PID

The library util.library provides the following PