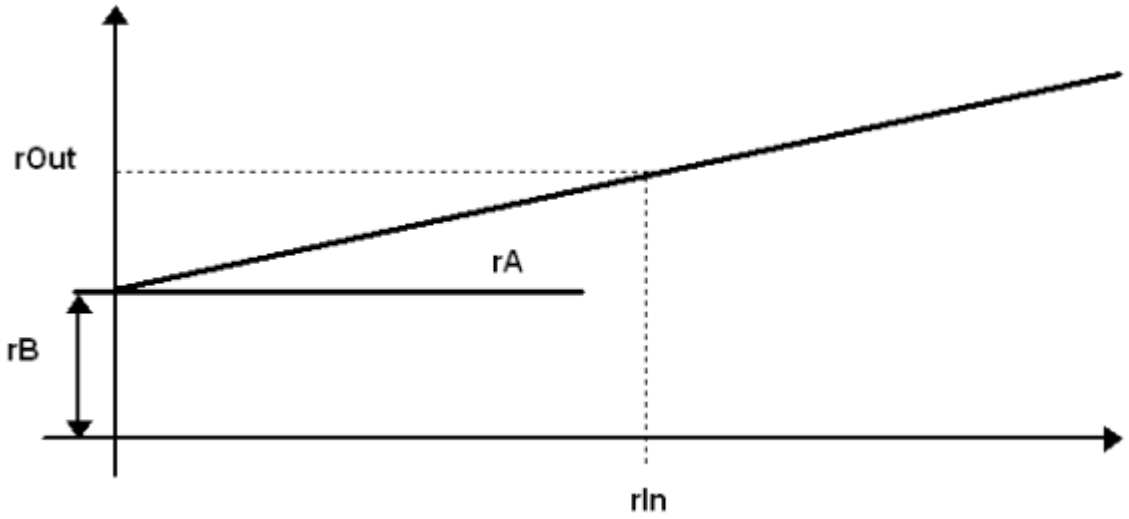
The following function blocks for the processing of analog values are available:

Function block	Description	Version	Supported articles	License
LTR	Linear transformation.	1	-	none
A2_OF_A3	Analog value selection 2 out of 3.	1	-	none
LIMITVAL	Amplitude limiter.	1	-	none
LIMITROC	Rate of change limiter.	1	-	none
ALARM_2Q	Limit value indicator with 2 alarm limits.	1	-	none
ALARM_4Q	Limit value indicator with 2 warning and 2 alarm limits.	1	-	none

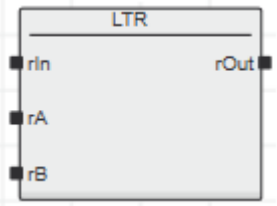
6.1 LTR

This function block realizes a linear transformation and can be used for instance for operating point shifts and gain corrections.

Function
rOut := rA * rIN + rB



6.1.1 Function block call



6.1.2 Input parameters

Name	Type	Description
rIn	REAL	Input value.
rA	REAL	Coefficient (linear equation).
rB	REAL	Zero offset.

6.1.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

6.2 A2_OF_A3

This function block monitors three signals whether they are equal inside of a predefined tolerance band DIST and applies the arithmetic mean calculated from the input signals at the output.

1. If all three input values are inside the tolerance band the average value is calculated of all three input signals. The warning bit xWarn and the alarm bit xError are set to FALSE.
2. If one of the input signals exceeds the tolerance limit (if the distance to both other input signals is greater than rDist) the average value is calculated of the remaining two input signals. The warning bit xWarn is set to TRUE and the alarm bit xError is set to FALSE.
3. If the mutual deviation between all signals is greater than the tolerance limit, the last valid calculated average value is output. The warning bit xWarn and the alarm bit xError are then set to TRUE.

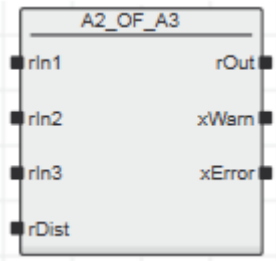
Function

xWarn := TRUE, if (rIn1 - rIn2 > rDist) AND (rIn2 - rIn3 > rDist)
OR xWarn := TRUE, if (rIn1 - rIn2 > rDist) AND (rIn1 - rIn3 > rDist)
OR xWarn := TRUE, if (rIn2 - rIn3 > rDist) AND (rIn1 - rIn3 > rDist)

xError := TRUE, if (rIn1 - rIn2 > rDist) AND (rIn2 - rIn3 > rDist) AND (rIn1 - rIn3 > rDist)

Case	xWarn	xError	rOut	Description
1	FALSE	FALSE	rOut(n):=(rIn1 + rIn2 + rIn3) / 3	rOut is error-free
2	TRUE	FALSE	rOut(n):=(INj + INk) / 2	The faulty input is ignored
3	TRUE	TRUE	rOut(n):= rOut(n-1)	Output of the last valid calculated average value

6.2.1 Function block call



6.2.2 Input parameters

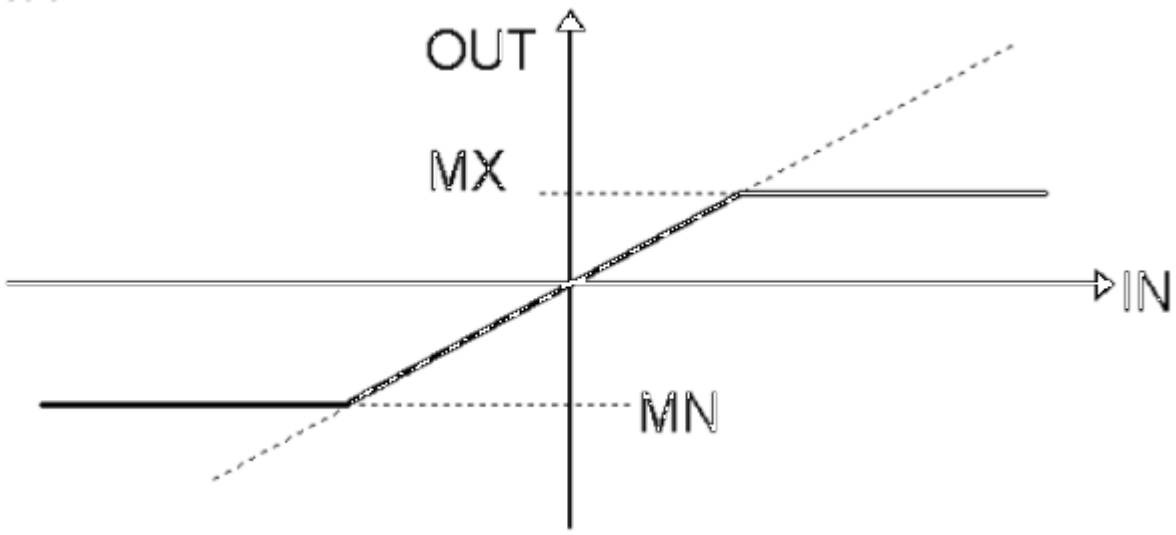
Name	Type	Description
rIn1	REAL	Input value 1.
rIn2	REAL	Input value 2.
rIn3	REAL	Input value 3.
rDist	REAL	Tolerance limit (rDist >= 0).

6.2.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.
xWarn	BOOL	Warning output.
xError	BOOL	Error output.

6.3 LIMITVAL

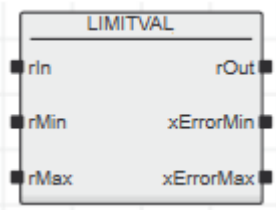
This function block limits the analog input quantity to values inside of a range which is defined by the parameters “minimum limit value” (rMin) and “maximum limit value” (rMax). If the input signal exceeds one of those two limits, the corresponding signal is set.



rOut = rMin, if rIn < rMin
rOut = rIn, if rMin <= rIn <= rMax
rOut = rMax, if rIn > rMax

Note
rMin <= rMax, if rMin > rMax then rMin is set internally = rMax.

6.3.1 Function block call



6.3.2 Input parameters

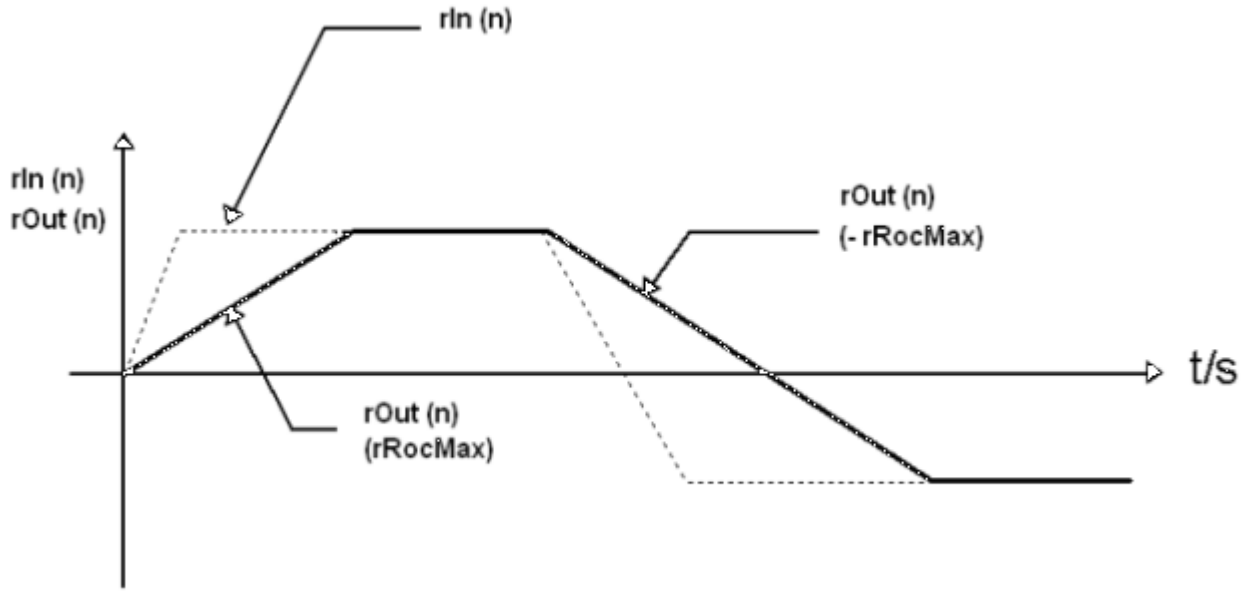
Name	Type	Description
rIn	REAL	Input value.
rMin	REAL	Minimum limit value.
rMax	REAL	Maximum limit value.

6.3.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.
xErrorMin	BOOL	Error message: Range exceeded (min. limit value).
xErrorMax	BOOL	Error message: Range exceeded (max. limit value).

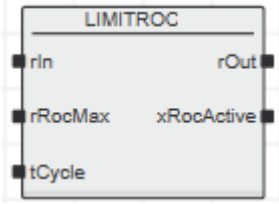
6.4 LIMITROC

This function block limits the gradient value of the input quantity to a maximum permitted rate of change of the output quantity. This gradient is predetermined at the input rRocMax. Thus the controller signals can be adapted to the actuation rate of actuators which is limited by technological conditions.



rOut = rOut(n-1) + rRocMax for rIn(n) - rOut(n-1) > rRocMax
rOut = rIn(n) for rIn(n) - rOut(n-1) <= rRocMax
rOut = rOut(n-1) - rRocMax for rIn(n) - rOut(n-1) < - rRocMax

6.4.1 Function block call



6.4.2 Input parameters

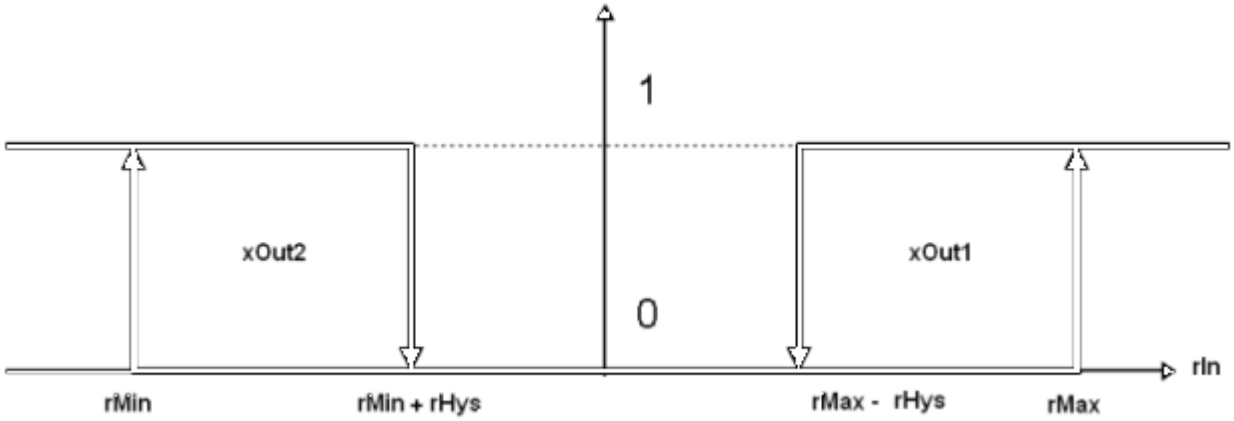
Name	Type	Description
rIn	REAL	Input value.
rRocMax	REAL	Maximum gradient / maximum rate of change of the output value per second (ROC = RATE OF CHANGE) If rRocMax < 0.0, rRocMax is set internally to 0.0.
tCycle	TIME	Cycle time.

6.4.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.
xRocActive	BOOL	Limitation active.

6.5 ALARM_2Q

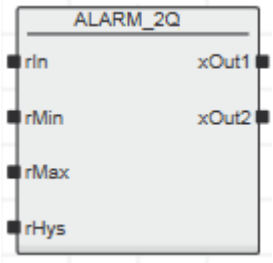
Using the function block ALARM_2Q the amplitude of analog values can be monitored. The function block ALARM_2Q can also be used as a simple two-position controller with hysteresis. A further application is the extraction of binary signals for an operating point dependant parameter switching.



xOut1 = 0 for rIn < rMax - rHys
xOut1 = xOut1 for rMax - rHys < rIn < rMax
xOut1 = 1 for rMax <= rIn

xOut2 = 0 for rMin + rHys < rIn
xOut2 = xOut2 for rMin < rIn < rMin + rHys
xOut2 = 1 for rIn <= rMin

6.5.1 Function block call



6.5.2 Input parameters

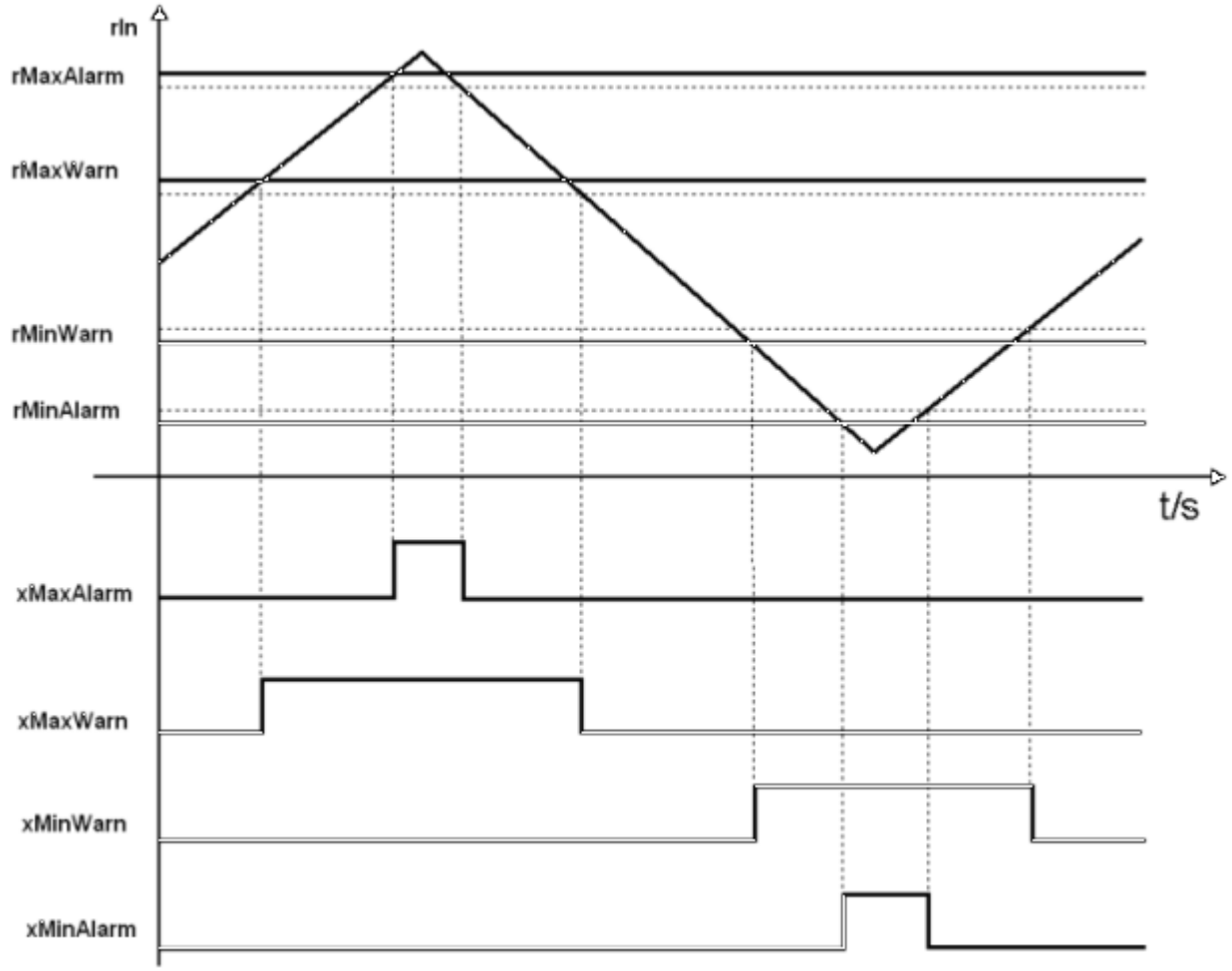
Name	Type	Description
rIn	REAL	Input value.
rMin	REAL	Lower limit value.
rMax	REAL	Upper limit value.
rHys	REAL	Hysteresis width rHys >= 0 (if rHys < 0, then rHys = 0).

6.5.3 Output parameters

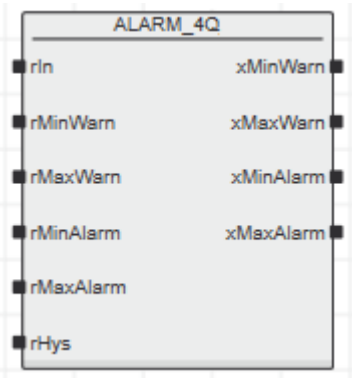
Name	Type	Description
xOut1	BOOL	Logical output 1.
xOut2	BOOL	Logical output 2.

6.6 ALARM_4Q

Using the function block ALARM_4Q the amplitude of analog values can be monitored. For this purpose 2 warning and 2 alarm limits are available. If desired, a hysteresis can be switched on.



6.6.1 Function block call



6.6.2 Input parameters

Name	Type	Description
rIn	REAL	Input value.
rMinWarn	REAL	Lower warning limit value.
rMaxWarn	REAL	Upper warning limit value.
rMinAlarm	REAL	Lower alarm limit value.
rMaxAlarm	REAL	Upper alarm limit value.
rHys	REAL	Hysteresis width rHys >= 0 (if rHys < 0, then rHys = 0).

6.6.3 Output parameters

Name	Type	Description
xMinWarn	BOOL	Range underrun warning message.
xMaxWarn	BOOL	Range overrun warning message.
xMinAlarm	BOOL	Range underrun alarm message.
xMaxAlarm	BOOL	Range overrun alarm message.

7 Function blocks for processing of binary values

The following function block for the processing of binary values is available:

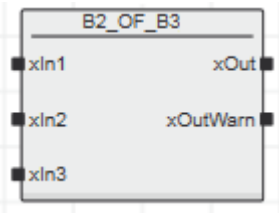
Function block	Description	Version	Supported articles	License
B2_OF_B3	Binary value selection 2 out of 3.	1	-	none

7.1 B2_OF_B3

This function block has a two out of three logic implemented and is used for the interconnection of three binary sensors. The function block is used in fault tolerant systems for the evaluation of redundant binary sensors.

1. If the input signals are equal at all three inputs, this logical value is applied to output xOut and the error bit xOutWarn is set to FALSE.
2. If only two inputs have the same logical value, this logical value is applied to output xOut and the error bit xOutWarn is set to TRUE.

7.1.1 Function block call



7.1.2 Input parameters

Name	Type	Description
xIn1	BOOL	Logical input value 1.
xIn2	BOOL	Logical input value 2.
xIn3	BOOL	Logical input value 3.

7.1.3 Output parameters

Name	Type	Description
xOut	BOOL	Logical output value (evaluation 2 out of 3).
xOutWarn	BOOL	Warning output is FALSE if all 3 inputs are equal Warning output is TRUE, only 2 inputs are equal

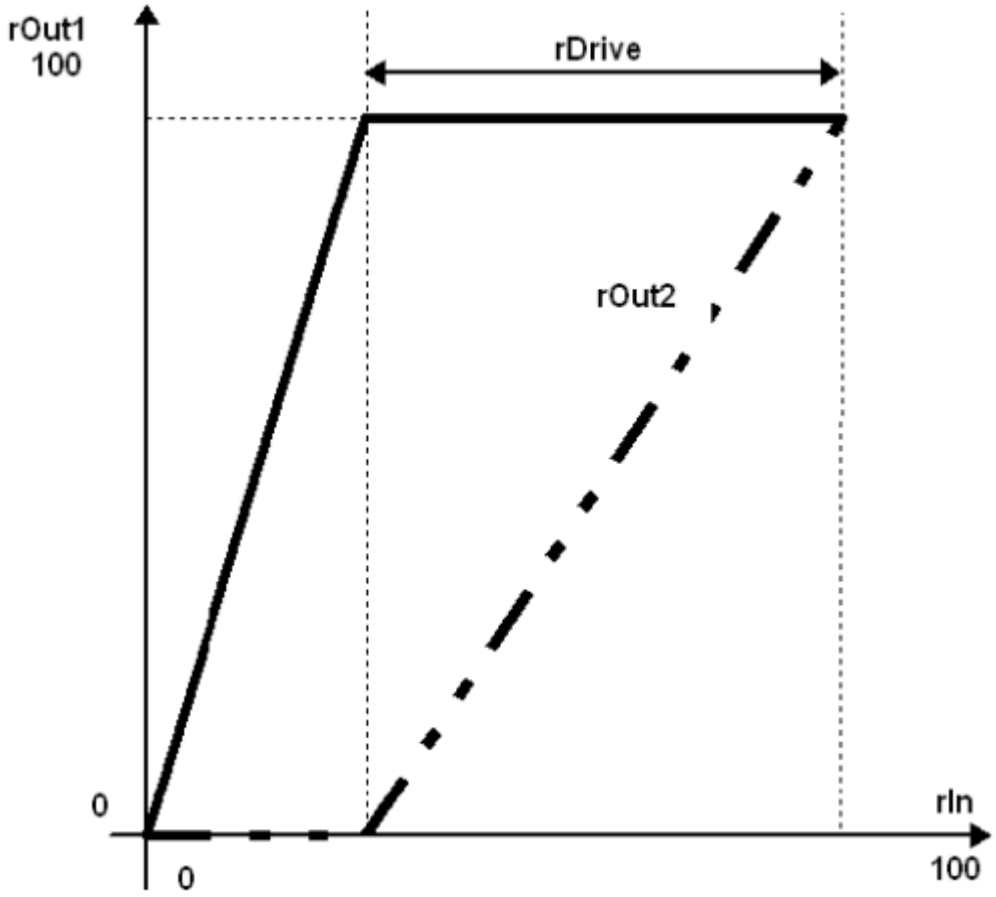
8 Function blocks for driving actuators

The following function blocks for driving actuators are available:

Function block	Description	Version	Supported articles	License
TWIN_DRIVE	Simultaneous driving of two actuators.	1	-	none
THREE	Three-position control element (two-position control element).	1	-	none
REV_LOCK	Reversing interlock.	1	-	none

8.1 TWIN_DRIVE

The TWIN_DRIVE function block is used for simultaneous control of two actuators. For this purpose, the function block provides two outputs, so that two actuators operating in parallel can be controlled in a coordinated manner. The value ranges correspond to 0.0 to 100.0 percent.



Function

$$rOut1 = \frac{rIn * 100.0}{100.0 - rDrive2}$$

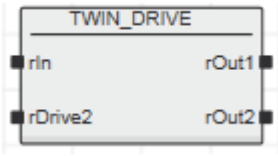
$$rOut2 = \frac{(rIn - (100.0 - rDrive2)) * 100.0}{rDrive2}$$

Note

If rDrive2 approaches the interval limits 0.0 < rDrive2 < 100.0 too close, the resulting function increases at the outputs rOut1 and rOut2 can lead to a high load for the actuators.

If rDrive2 = 0.0 the output for the actuating mechanism 2 is switched off.
If rDrive2 = 100.0 the output for the actuating mechanism 1 is switched off.

8.1.1 Function block call



8.1.2 Input parameters

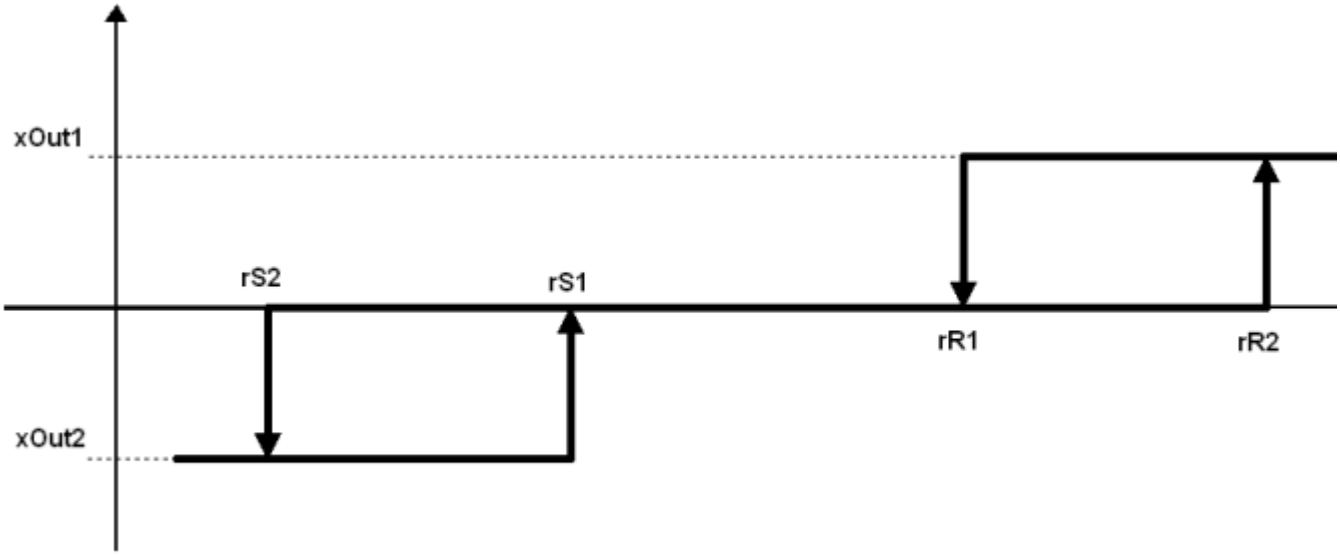
Name	Type	Description
rIn	REAL	Actuating signal (0.0 <= rIn <= 100.0).
rDrive2	REAL	Percentage of the total actuating range for the 2. actuating mechanism (0.0 < rDrive2 < 100.0).

8.1.3 Output parameters

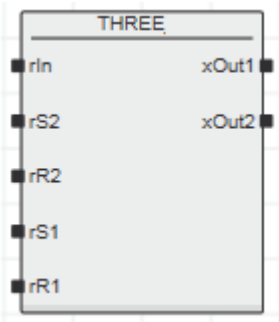
Name	Type	Description
rOut1	REAL	Output for actuating mechanism 1 (0.0 <= rOut1 <= 100.0).
rOut2	REAL	Output for actuating mechanism 2 (0.0 <= rOut2 <= 100.0).

8.2 THREE

The function block implements a three-point element. Any zero width and hysteresis width can be set with the switch-on and switch-off points. If only one output signal (output xOut1) is used, the function block can be used as a two-point element. If the condition (rS2 <= rR2 <= rR1 <= rS1) is not met, the function block is used independently of the input value. Both outputs xOut1 and xOut2 are set to FALSE.



8.2.1 Function block call



8.2.2 Input parameters

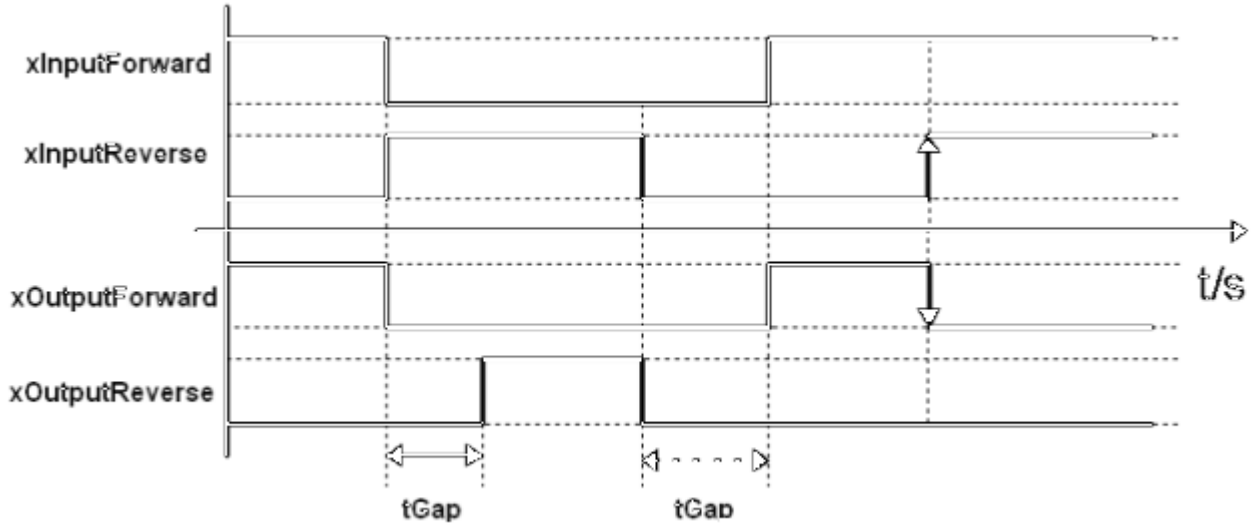
Name	Type	Description
rIn	REAL	Input value.
rS2	REAL	Operate point 2.
rR2	REAL	Release point 2.
rR1	REAL	Release point 1.
rS1	REAL	Operate point 1.

8.2.3 Output parameters

Name	Type	Description
xOut1	BOOL	Output 1.
xOut2	BOOL	Output 2.

8.3 REV_LOCK

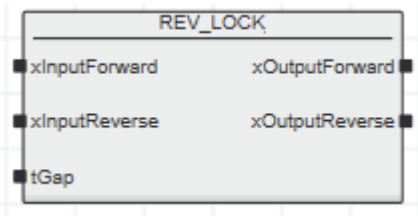
The function block prevents a direct switching between OUT1 and OUT2 (pre-switching and Return) by inserting a wait time of length tGap. For example, the function block can be assigned to the THREE three-step element and connected downstream if required by the controlled final control element.



Note

If the inputs xInputForward and xInputReverse are TRUE, the outputs xOutputForward and xOutputReverse immediately set to FALSE.

8.3.1 Function block call



8.3.2 Input parameters

Name	Type	Description
xInputForward	BOOL	Control signal forwards.
xInputReverse	BOOL	Control signal backwards.
tGap	TIME	Idle time.

8.3.3 Output parameters

Name	Type	Description
xOutputForward	BOOL	Control signal flow.
xOutputReverse	BOOL	Control signal return flow.

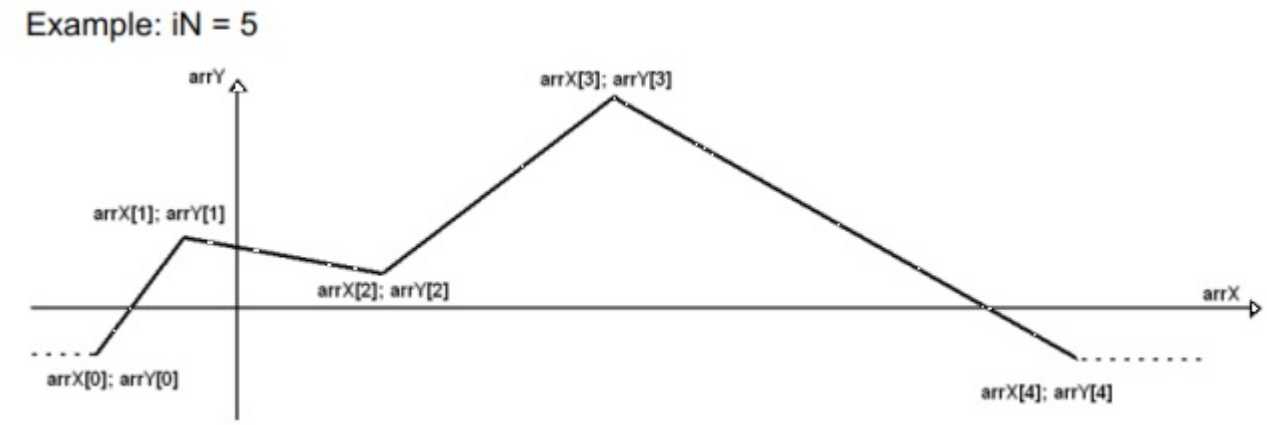
9 Function blocks for the non-linear signal processing

The following function blocks for the non-linear signal processing are available:

Function block	Description	Version	Supported articles	License
POLG_N	Polygonal line.	1	-	none
POLN_N	Polynomial.	1	-	none
DEADBAND	Dead band (without hysteresis).	1	-	none

9.1 POLG_N

The function block reproduces non-linear characteristic curves by straight line approximation. It can be used for characteristic curve correction of actuators, for non-linear feedback or non-linear couplings in controllers. The straight line approximation can be realized with a maximum of 20 reference points.



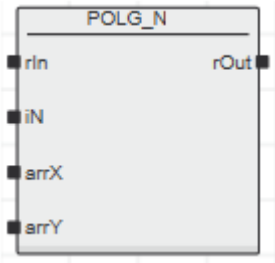
Note

All parameters are read in the first cycle and cannot be changed at runtime. The inputs “arrX” and “arrY” are of the user data type ARR20R, i.e. ARRAY[0..19] OF REAL, and must be parameterized accordingly.

When parameterizing the sampling points, note that the first parameter set is arrX[0];arrX[0] and the last parameter set is arrX[iN-1];arrY[iN-1]. The ordinate parameters arrY(I) can be freely selected. If iN is outside of the determined limits, the value 0.0 is applied to output rOut. The sequence of the abscissa parameters arrX(I) must be observed [arrX(iN) < arrX(iN+1)]. The value 0.0 is also applied to the output rOut if the parameterization is faulty.

If the value at input rIn is lower than arrX[0], then rOut:= arrY[0] is set. If the value at input rIn is greater than arrX[iN-1], then rOut:= arrY[iN-1] is set.

9.1.1 Function block call



9.1.2 Input parameters

Name	Type	Description
rIn	REAL	Input value.
iN	INT	Number of nodes (2 <= iN <= 20).
arrX	ARR20R	Nodes arrX(I) (arrX(I) < arrX(I+1)).
arrY	ARR20R	Nodes arrY(I).

9.1.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

9.2 POLN_N

This function block realizes a polynomial up to the 9th order maximally. It can be used for correcting characteristic curves, for example.

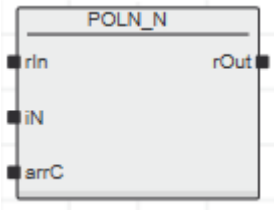
Function

rOut := arrC[0] * rIn 0 + arrC[1] * rIn 1 + … + arrC[9] * rIn 9

Note

All parameters are read in during the first cycle and cannot be changed at runtime. The input “arrC” is of the user data type ARR10R, i.e. ARRAY[0..9] OF REAL, and must be parameterized accordingly. If iN is outside the specified limits, then the value “0.0” is output at “rOut”.

9.2.1 Function block call



9.2.2 Input parameters

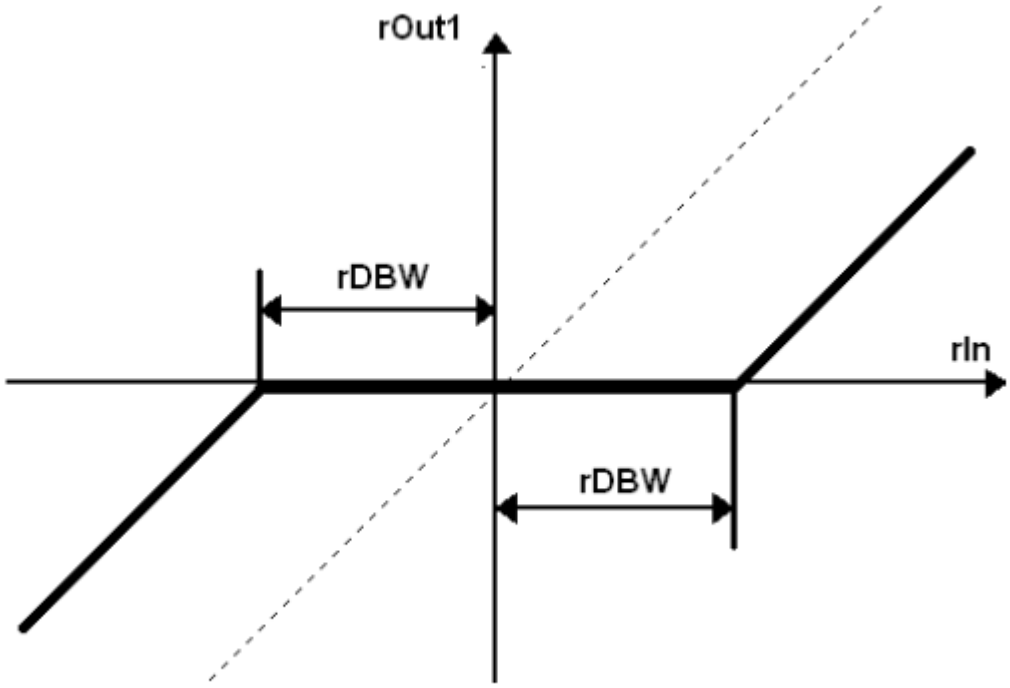
Name	Type	Description
rIn	REAL	Input value.
iN	INT	Degree of the polynomial (0 <= iN <= 9).
arrC	ARR10R	Polynomial coefficients.

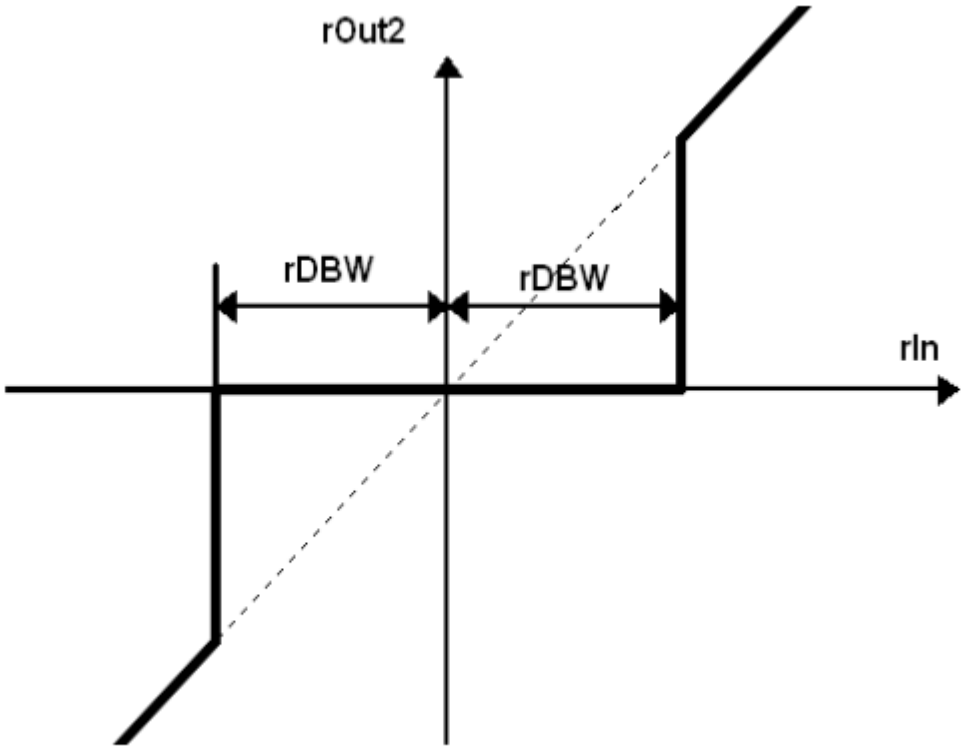
9.2.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

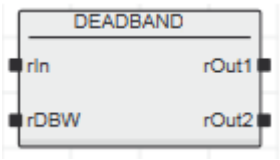
9.3 DEADBAND

This function block realizes an adjustable dead band without hysteresis. Two outputs can be wired to the requirements accordingly. The function block can be used to suppress the quantization noise of analog-to-digital converters (ADC), for example.





9.3.1 Function block call



9.3.2 Input parameters

Name	Type	Description
rIn	REAL	Input value.
rDBW	REAL	Half of the dead band width (rDBW >= 0, if rDBW < 0, then rDBW = 0).

9.3.3 Output parameters

Name	Type	Description
rOut1	REAL	Output value 1.
rOut2	REAL	Output value 2.

10 Function blocks for dynamic signal processing

The following function blocks for the dynamic signal processing are available:

Function block	Description	Version	Supported articles	License
INT_C	Integrator.	1	-	none
LAG1ST	PT1 element / PT1 filter.	1	-	none
DYN	Dynamic element.	1	-	none
HOLD	Holding element.	1	-	none

10.1 INT_C

The function block can be used as a simple I-controller or in calculation functions (e.g. for calculating quality values). The trapezoidal rule is used for integration.

Function

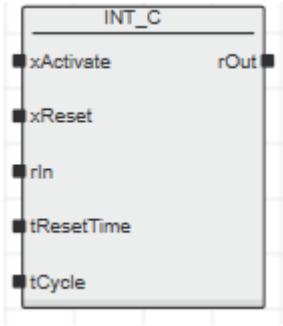
Transfer function in the continuous range
$G(s) = \frac{1}{s * tResetTime}$

Note

Recommended range of values for parameter RESET_T:
0.1 * tCycle <= tResetTime <= 10^4 *tCycle.

If tResetTime = T#0s the last output value is held (refer to holding element).

10.1.1 Function block call



10.1.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xReset	BOOL	Rising edge: Resets the function block.
rIn	REAL	Input value.
tResetTime	TIME	Integration time constant.
tCycle	TIME	Sampling time.

10.1.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

10.2 LAG1ST

This function block realizes a delay element of the first order including a PT1 transfer function. For instance, to perform a digital filtering it can be used individual as well as cascaded.

Function

Transfer function in the continuous range

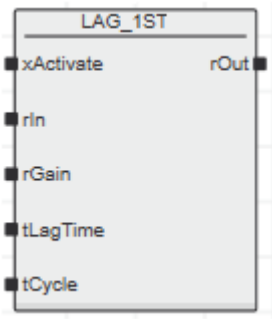
$$G(s) = \frac{rGain}{1 + s * tLagTime}$$

Note

Recommended range of values for parameter tLagTime:
$0.5 * tCycle <= tLagTime <= 10^4 * tCycle$.

If tLagTime = 0 (T#0s) then rOut = tGain * rIn.

10.2.1 Function block call



10.2.2 Input parameters

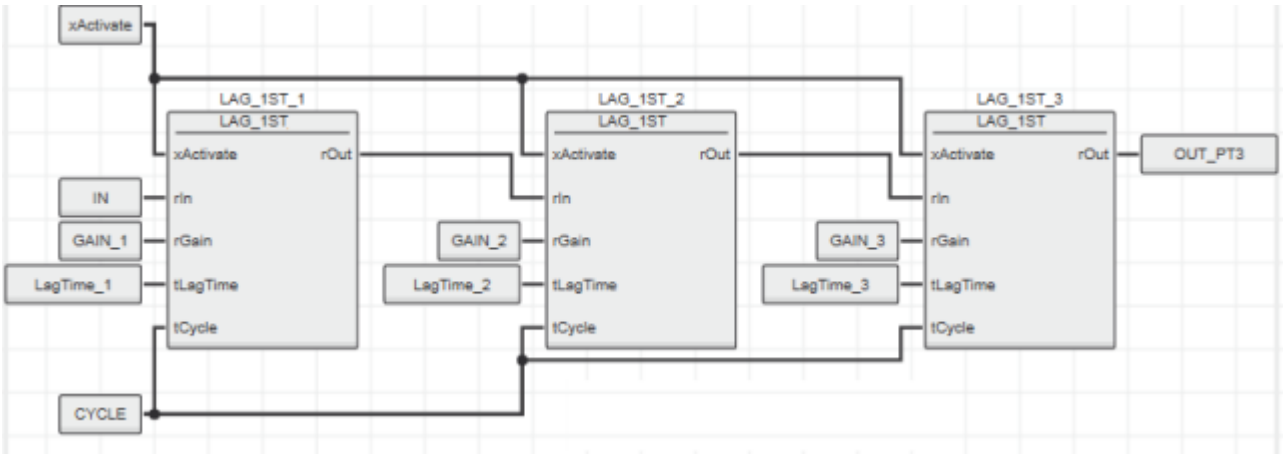
Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rIn	REAL	Input value.
rGain	REAL	Gain factor (proportinal constant).
tLagTime	TIME	Time constant.
tCycle	TIME	Sampling time.

10.2.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

Example

The following figure shows a cascading of three LAG1ST function blocks in order to simulate a PT3 behavior.



10.3 DYN

This function block realizes a PDT1, PT1 or a DT1 behavior. It can be used for the dynamic signal processing (e.g. disturbance-variable compensation).

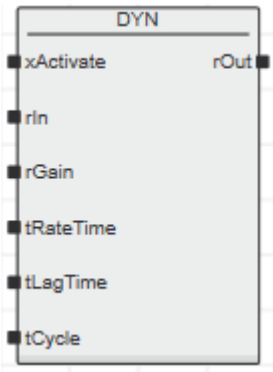
Function

Transfer functions in the continuous range		
PDT1	rGain<> 0 tRateTime <> 0 tLagTime >0	$G(s) = rGain * \frac{1 + s * tRateTime}{1 + s * tLagTime}$
PT1	rGain<> 0 tRateTime = 0 tLagTime > 0	$G(s) = rGain * \frac{1}{1 + s * tLagTime}$
DT1	rGain= 0 tRateTime <> 0 tLagTime > 0	$G(s) = \frac{s * tRateTime}{1 + s * tLagTime}$

Note

Recommended ranges of values for the parameters RATE_T and LAG_T:
0.1 * tCycle <= rRateTime <= 10^4 * tCycle
0.5 * tCycle <= tLagTime <= 10^4 * tCycle

10.3.1 Function block call



10.3.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rIn	REAL	Input value.
rGain	REAL	Amplification.
tRateTime	TIME	Derivative rate time.
tLagTime	TIME	Lag time.
tCycle	TIME	Sampling time.

10.3.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

10.4 HOLD

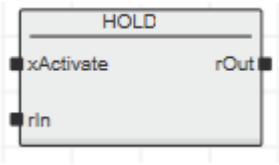
This function block is used to hold (store) process values.

If xActivate = FALSE, the function block switches the input signal rIn through to the output rOut. As long as xActivate = TRUE, the output signal is held.

Function

rOut (n) = rIn (n) for xActivate = FALSE
rOut (n) = rOut (n-1) for xActivate = TRUE

10.4.1 Function block call



10.4.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rIn	REAL	Input value.

10.4.3 Output parameters

Name	Type	Description
rOut	REAL	Output value.

11 Controller blocks

The following controller blocks are available:

Function block	Description	Version	Supported articles	License
C_N	Keying controller of the n-th order.	1	-	none
PID_C	Continuous PID-type controller.	1	-	none
PID_R	Continuous PID-type controller (with reduced performance range).	1	-	none
THREE_C	Three-position controller attachment for PID_C and PID_R.	1	-	none
PID_ADA	PID-type controller attachment module for controlled adaption.	1	-	none
PID_STR	PID-type controller attachment module for controlled adaption.	1	-	none
PID_MAN	PID-type controller attachment module for the input of manual manipulated values.	1	-	none
PID_MODE	PID-type controller attachment module for the input of the MODE control commands.	1	-	none
PID_PAR	PID-type controller attachment module for the input of parameterizing values.	1	-	none
PID_STAT	PID-type controller attachment module for the output of status information.	1	-	none

11.1 C_N

The function block C_N realizes a keying controller of the n-th order. A maximum of 10 numerator coefficients and accordingly 9 denominator coefficients can be parameterized. By default the 0th denominator coefficient is always “1”. Using the keying controller function block a number of various higher controller functions can be performed. Applications are deadbeat controllers, minimum variance controllers or pole preset controllers, for instance.

The keying controller C_N has only the “pure” controller algorithm integrated. Set point and actual value limitations or a rate of rise monitoring of the set point and the manipulated value are not integrated. In conjunction with the function blocks LIMITVAL and LIMITROC an user performed universal configuration is possible.

The z transfer function is realized.

$$G(z) = \frac{T0 + T1 * (z - 1) + \ldots + TN * (z - N)}{1 + R1 * (z - 1) + \ldots + RN * (z - N)} \quad ; \quad \frac{T0 = arrTC[0] ; \ldots\ldots ; T9 = arrTC[C9]}{R1 = arrRC[1]; \ldots\ldots ; R9 = arrRC[9]}$$

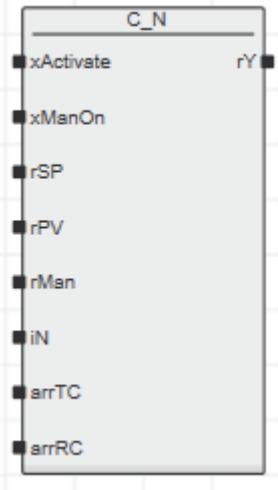
Note

The order “iN” is determined at the keying controller start and cannot be changed while the process is running. Changes to the input “iN” are not activated until a restart occurs.

The coefficients can be parameterized online. However, critical transient processes can occur during the initial transient for the new parameters which depend on the current internal conditions.

The field element arrRC[0] is only existent due to software engineering reasons and does not have any function. Parameterization is not required.

11.1.1 Function block call



11.1.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xManOn	BOOL	TRUE: Switch manual input.
rSP	REAL	Set point (reference input variable).
rPV	REAL	Process (controlled) variable.
rMan	REAL	Manual value.
iN	INT	Order N <= 9.
arrTC	ARR10R	Numerator coefficient.
arrRC	ARR10R	Denominator coefficient.

11.1.3 Output parameters

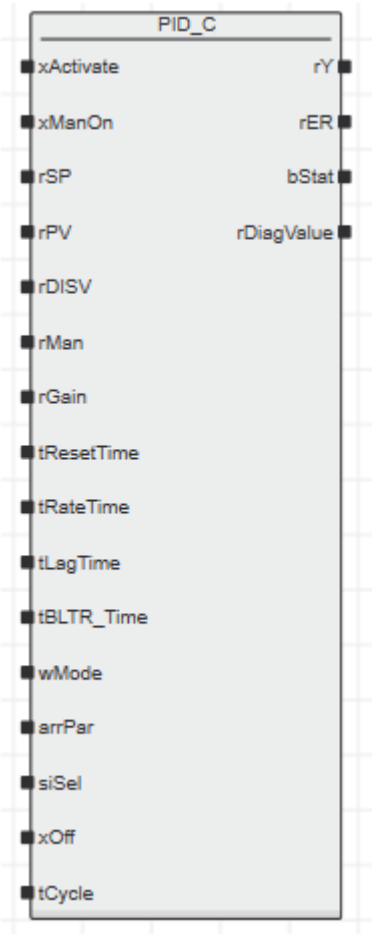
Name	Type	Description
rY	REAL	Manipulated variable.

11.2 PID_C

This function block realizes a PID-type controller with a delayed D portion including all possible sub variants (PI, PD, P and I). The transfer functions which are equivalent to continuous-action controllers are:

PID	$G(s) = rGain * \left(1 + \frac{1}{s * tResetTime} + \frac{s * tRateTime}{1 + s * tLagTime} \right)$
PI	$G(s) = rGain * \left(1 + \frac{1}{s * tResetTime} \right)$
PD	$G(s) = rGain * \left(1 + \frac{s * tRateTime}{1 + s * tLagTime} \right)$
P	$G(s) = rGain$
I	$G(s) = \frac{1}{s * tResetTime}$

11.2.1 Function block call



11.2.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xManOn	BOOL	TRUE: Switch manual input.
rSP	REAL	Set point (reference input variable).
rPV	REAL	Process (controlled) variable.
rDISV	REAL	Disturbance-variable compensation.
rMan	REAL	Manual value.
rGain	REAL	Proportional constant Kr.
tResetTime	TIME	Reset time Tn.
tRateTime	TIME	Derivative rate time Tv.
tLagTime	TIME	Additional time constant Tz in the D portion.
tBLTR_Time	TIME	Transition time period for chock-free manual-automatic switching; no shock-free switching is performed if tBLTR_Time:= T#0s.
wMode	WORD	wMode control input (direct or PID_MODE).
arrPar	ARR10R	PARameterization (direct or PID_PAR).
siSel	SINT	Selection for output “rDiagValue”.
xOff	BOOL	Connect to output xOff of the function block THREE_C.
tCycle	TIME	Sampling time.

11.2.3 Output parameters

Name	Type	Description
rY	REAL	Manipulated variable.
rER	REAL	Control deviation after set point limitation.
bStat	BYTE	STATus output (direct or PID_STAT).
rDiagValue	REAL	Diagnosis value (according to selection “siSel”).

11.2.4 Functional principle

The function of the implemented PID-type controller is based on difference equations which are calculated through a discretization with the sampling time tCycle. The realized function block works internally based on the velocity algorithm. This means that it internally first supplies an incremental signal for the actuating signal. With each sampling step this incremental signal is then summed to a entire value signal (position signal).

Controller portions are parameterized by entering the parameters (rGain, tResetTime, tRateTime and tLagTime). In order to switch off individual functions (P, I or D portion) the parameters of these functions must be set to zero.

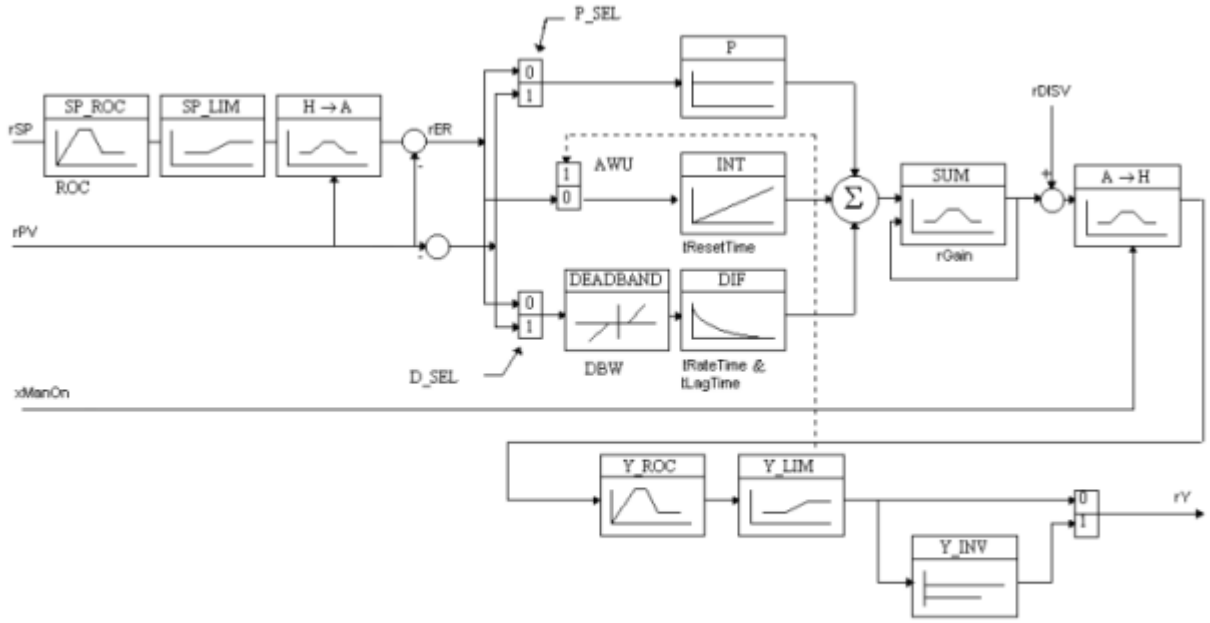
By performing a suitable selection of the sampling time tCycle and the controller parameters rGain, tResetTime, tRateTime and tLagTime, the PID-type function block can be used as a quasi-continuous controller and as a controller which is operated in a clearly discontinuous mode of operation.

The actuating increments may not become too small in order to ensure that minimum changes of the individual input signal can cause changes at the output. The size of these increments is considerably influenced by the ratio of the controller function block cycle time (CYCLE time of the CYCLE task) to the individual controller parameters.

11.2.5 Range of values

Regarding to the internal algorithms of the PID function block the following ranges of values are recommended for the parameters rResetTime, tRateTime and tLagTime.

Variable	Description	Value range
tResetTime	Reset time Tn	5 * tCycle <= tResetTime <= 1000 * tCycle
tRateTime	Derivative rate time Tv	tCycle <= tRateTime <= 1000 * tCycle
tLagTime	Additional time constant Tz in the D portion	0.5 * tCycle <= tLagTime <= 1000 * tCycle



11.2.6 Encoding of the structure control, parameterizing and status variables

The structure control of the controller is performed by the way of seizing the parameters.

	rGain	tResetTime	tRateTime	tLagTime
P	<> 0.0	= T#0s	= T#0s	= T#0s
I	= 0.0	<> T#0s	= T#0s	= T#0s
PI	<> 0.0	<> T#0s	= T#0s	= T#0s
PD	<> 0.0	= T#0s	<> T#0s	<> T#0s
PID	<> 0.0	<> T#0s	<> T#0s	<> T#0s

If (tRateTime > T#0s) & (tLagTime = T#0s), then tLagTime = tRateTime / 5.

11.2.7 Control word MODE

Several switching inputs of the PID_C controller were put together in one wMode control word. Access can be performed either directly or via the function block PID_MODE.

Variables of the input wMode:

PAR_OFF	BOOL	FALSE: The connected PID_C reads-in the MODE control commands and the parameter values (rGain, tResetTime, tRateTime and tLagTime) again during every new cycle. TRUE: No parameter values are read-in, i.e. they are “frozen”.
YINV_ON	BOOL	TRUE : rY is inverted: rY := - rY
D_SEL	BOOL	FALSE: D element obtains control deviation (rSP-rPV) TRUE: D element obtains process variable (rPV).Selector switch for switching on the D portion.
P_SEL	BOOL	FALSE: P element obtains control deviation (rSP - rPV) TRUE: P element obtains process variable (rPV). Selector switch for switching on the P portion.
WU_ON	BOOL	TRUE : Switch on “Wind-up”.
DB_ON	BOOL	TRUE : Switch on dead band located before D portion.
SPROC_ON	BOOL	TRUE: Switch on the set point rising limiter.
YROC_ON	BOOL	TRUE: Switch on the manipulated value rising limiter.
HALT	BOOL	TRUE : The last manipulated value is held (in automatic mode).
RESET	BOOL	TRUE : (positive edge)Reset of all internal status variables.

Structure of the control word wMode:

	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
wMode : WORD	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAR_OFF	X															
YINV_ON		X														
not used			X													
D_SEL				X												
P_SEL					X											
not used						X										
WU_ON							X									
DB_ON								X								
not used									X							
SPROC_ON										X						
YROC_ON											X					
not used												X				
not used													X			
not used														X		
HALT															X	
RESET																X

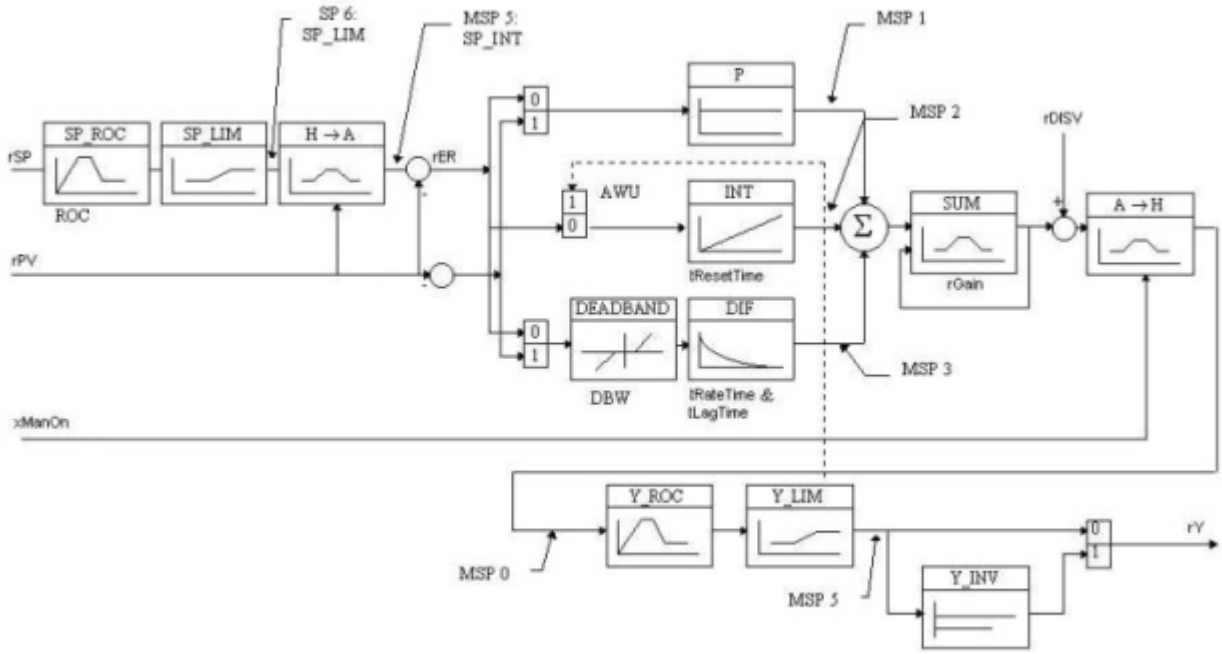
11.2.8 Parameter array arrPar

The assignment of the input arrPar can be done either directly or via the function block PID_PAR. If the input arrPar is not assigned or if the value arrPar [9] is zero, the internal default settings of PID_C are valid.

	Variable	Data type	Description	Default setting
arrPar[0]	SP_MN	REAL	Set point limitation (lower limit).	0.0(%)
arrPar[1]	SP_MX	REAL	Set point limitation (upper limit).	100.0(%)
arrPar[2]	SP_ROC	REAL	Set point limitation (rate of rise).	10.0(%)
arrPar[3]	Y_MN	REAL	Manipulated value limitation (lower limit).	0.0(%)
arrPar[4]	Y_MX	REAL	Manipulated value limitation (upper limit).	100.0(%)
arrPar[5]	Y_ROC	REAL	Manipulated value limitation (rate of rise).	10.0(%)
arrPar[8]	DBW	REAL	Dead band width before D portion.	0.3(%)
arrPar[9]	PAR_ON	REAL	arrPar[9]:= 0.0 if the internal default settings are used. arrPar[9]:= 1.0 if the external values are used. (automatically set when xActive=TRUE if the function block PID_PAR is used)	

11.2.9 siSel and rDiagValue

Using the siSel input the particular measuring points (MSP) can be switched to the rDiagValue output. The value to be entered for siSel then corresponds to the numbering (see figure).
For example, siSel = 0 switches the measuring point MSP0 to the output rDiagValue.



11.2.10 Status output bStat

The output STAT is used for replying various internal states. It can be read out either directly or via the function block PID_STAT.

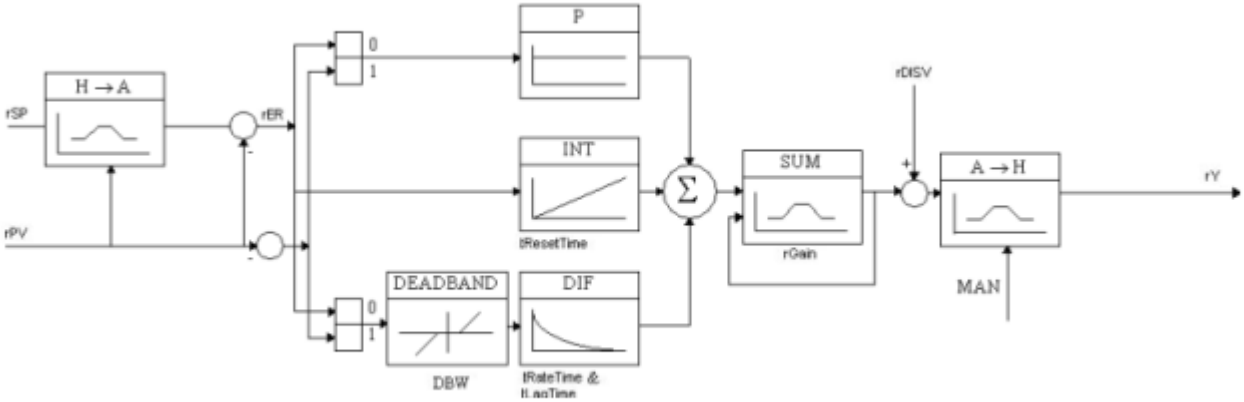
		x	x	x	x	x	x	x	x
bStat (Datentyp BYTE)		7	6	5	4	3	2	1	0
QSP_MN	Lower limitation of set point SP	X							
QSP_MX	Upper limitation of set point SP		X						
QSP_ROC	Set point rising limitation active			X					
QY_MN	Lower limitation of manipulated value Y				X				
QY_MX	Upper limitation of manipulated value Y					X			
QY_ROC	Manipulated value rising limitation active						X		
QWU_ACTV	„Wind-up“ active (limitation reached)							X	
PAR_ERR	Parameter error								X

11.3 PID_R

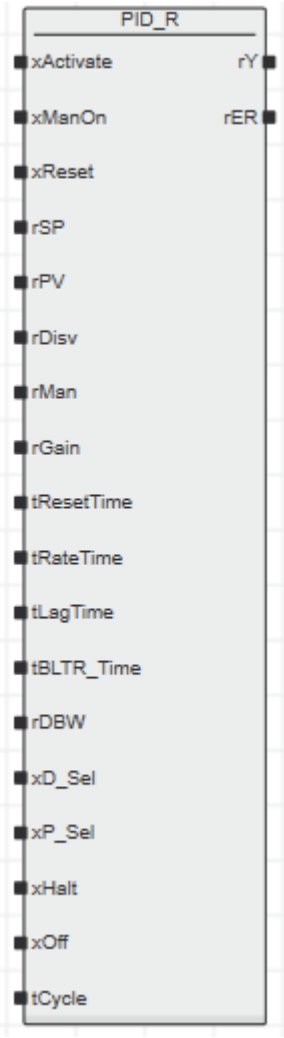
This function block realizes the functionality of a PID-type controller having a reduced performance range compared with PID_C ('_R' = reduced).

Function

Refer to the function block PID_C for a description of the performance range. In contrast to the PID_C no set point or actual value limitations or rate of rise monitoring is integrated. Furthermore the function “Auto- Windup” is not available. In conjunction with the function blocks LIMITVAL and LIMITROC an user performed universal configuration is possible.



11.3.1 Function block call



11.3.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xManOn	BOOL	TRUE: The manual value (paramater rMan) is applied at output rY.
xReset	BOOL	TRUE: (positiv edge) Reset of internal status variables.
rSP	REAL	Set point (reference input variable).
rPV	REAL	Process (controlled) variable.
rDisv	REAL	Disturbance-variable compensation.
rMan	REAL	Manual value (see xManOn).
rGain	REAL	Proportional constant Kr.
tResetTime	TIME	Reset time Tn.
tRateTime	TIME	Derivative rate time Tv.
tLagTime	TIME	Additional time constant Tz in the D portion.
tBLTR_Time	TIME	Transition time period for shock-free manual-automatic switching; Switching shock occurs if tBLTR_Time:= 0.0.
rDBW	REAL	Dead band widht before D portion.
xD_Sel	BOOL	FALSE: D element obtains control deviation (rSP - rPV). TRUE: D element obtains process variable (rPV).
xP_Sel	BOOL	FALSE: P element obtains control deviation (rSP - rPV). TRUE: P element obtains process variable (rPV).
xHalt	BOOL	TRUE : The last manipulated value is held (in automatic mode).
xOff	BOOL	Connect to output xOff of the function block THREE_C!
tCycle	TIME	Sampling time.

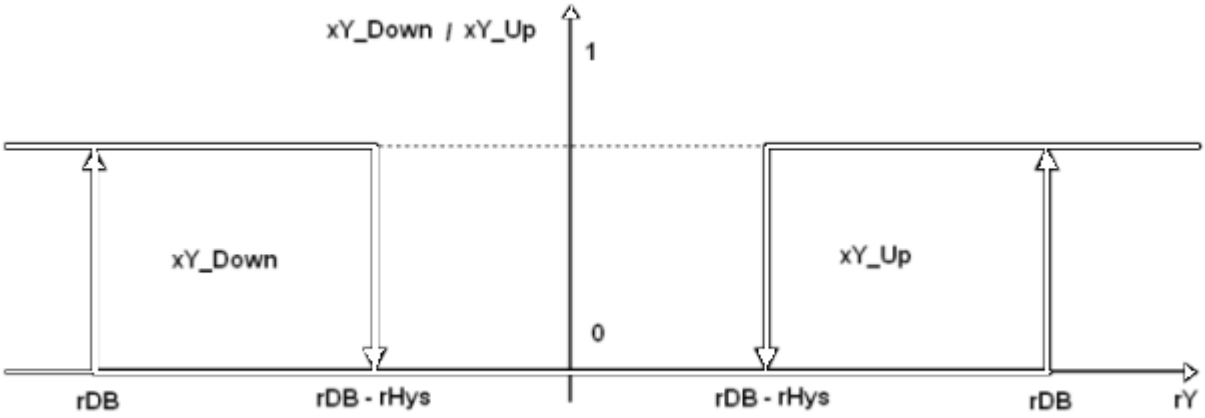
11.3.3 Output parameters

Name	Type	Description
rY	REAL	Manipulated variable.
rER	REAL	Control system deviation (rER:= rSP - rPV).

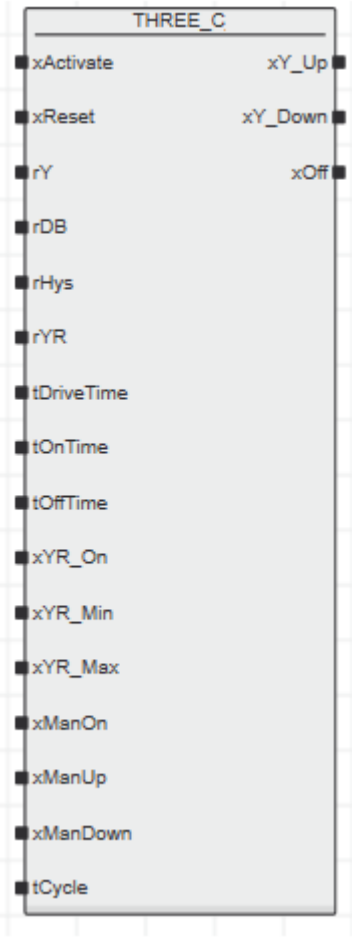
11.4 THREE_C

The PID-type controller attachment module THREE_C is used in conjunction with the continuous PID- type controllers PID_C and PID_R as a three-position step controller or as a three-position or two- position controller. The function block THREE_C converts the analog manipulated variable coming from PID_C or PID_R into boolean UP and DOWN pulses.
If a motor-driven actuator (having I behaviour) is connected to the output, existing limit stop signals can be evaluated as well as analog position reply signals.

Function



11.4.1 Function block call



11.4.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
xReset	BOOL	Rising edge: Resets the function block.
rY	REAL	Manipulated variable from PID_C or PID_R.
rDB	REAL	Operating point of the three-position control element (zero-point symmetrical).
rHys	REAL	Hysteresis in relation to “DB”.
rYR	REAL	Position reply from acuator.
tDriveTime	TIME	Motor actuating time.The motor actuating time tDriveTime must be set to T#0s if no motor-driven actuator with position reply is connected.
tOnTime	TIME	Minimum on-period.
tOffTime	TIME	Minimum pause-period.
xYR_On	BOOL	Activation of position reply.
xYR_Min	BOOL	Position reply: minimum stop.
xYR_Max	BOOL	Position reply: maximum stop.
xManOn	BOOL	TRUE: Manual operation (xManUp and xManDown activated) FALSE: Automatic operation (xManUp and xManDown deactivated)
xManUp	BOOL	Switch output xY_Up.
xManDown	BOOL	Switch output xY_Down.
tCycle	TIME	Sampling time / cycle time.

11.4.3 Output parameters

Name	Type	Description
xY_Up	BOOL	Output pulse “UP”.
xY_Down	BOOL	Output pulse “DOWN”.
xOff	BOOL	Connect to input “xOff” at PID_C or PID_R.

11.4.4 Application in connection with PID_C or PID_R

The following figure shows the communication connections between the THREE_C three-point controller and the PID_C controller. The connections between THREE_C and PID_R are identical.

11.5 PID_ADA

This PID-type controller attachment module is used in conjunction with the continuous PID-type controller PID_C or PID_R in order to realize a controlled adaption. The function block allows a selection out of a maximum of four parameter sets. By means of cascading also a higher number of parameter sets is possible. When connecting the PID_ADA to PID_C or PID_R the function block PID_STR must be inserted.

11.5.1 Function block call

11.5.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
udtIn1	STR	Parameter structure 1.
udtIn2	STR	Parameter structure 2.
udtIn3	STR	Parameter structure 3.
udtIn4	STR	Parameter structure 4.
rY	REAL	Process size.
rP1	REAL	Flip-over point 1.
rP2	REAL	Flip-over point 2 rP2 > rP1.
rP3	REAL	Flip-over point 3 rP3 > rP2.

11.5.3 Output parameters

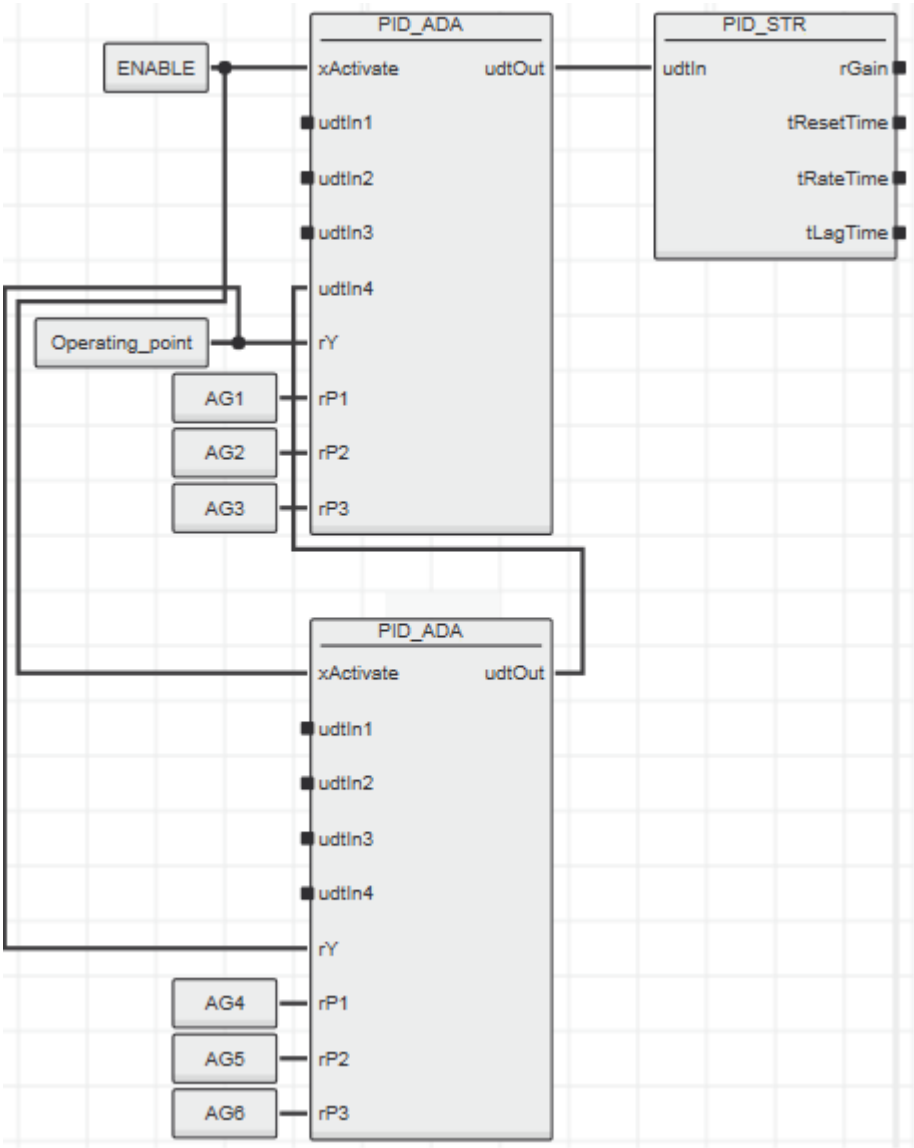
Name	Type	Description
udtOut	STR	Parameter structure of the output.

Note

If rP3 > rP2 > rP1 is not fulfilled all four output parameters udtOut are set to zero!
If only two flip-over points (i.e. 3 full operating ranges) are used, rP3 must be selected higher than the maximum value of the input variable rY.

11.5.4 Example Cascading

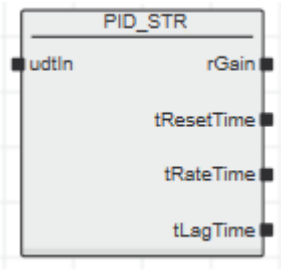
The following figure illustrates the possibilities for cascading of the adaption modules PID_ADA having seven full operating ranges. It has to be observed that the full operating range limits AG1..6 must be parameterized with ascending values.



11.6 PID_STR

This PID-type controller attachment module is used in conjunction with the continuous PID-type controller PID_C and PID_ADA in order to realize a controlled adaption. This function block converts a parameter set which is available in the form of the structure variable udtIn into the four controller parameters rGain, tResetTime, tRateTime and tLagTime.

11.6.1 Function block call



11.6.2 Input parameters

Name	Type	Description
udtIn	STR	Input parameter set.

11.6.3 Output parameters

Name	Type	Description
rGain	REAL	Proportional constant Kr.
tResetTime	TIME	Reset time Tn.
tRateTime	TIME	Derivative rate time Tv.
tLagTime	TIME	Additional time constant Tz in the D portion.

11.7 PID_MAN

This PID-type controller attachment module is used in conjunction with the continuous PID-type controller PID_C or PID_R to allow the input of manual manipulated values using the pushbuttons “UP” and “DOWN”.

Function

The output xManOn is set to TRUE after input xActivate is activated. In addition, the applied set point (rSP) is also written to output ’rMan’. Subsequently the value ’rMan’ can be changed in incremental steps (rInc) starting from this value.

Each time the input xManUp or xManDown is set to TRUE for less than one second, the output value is increased (xManUp) or decreased (xManDown) by one increment (rInc).

Each time the input xManUp or xManDown is set to TRUE for more than one second, the output value is continuously increased or decreased by one increment until the logical state FALSE is applied to the inputs xManUp or xManDown, respectively. The increment rate is rInc per second:

Change per second = 1 * rInc * tCycle (in msec) / 1000.0

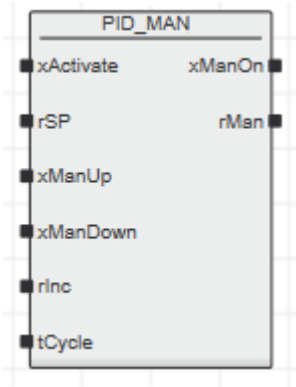
If the input xManUp or xManDown is set to TRUE for more than five seconds, the rate of change is increased tenfold.

Change per second = 10 * rInc * tCycle (in msec) / 1000.0

Note

If the output rMan shall not be changed starting from the set point (rSP) any other value (e.g. zero) can be applied to input rSP.

11.7.1 Function block call



11.7.2 Input parameters

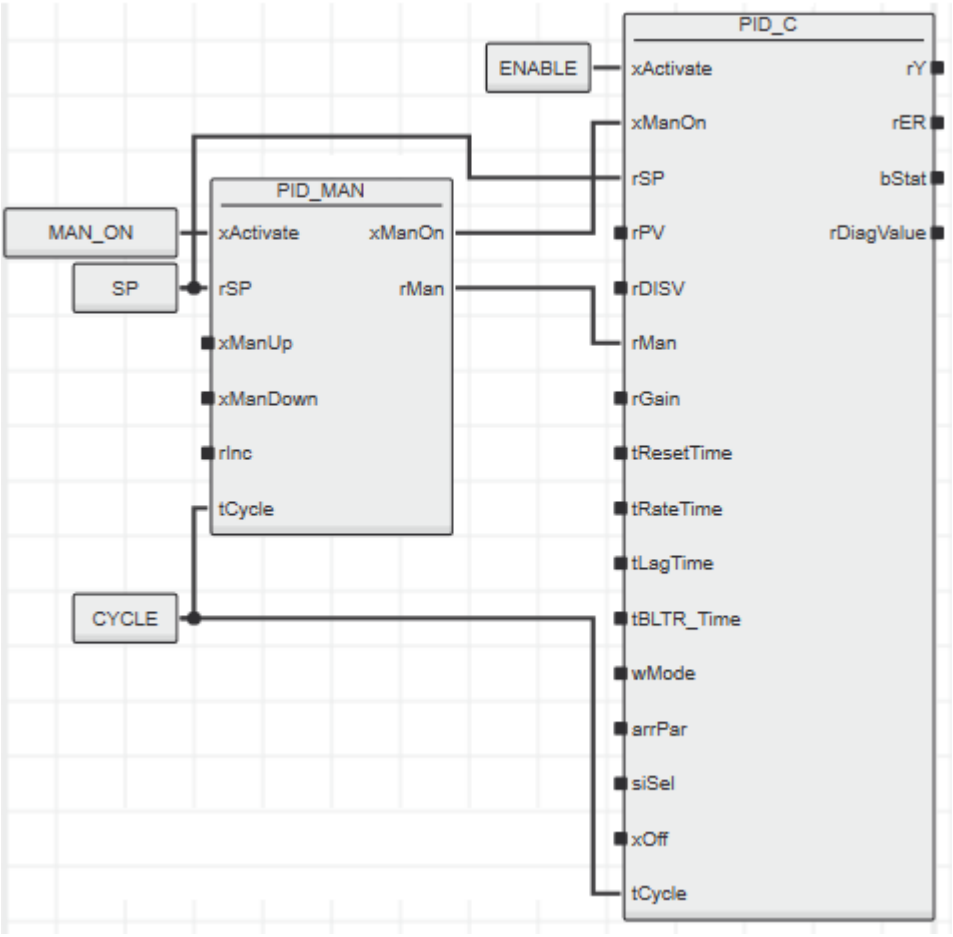
Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rSP	REAL	Set point input fort he automatic tracking of rMan.
xManUp	BOOL	Step up.
xManDown	BOOL	Step down.
rInc	REAL	Step size.
tCycle	REAL	Task cycle time.

11.7.3 Output parameters

Name	Type	Description
xManOn	BOOL	Must be connected to the controller block xManOn input.
rMan	REAL	Output value.

11.7.4 Application in connection with PID_C or PID_R

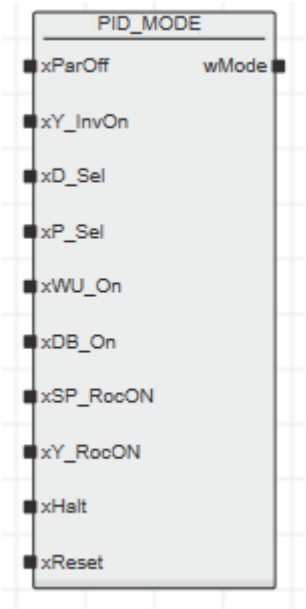
The following figure illustrates the communication interconnections between PID_MAN and the controller PID_C. The connections between PID_MAN and PID_R are identical. The function block PID_C is not completely wired for reasons of clarity.



11.8 PID_MODE

This PID-type controller attachment module is used in conjunction with the continuous PID-type controller PID_C to allow the clear input of the wMode control commands. The data exchange is performed via a structure of the type WORD. This function block is not absolutely necessary for the function of PID_C.

11.8.1 Function block call



11.8.2 Input parameters

Name	Type	Description
xPar_OFF	BOOL	FALSE: The connected PID_C newly reads-in the wMode and arrPar control commands as well as rGain, tResetTime, tRateTime and tLagTime during every cycle. TRUE: The parameter values are not read-in again.
xY_InvOn	BOOL	TRUE: rY is inverted: rY := - rY.
xD_Sel	BOOL	FALSE: D element obtains control deviation (rSP-rPV). TRUE:: D element obtains process variable (rPV) (Selector switch for switching on the D portion).
xP_Sel	BOOL	FALSE: P element obtains control deviation (rSP-rPV). TRUE:: P element obtains process variable (rPV) (Selector switch for switching on the D portion).
xWU_On	BOOL	TRUE: “WIND-UP” active.
xDB_On	BOOL	TRUE: Dead band located before D portion is switched on.
xSP_RocON	BOOL	TRUE: Set point rising limiter switched on.
xY_RocON	BOOL	TRUE: Manipulated value rising limiter switched on.
xHalt	BOOL	TRUE: The last manipulated value is held (in automatic mode).
xReset	BOOL	Rising edge: Resets the function block.

11.8.3 Output parameters

Name	Type	Description

wMode	WORD	Output word.

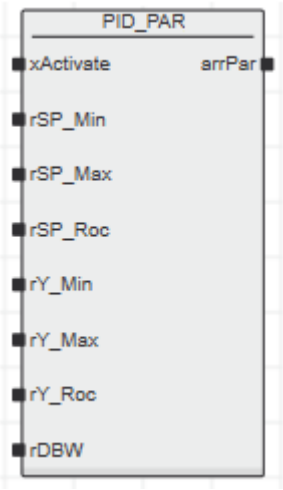
11.8.4 Structure of the output word

	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
wMode (Data type WORD)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAR_OFF	X															
YINV_ON		X														
not used																
D_SEL				X												
P_SEL					X											
not used																
WU_ON							X									
DB_ON								X								
not used																
SPROC_ON										X						
YROC_ON											X					
not used																
not used																
not used																
HALT															X	
RESET																X

11.9 PID_PAR

This PID-type controller attachment module is used in connection with the continuous PID-type controller PID_C to allow an easy input of the parameterization values.
The data exchange with PID-type controller PID_C is performed via the variable “arrPar” of the type ARR10R := ARRAY[0..9] OF REAL.
This function block is not absolutely necessary for the function of PID_C.

11.9.1 Function block call



11.9.2 Input parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rSP_Min	REAL	Set point limitation (lower limit).
rSP_Max	REAL	Set point limitation (upper limit).
rSP_Roc	REAL	Set point limitation (rate of rise).
rY_Min	REAL	Manipulated value limitation (lower limit).
rY_Max	REAL	Manipulated value limitation (upper limit).
rY_Roc	REAL	Manipulated value limitation (rate of rise).
rDBW	REAL	Dead band width before D portion.

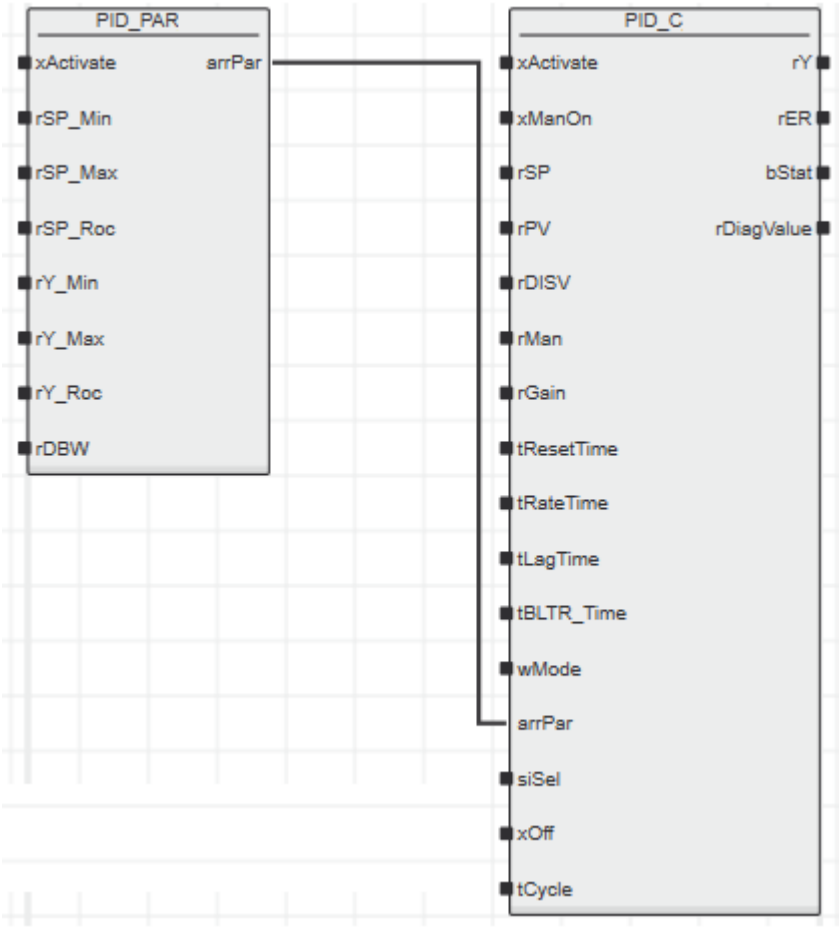
11.9.3 Output parameters

Name	Type	Description
arrPar	ARR10R	Output array.

11.9.4 Encoding of the output array arrPar

arrPar	Variable	Description
arrPar[0]	SP_MN	Set point limitation (lower limit).
arrPar[1]	SP_MX	Set point limitation (upper limit).
arrPar[2]	SP_ROC	Set point limitation (rate of rise).
arrPar[3]	Y_MN	Manipulated value limitation (lower limit).
arrPar[4]	Y_MX	Manipulated value limitation (upper limit).
arrPar[5]	Y_ROC	Manipulated value limitation (rate of rise).
arrPar[8]	DBW	Dead band width before D portion.
arrPar[9]	PAR_ON	Encoding of the internally generated variable PAR_ON PAR_ON :=0.0 Block PID_PAR is deactivated (xActivate = FALSE). PAR_ON :=1.0 Block PID_PAR is activated (xActivate = TRUE).

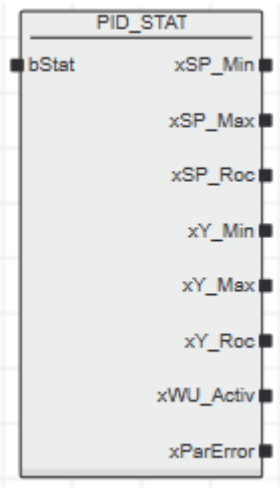
Example



11.10 PID_STAT

This PID-type controller attachment module is used in conjunction with the continuous PID-type controller PID_C for the output of status information (e.g. exceeding of set point or actual value). For this purpose the input bStat of the function block PID_STAT is connected to the output bstat of PID_C. The data exchange with PID_C is performed via a structure of the type BYTE. This function block is not absolutely necessary for the errorfree function of PID_C.

11.10.1 Function block call



11.10.2 Input parameters

Name	Type	Description
bStat	BYTE	Input data byte.

11.10.3 Output parameters

Name	Type	Description
xSP_Min	BOOL	Lower limitation of set point SP.
xSP_Max	BOOL	Upper limitation of set point SP.
xSP_Roc	BOOL	Set point rising limitation active.
xY_Min	BOOL	Lower limitation of manipulated value rY.
xY_Max	BOOL	Upper limitation of manipulated value rY.
xY_Roc	BOOL	Manipulated value rising limitation active.
xWU_Activ	BOOL	Wind-up active (limitation reached)
xParError	BOOL	Parameter error.

11.10.4 Structure of the input word bStat

	x	x	x	x	x	x	x	x
bStat (Data type BYTE)	7	6	5	4	3	2	1	0
QSP_MN	X							
QSP_MX		X						
QSP_ROC			X					
QY_MN				X				
QY_MX					X			
QY_ROC						X		
QWU_ACTV							X	
PAR_ERR								X

11.11 Startup information

The function block C_N (keying controller) can be directly programmed with the parameters known from the system theory. For that, however, profound theoretical knowledge is necessary. Here we must refer to suitable literature.

The function blocks PID_C and PID_R contain a complete PIDTz controller (PID-type controller with a lag time constant in the D portion). For the internal processing a so-called velocity algorithm is used. When calculating the actuating signal, only the differences (e.g. between set point and actual value) between two function block calls are used. Afterwards the calculated actuating increments are summed to one position signal. The algorithm inside the function blocks PID_C and PID_R uses floating-point numbers (REAL). These two function blocks are parameterized with the same parameters as they are usual for compact controllers. Consequently the same setting rules are applicable. Selecting the sampling times short enough allows you to create a quasi-continuous controller using this function blocks.

Additionally two-position, three-position or step controllers can also be realized in conjunction with the function block THREE_C.

Further attachment modules (PID_MODE, PID_PAR or PID_STAT) are provided for the simple input and output of parameter values in conjunction with the function block PID_C.

If it is necessary for an automatic control to use different parameter sets for distinct operating points (different values for GAIN, RESET_T, RATE_T and/or LAG_T) the function blocks PID_ADA and PID_STR are used. If the four operating ranges which can be obtained this way are not enough also a cascading of the function block PID_ADA is possible.

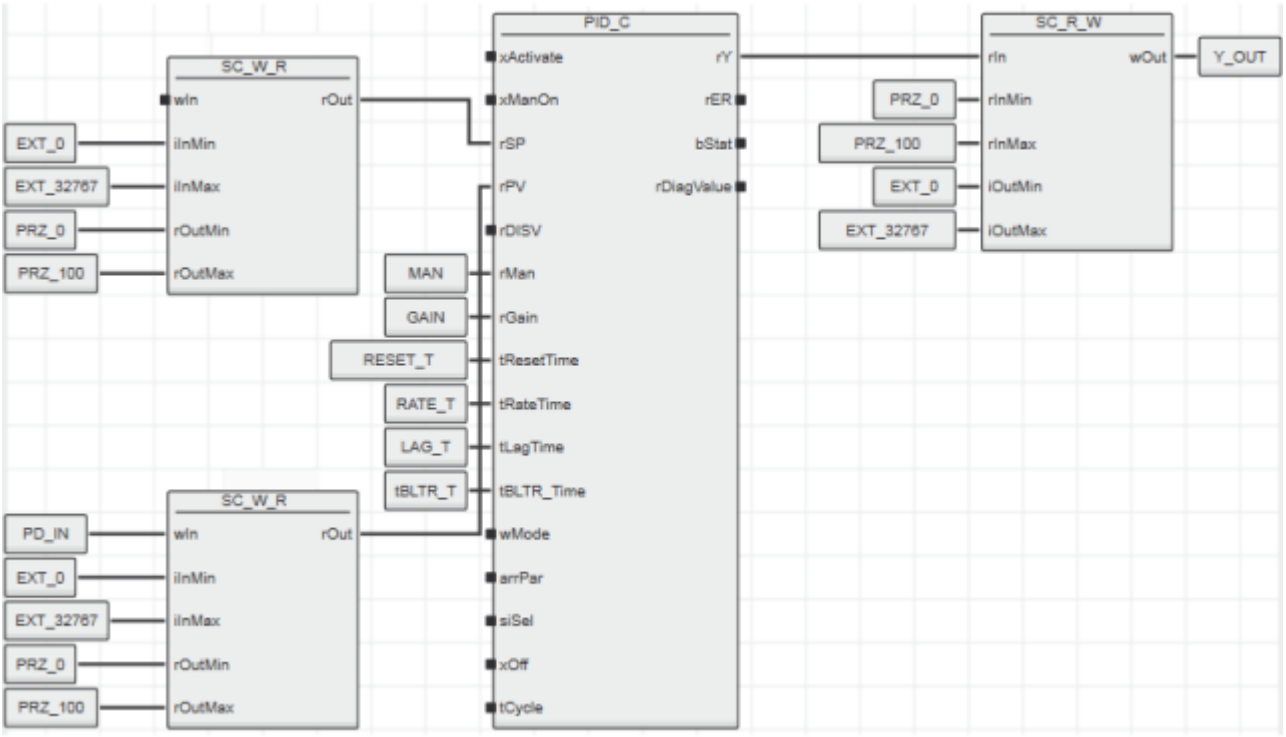
In the field of automatic control engineering a large number of different actuators exist. Depending on the type of actuator the configuration of the controller function blocks differs. Following sections provide an overview of the application of the controller function blocks.

11.11.1 Proportional actuators with continuous actuating signal

Here, for instance the degrees of opening, the rotational angles or the positions are proportional to the actuating signal.

Kind of control	Kind of manipulated variable (Y)	Range of values	Controller structure
Continuous	proportional	Floating-point (REAL): e.g. 0.0..100.0%	PID_C or PID_R

The following figure shows the controller function block PID_C in connection with two scaling function blocks SC_W_R (conversion from peripheral format WORD to REAL) and one scaling function block SC_R_W (conversion from REAL to peripheral format WORD). This example is fully functional although some of the inputs are not wired. Then non-wired inputs of PID_C are pre-assigned with internal default values.



11.11.2 Integrating actuators with three-position actuating signal (with/without position reply) and (with/without limit stops)

For integrating actuators the on-period of the servomotor is proportional to the shifting (I behavior).

Kind of control	Kind of manipulated	Range of values	Controller structure
Step control	–	–	PID_C and THREE_C PID_R and THREE_C
Without position reply	Three-position switching (motor drive)	Up – 0 - Down (Up – 0 - DN)	(YR_ON = FALSE) (DRV_T <> T#0s)
With position reply	Three-position switching (motor drive)	Up – 0 - Down (Up – 0 - DN)	(YR_ON = FALSE) (DRV_T <> T#0s)
With position reply	proportional (position reply)	Floating point (REAL): e.g. 0.0 … 100.0%	–

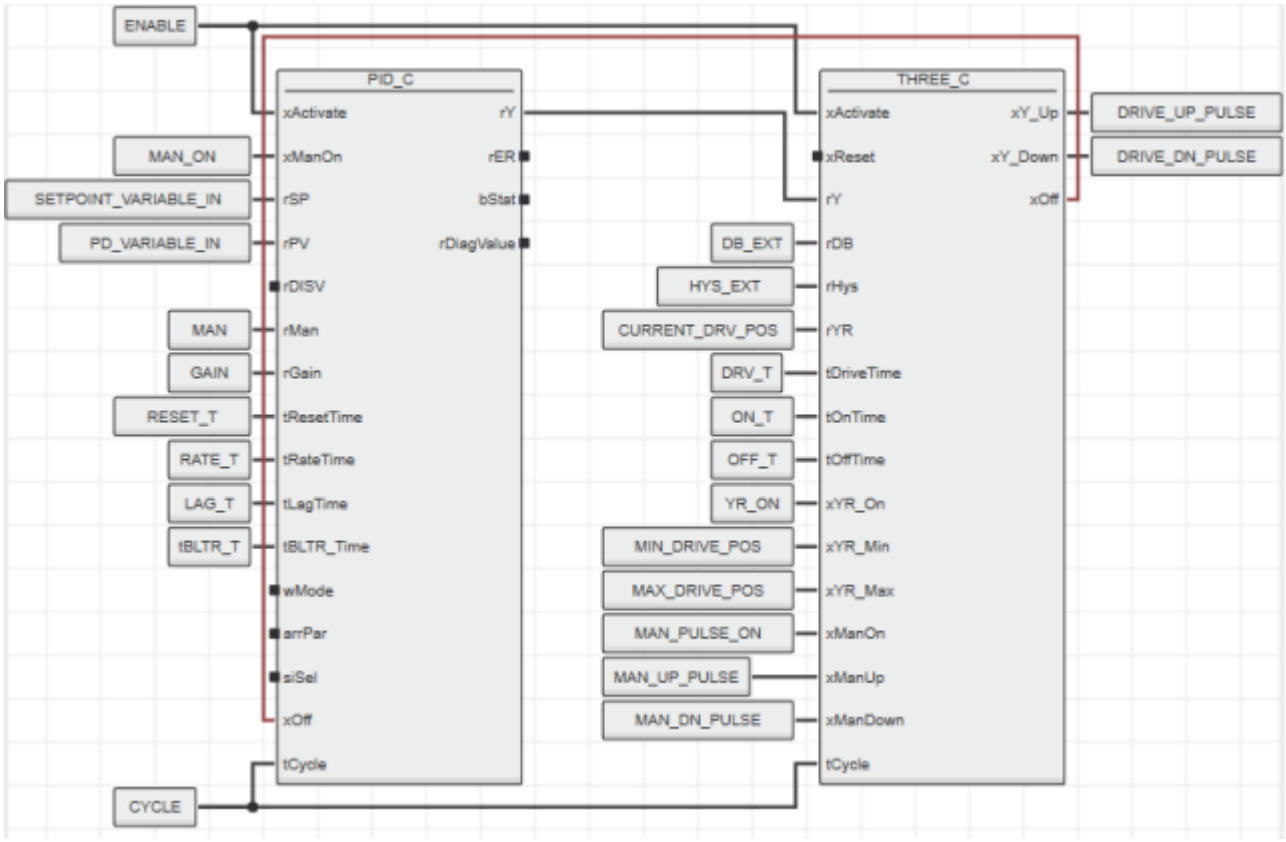
Heating installations are an example for the usage of integrating actuators with a three-position actuating signal. Here, the supply line temperatures of several heating circuits are controlled via three- way or four-way mixer valves which are driven by servomotors. Due to cost reasons this is often done without a position reply or a reply of the limit stops.

If there is no position reply, the exact servomotor actuation time (input DRV_T) must be announced to THREE_C (the servomotor actuation time is the run time between the two limit stops). This is important because the current position of the servomotor must be simulated inside of THREE_C as a reference value. This is the only possible way to achieve an optimum control.

For servomotors with position reply, the reply is performed using a potentiometer. For this purpose the input YR_ON must be set to TRUE at the controller attachment module THREE_C. The motor actuating time must be set to zero (DRV_T = T#0s).

If the servomotor has limit stops, these limit stops should be wired. The output of the corresponding actuating signal is then prevented if one of the limit stops is reached.

The following figure shows the wiring of the function blocks PID_C and THREE_C.


ENABLE
PID_C
xActivate
rY
MAN_ON
xManOn
rER
SETPOINT_VARIABLE_IN
rSP
bStat
PD_VARIABLE_IN
rPV
rDiagValue
rDISV
MAN
rMan
GAIN
rGain
RESET_T
tResetTime
RATE_T
tRateTime
LAG_T
tLagTime
tBLTR_T
tBLTR_Time
wMode
arrPar
siSel
xOff
tCycle
CYCLE
THREE_C
xActivate
xY_Up
DRIVE_UP_PULSE
xReset
xY_Down
DRIVE_DN_PULSE
rY
xOff
DB_EXT
rDB
HYS_EXT
rHys
CURRENT_DRV_POS
rYR
DRV_T
tDriveTime
ON_T
tOnTime
OFF_T
tOffTime
YR_ON
xYR_On
MIN_DRIVE_POS
xYR_Min
MAX_DRIVE_POS
xYR_Max
MAN_PULSE_ON
xManOn
MAN_UP_PULSE
xManUp
MAN_DN_PULSE
xManDown
tCycle

11.11.3 Switching actuators with a bistable actuating signal or a three-position signal

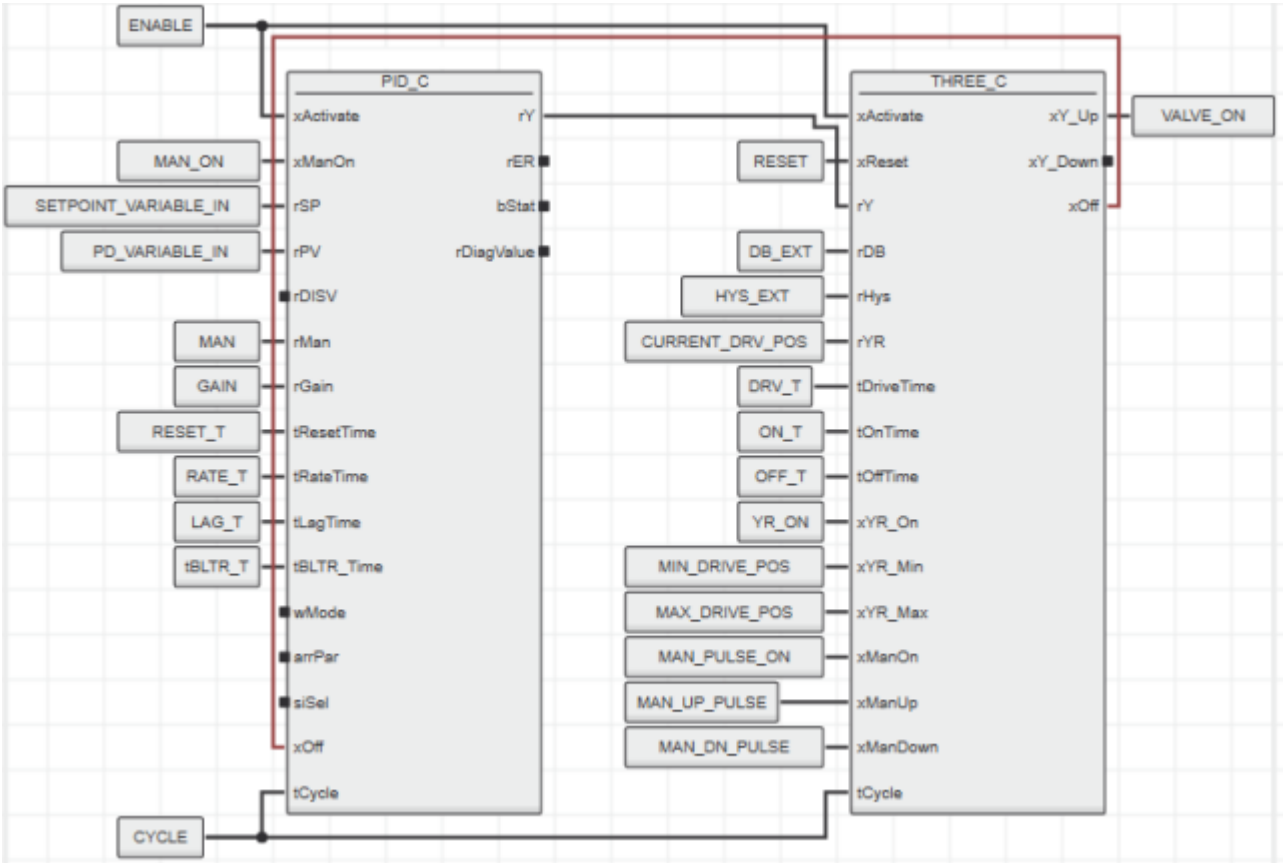
Two-position control

For a two-position control for example the on-period of a valve is proportional to a material stream which has to be controlled. Here, only one output of the three-position step controller is used.

Three-position control

For a three-position control both outputs of the three-position step controller are used. For instance two relays are connected to the outputs. This way a simple air-conditioning system (heating / cooling) can be realized.

Kind of control	Kind of manipulated variable (Y)	Range of values	Controller structure
Two-position-control	1x binary	On – Off // (ON - OFF)	PID_C and THREE_C or PID_R and THREE_C (YR_ON = FALSE) (DRV_T = T#0s)
Three-position-control	2x binary	On – Off // (ON - OFF)	PID_C and THREE_C or PID_R and THREE_C (YR_ON = FALSE) (DRV_T = T#0s)



12 Function blocks for parameterizing and special functions

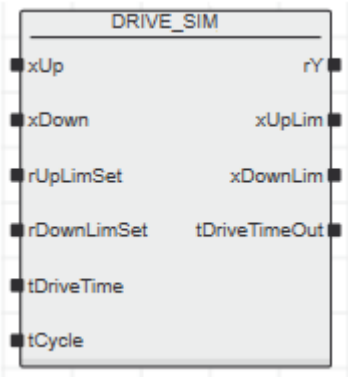
Further function blocks are available:

Function block	Description	Version	Supported articles	License
DRIVE_SIM	Simulation of a servo motor.	1	-	none
SEND50	Storage of 50 successive values.	1	-	none
RCV50	Display of the 50 determined values in FBD.	1	-	none
ADA_PAR	Clear parameter transfer in FBD.	1	-	none
AG_PAR	Clear parameter transfer in FBD.	1	-	none
C_N_PAR	Simplified parameter setting for the function block C_N.	1	-	none
POL_PAR	Simplified parameter setting for the function block POLN_N.	1	-	none
MODE_PAR	Simplified parameter setting for the function block PID_MODE.	1	-	none
PAR_PAR	Simplified parameter setting for the function block PID_PAR.	1	-	none

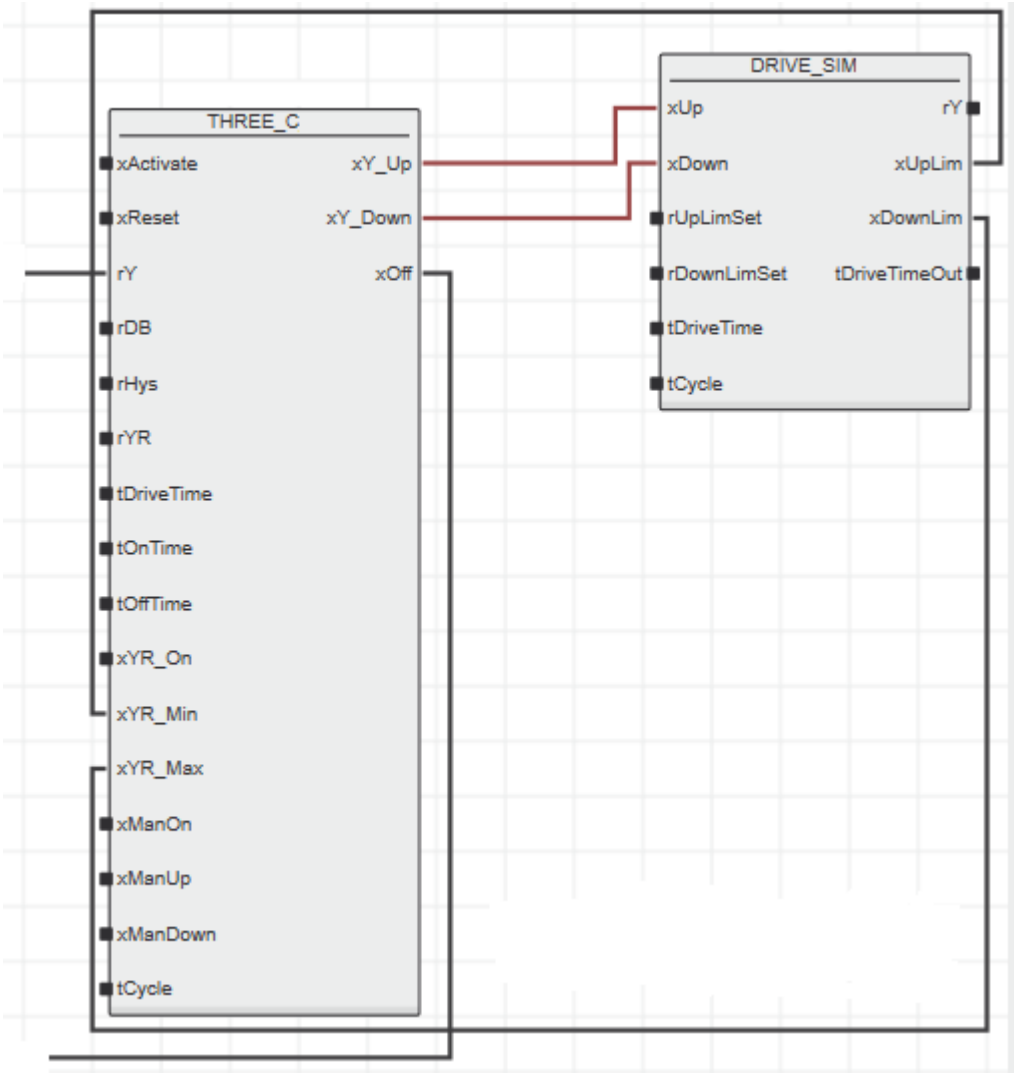
12.1 DRIVE_SIM

This function block simulates a servomotor (i.e. an actuator with integrating behaviour) with an upper limit stop (Q_UP_LIM) and a lower limit stop (Q_DN_LIM) as well as the position reply (rY). The values to be parameterized are two limit stops and the motor actuating time which is to be simulated (tDriveTime). DN_LIM_SET 0.0 (%) and UP_LIM_SET 100(%) are recommended.

12.1.1 Function block call



12.1.2 Block connection



12.1.3 Input parameters

Name	Type	Description
xUp	BOOL	Upper limit stop.
xDown	BOOL	Lower limit stop.
rUpLimSet	REAL	Upper limit set.
rDownLimSet	REAL	Lower limit set.
tDriveTime	TIME	Motor actuating time
tCycle	TIME	Cycle time

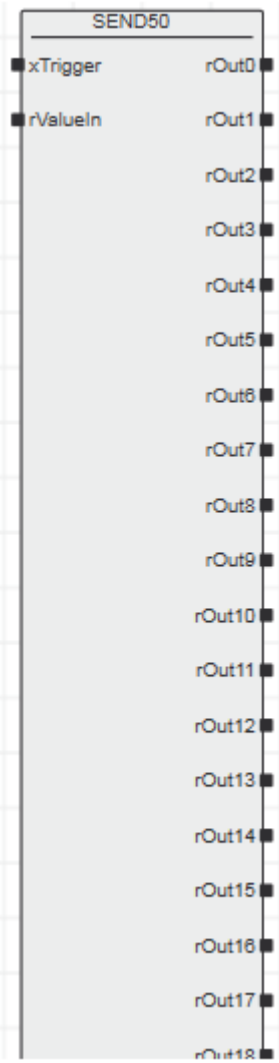
12.1.4 Output parameters

Name	Type	Description
rY	REAL	Position reply.
xUpLim	BOOL	rY > rUpLimSet
xDownLim	BOOL	rY < rDownLimSet
tDriveTimeOut	TIME	Motor actuating timeout.

12.2 SEND50

The function block SEND50 is used to store 50 successive values after the activation of the memory function by the condition ENABLE = TRUE. The time interval of the storage process between the individual values corresponds exactly to the particular cycle time difference.

12.2.1 Function block call



12.2.2 Input parameters

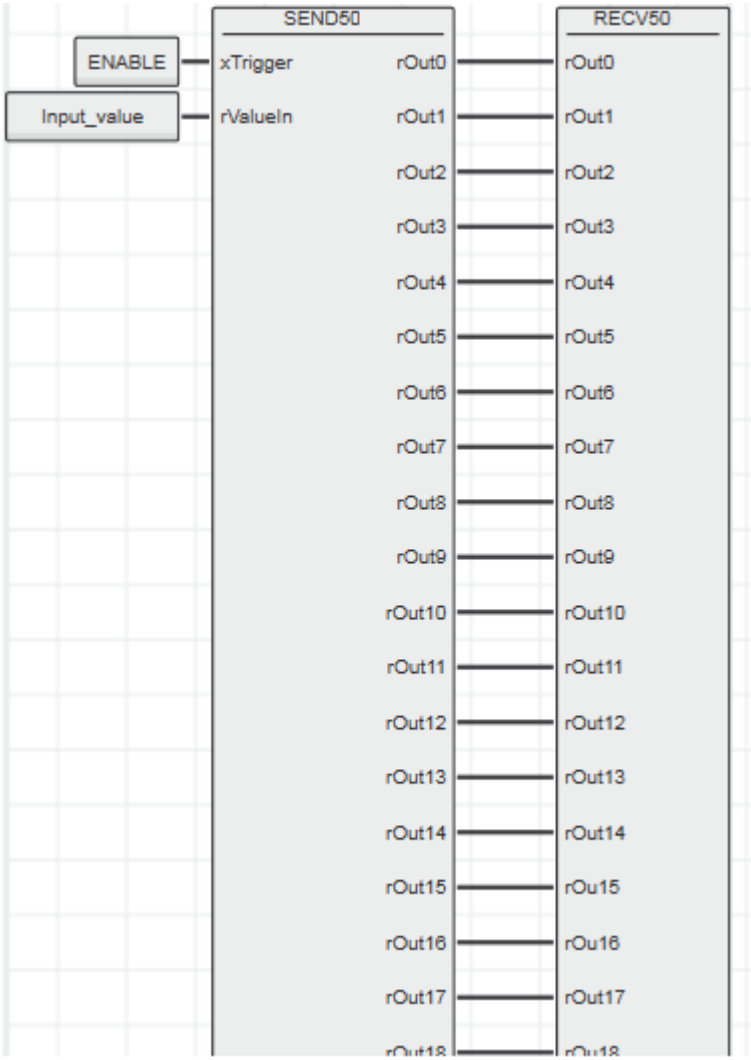
Name	Type	Description
xTrigger	BOOL	TRUE: Activates function block.
rValueIn	REAL	Input value.

12.2.3 Output parameters

Name	Type	Description
rOut0..rOut50	REAL	Stored input value.

12.2.4 Example

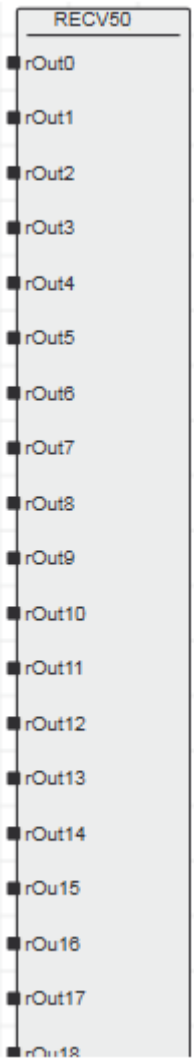
Function block connection



12.3 RECV50

The function block RECV50 is intended for the connection to the outputs of the function block SEND50. This enables the determined numerical values, e.g. a step response, to be displayed in FBD very easily.

12.3.1 Function block call

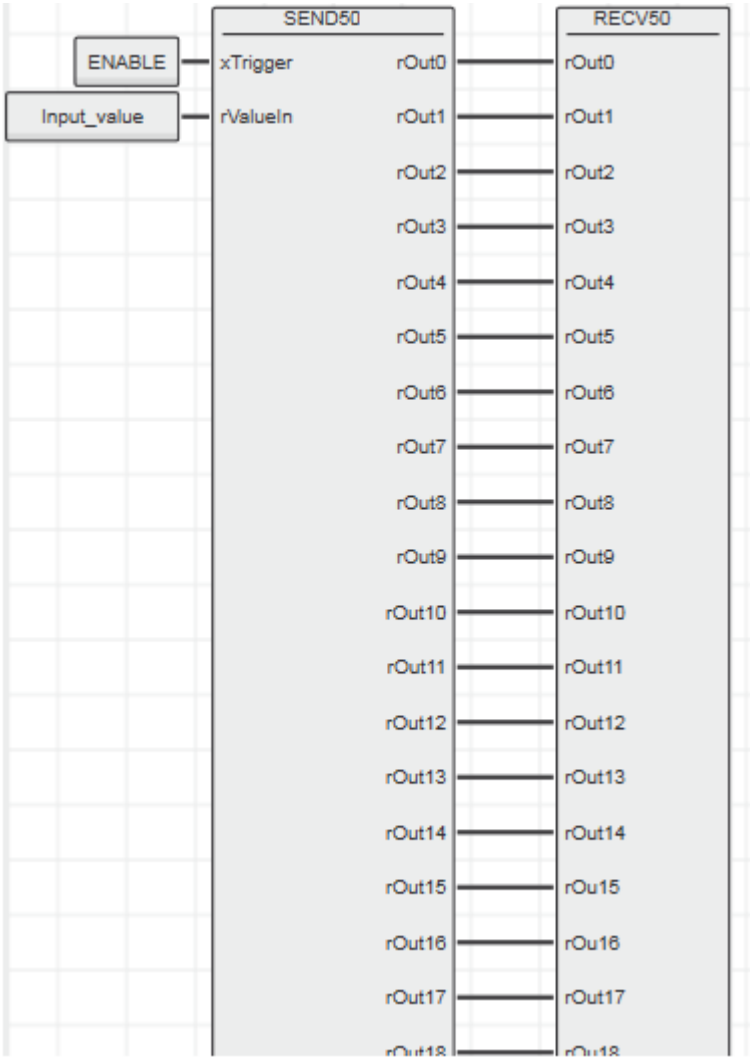


12.3.2 Output parameters

Name	Type	Description
rOut0..rOut50	REAL	Transmitted stored input values of the SEND50 function block.

12.3.3 Example

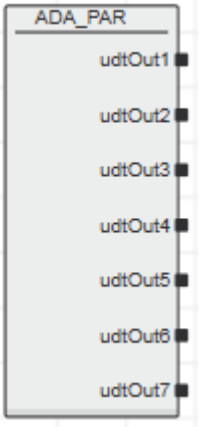
Function block connection



12.4 ADA_PAR

The function blocks ADA_PAR and AG_PAR are primarily intended to acilitate a clear parameter transfer in the IEC programming language FBD. The following figure illustrates the application.

12.4.1 Function block call



12.4.2 Output parameters

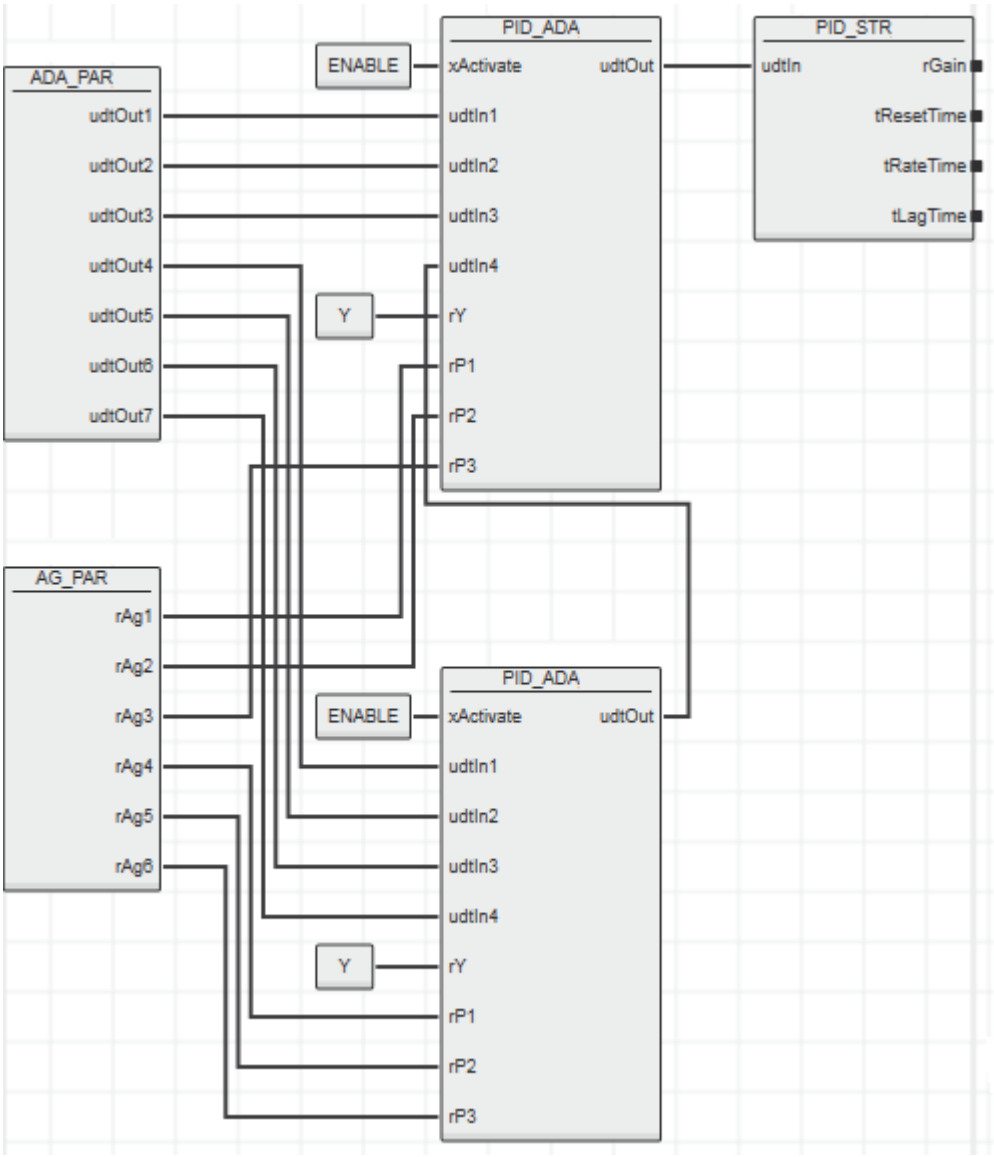
Name	Type	Description
udtOUT1..udtOUT7	Control_UDT_Parameter_V1	Parameter structure.

12.4.3 Example

Parameterization

```
udtOut1.rGain := 1.658;
udtOut1.tResetTime := T#6.343s;
udtOut1.tRateTime := T#1.043s;
udtOut1.tLagTime := T#0.985s;

udtOut2.rGain := 2.2;
udtOut2.tResetTime := T#100ms;
udtOut2.tRateTime := T#200ms;
udtOut2.tLagTime := T#520ms;

udtOut3.rGain := 2.4;
udtOut3.tResetTime := T#100ms;
udtOut3.tRateTime := T#200ms;
udtOut3.tLagTime := T#530ms;

udtOut4.rGain := 2.5;
udtOut4.tResetTime := T#100ms;
udtOut4.tRateTime := T#200ms;
udtOut4.tLagTime := T#540ms;

udtOut5.rGain := 2.7;
udtOut5.tResetTime := T#100ms;
udtOut5.tRateTime := T#200ms;
udtOut5.tLagTime := T#550ms;

udtOut6.rGain := 2.8;
udtOut6.tResetTime := T#100ms;
udtOut6.tRateTime := T#200ms;
udtOut6.tLagTime := T#560ms;

udtOut7.rGain := 2.9;
udtOut7.tResetTime := T#100ms;
udtOut7.tRateTime := T#200ms;
udtOut7.tLagTime := T#570ms
```

Function block connection



12.5 AG_PAR

The function blocks ADA_PAR and AG_PAR are primarily intended to facilitate a clear parameter transfer in the IEC programming language FBD. The following figure illustrates the application.

12.5.1 Function block call

12.5.2 Output parameters

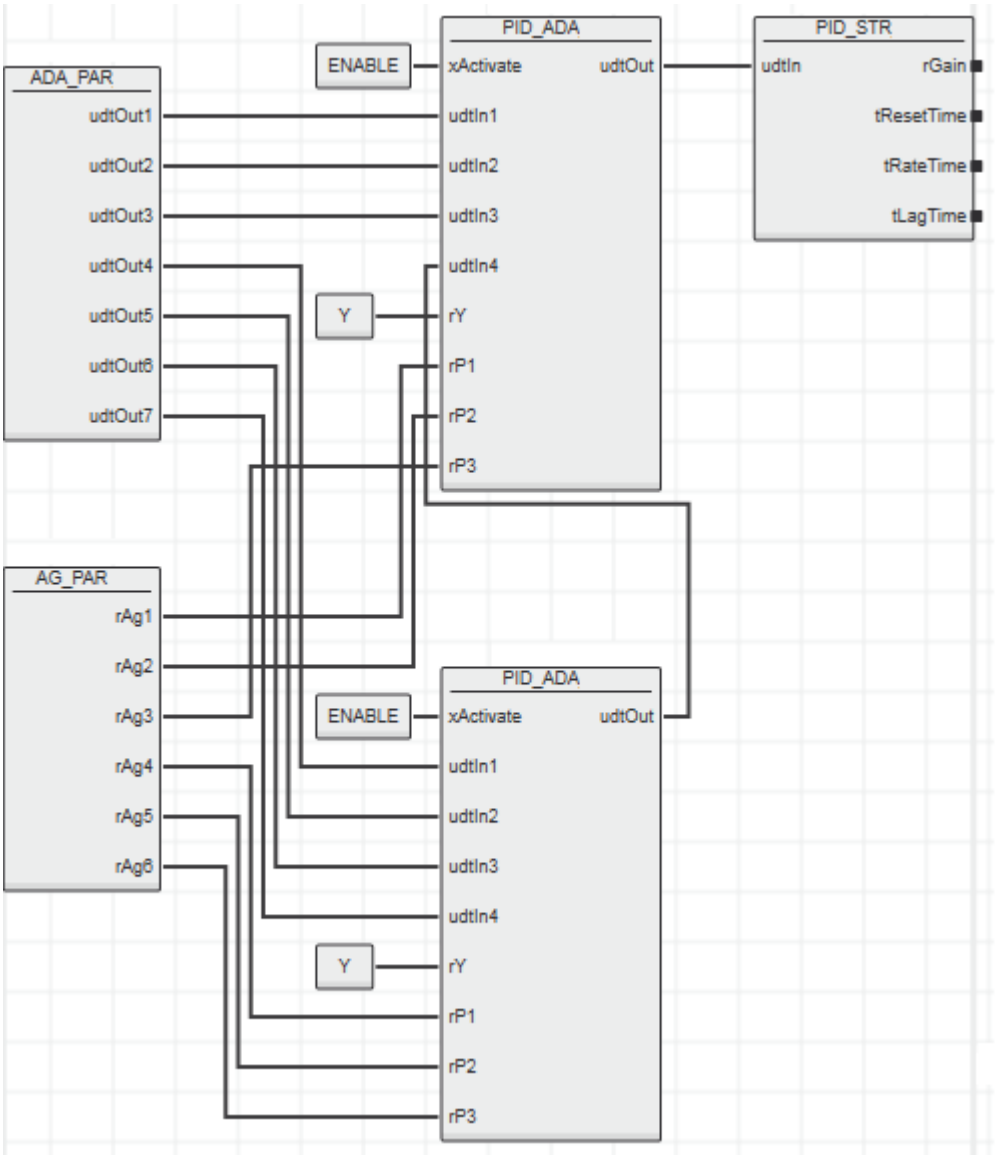
Name	Type	Description
rAg1..rAg5	REAL	Set values for flip over points of the PID_ADA function block.

12.5.3 Example

Parameterization

```
rAg1 := 100.0;
rAg2 := 1000.0;
rAg3 := 1500.0;
rAg4 := 2000.0;
rAg5 := 2500.0;
rAg6 := 3000.0;
```

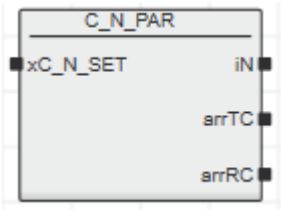
Function block connection



12.6 C_N_PAR

The function block C_N_PAR serves primarily for a simplified parameter transfer to the function block C_N in the IEC programming language FBD. In order to perform a parameterization in your project using this function block, you have to copy this function block into your project. This is required because changes are only performed in the project and not in the library.

12.6.1 Function block call



12.6.2 Input parameters

Name	Type	Description
xC_N_SET	BOOL	TRUE: Activates function block.

12.6.3 Output parameters

Name	Type	Description
iN	INT	Order N <= 9.
arrTC	ARR10R	Numerator coefficient.
arrRC	ARR10R	Denominator coefficient.

12.6.4 Example

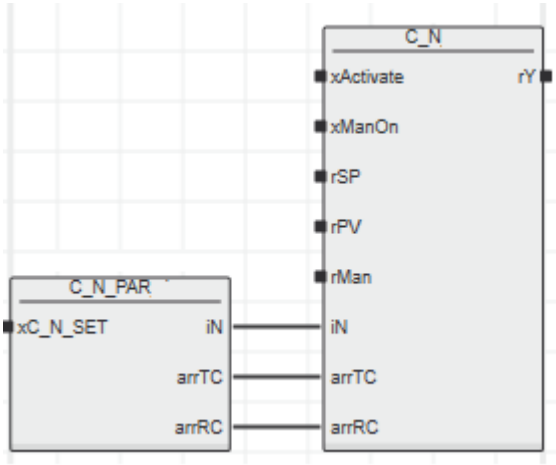
Parameterization

N :=5;

arrTC[0] := 3.86350542;
arrTC[1] := 0.0;
arrTC[2] := -5.57258394;
arrTC[3] := 3.14237012;
arrTC[4] := -0.43329160;
arrTC[5] := 0.0;
arrTC[6] := 0.0;
arrTC[7] := 0.0;
arrTC[8] := 0.0;
arrTC[9] := 0.0;

arrRC[0] := 0.0;
arrRC[1] := -0.11848690;
arrRC[2] := -0.42898162;
arrRC[3] := -0.40488611;
arrRC[4] := -0.04764536;
arrRC[5] := 0.0;
arrRC[6] := 0.0;
arrRC[7] := 0.0;
arrRC[8] := 0.0;
arrRC[9] := 0.0;

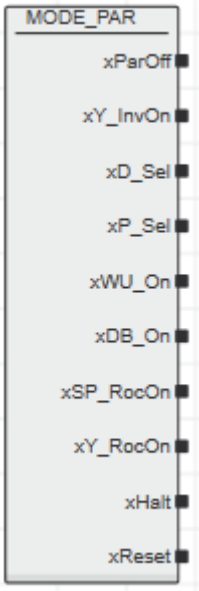
Function block connection



12.7 MODE_PAR

The function block MODE_PAR serves primarily for a simplified parameter transfer to the function block PID_MODE in the IEC programming language FBD. In order to perform a parameterization in your project using this function block, you have to copy this function block into your project. This is required because changes are only performed in the project and not in the library.

12.7.1 Function block call



12.7.2 Output parameters

Name	Type	Description
xPar_OFF	BOOL	FALSE: The connected PID_C newly reads-in the wMode and arrPar control commands as well as rGain, tResetTime, tRateTime and tLagTime during every cycle. TRUE: The parameter values are not read-in again.
xY_InvOn	BOOL	TRUE: rY is inverted: rY := - rY.
xD_Sel	BOOL	FALSE: D element obtains control deviation (rSP-rPV). TRUE:: D element obtains process variable (rPV) (Selector switch for switching on the D portion).
xP_Sel	BOOL	FALSE: P element obtains control deviation (rSP-rPV). TRUE:: P element obtains process variable (rPV) (Selector switch for switching on the D portion).
xWU_On	BOOL	TRUE: “WIND-UP” active.
xDB_On	BOOL	TRUE: Dead band located before D portion is switched on.
xSP_RocON	BOOL	TRUE: Set point rising limiter switched on.
xY_RocON	BOOL	TRUE: Manipulated value rising limiter switched on.
xHalt	BOOL	TRUE: The last manipulated value is held (in automatic mode).
xReset	BOOL	Rising edge: Resets the function block.

12.7.3 Example

Parameterization

xParOff := FALSE;

xY_InvOn := FALSE;
xD_Sel := FALSE;

xP_Sel := FALSE;

xWU_On := TRUE;

xDB_On := TRUE;

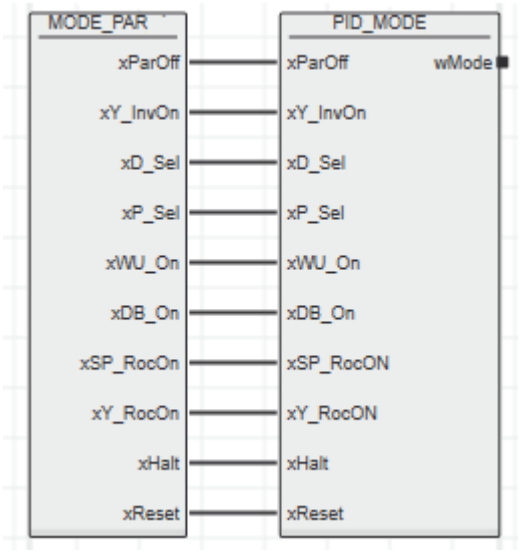
xSP_RocON := FALSE;

xY_RocON := FALSE;

xHalt := FALSE;

xReset := FALSE;

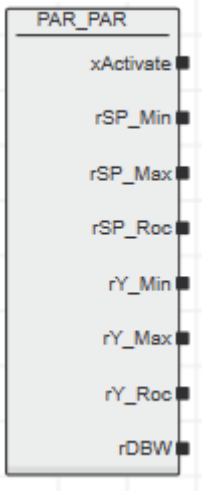
Function block connection



12.8 PAR_PAR

The function block PAR_PAR serves primarily for a simplified parameter transfer to the function block PID_PAR in the IEC programming language FBD. In order to perform a parameterization in your project using this function block, you have to copy this function block into your project. This is required because changes are only performed in the project and not in the library.

12.8.1 Function block call



12.8.2 Output parameters

Name	Type	Description
xActivate	BOOL	Rising edge: Activates the function block. FALSE: Deactivates the function block.
rSP_Min	REAL	Set point limitation (lower limit).
rSP_Max	REAL	Set point limitation (upper limit).
rSP_Roc	REAL	Set point limitation (rate of rise).
rY_Min	REAL	Manipulated value limitation (lower limit).
rY_Max	REAL	Manipulated value limitation (upper limit).
rY_Roc	REAL	Manipulated value limitation (rate of rise).
rDBW	REAL	Dead band width before D portion.

12.8.3 Example

Parameterization

xActivate := TRUE;

rSP_Min := -100.0;

rSP_Max := 100.0;

rSP_Roc := 10.0;

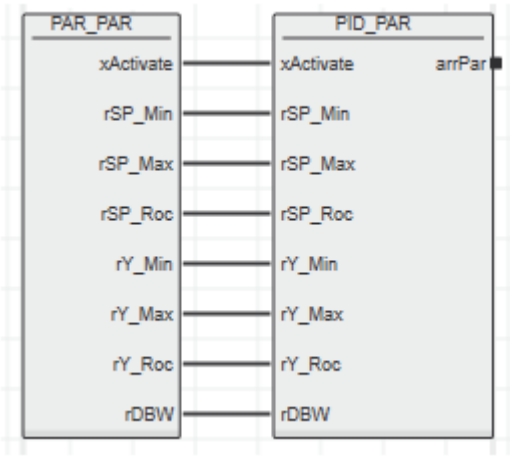
rY_Min := 20.0;

rY_Max := 80.0;

rY_Roc := 10.0;

rDBW := 0.3;

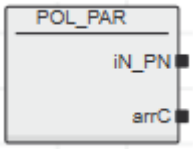
Function block connection



12.9 POL_PAR

The function block POL_PAR serves primarily for a simplified parameter transfer to the function block POLN_N in the IEC programming language FBD. In order to perform a parameterization in your project using this function block, you have to copy this function block into your project. This is required because changes are only performed in the project and not in the library.

12.9.1 Function block call



12.9.2 Output parameters

Name	Type	Description
iN_PN	INT	Degree of the polynomial (0 <= iN <= 9).
arrC	ARR10R	Polynomial coefficients.

12.9.3 Example

Parameterization

N_PN := 4;

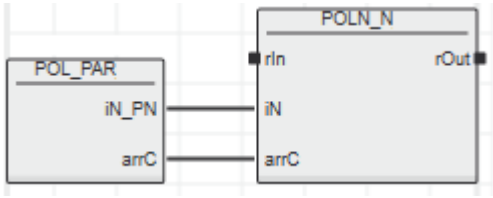
C[0] := 2.0;

C[1] := 1.5;

C[2] := 0.5;

C[3] := 0.2;

Function block connection



13 Appendix

13.1 Conventions for identifiers

Abbreviation	Meaning	Description
SP	Set point	Set point, reference input value
PV	Process variable	Process (controlled) variable
Disv	Disturbance variable	Disturbance variable
Man	Manual value	Manual (actuating) value
Alarm	Alarm	Alarm
Warn	Warning	Warning
ER	Error signal	Control system deviation
Y	Manipulated variable (control function blocks)	Manipulated variable (output quanity of the function blocks PID_C / PID_R)
OUT/_OUT	Non binary output	General output (e.g. type REAL)
Q	Binary output	General output of the type BOOL)
_ROC	Rate of change	Gradient (rate of change)
_LIM	Limited value	Limitation of the absolute value
_INT	Internal value	Internal quantity
_EXT	External value	External quantity
_ON	TRUE = ON	TRUE corresponds to function ON (boolean)
_OFF	TRUE = OFF	TRUE corresponds to function OFF (boolean)
_ACTV	TRUE = ON (activated)	TRUE corresponds to function ON / active (boolean)
_MN	Minimal value (BOOL)	Minimum value is reached (boolean)
_MX	Maximal value (BOOL)	Maximum value is reached (boolean)
_SEL	Binary selection (BOOL)	Binary selection (boolean)
Sel	Non binary selection	General selection input
Min	Minimal value (e.g. REAL)	Minimum value (e.g. type REAL)
Min	Maximal value (e.g. REAL)	Maximum value (e.g. type REAL)
Activate	Activate function block	Activate function block / function (boolean)
_T	Time	Attachment to identify the quanity “TIME”
Cycle	Cycle time (Function block)	Sampling time of the selected function block
Reset	Reset value	Reset input
HALT	Hold value or state	Holding
LagTime	Lag time	Lag time (e.g. LAG1ST or additional time constant Tz in the D portion (PID_C / PID_R)
RateTime	Rate time	Derivative rate time Tv
ResetTime	Reset time	Reset time Tn
Gain	Gain	Gain, proportional constant Kr

13.2 User defined variables

To simplify the parameterizing and the transfer of values between the function blocks it proved to be necessary to define special variables. These variables (USERDEFINED VARIABLES) are shown below.

Variables of the type ARRAY

VARIABLES DEFINITION	used in function block:
ARR10R = ARRAY[0..9] OF REAL	PID_C, PID_PAR, POLN_N, C_N
ARR20R = ARRAY[0..19] OF REAL	POLG_N, DELAY20
ARR100R = ARRAY[0..99] OF REAL	DELAY100

Variables of the type STRUCT

VARIABLES DEFINITION	used in function block
TYPE STR : STRUCT GAIN : REAL; RESET_T : TIME; RATE_T : TIME; LAG_T : TIME; END_STRUCT; END_TYPE	PID_ADA PID_STR

13.3 Data types

```
TYPE
Control_ARR_R_0_9 : ARRAY[0..9] OF REAL;
Control_ARR_R_0_19 : ARRAY[0..19] OF REAL;
Control_ARR_R_0_99 : ARRAY[0..99] OF REAL;

    (* Struct for PID_ADA, PID_STR and ADA_PAR function block outputs *)
    Control_UDT_Parameter_V1 : STRUCT
        rGain       : REAL; (*proportional constant Kr*)
        tResetTime  : TIME; (*reset time Tn*)
        tRateTime   : TIME; (*derivative rate time Tv*)
        tLagTime    : TIME; (*additional time constant Tz in the D portion*)
    END_STRUCT;
END_TYPE
```

14 Support

For technical support please contact your local PHOENIX CONTACT agency

at https://www.phoenixcontact.com

Owner:

PHOENIX CONTACT Electronics GmbH
Business Unit Automation Systems
System Services
Library Services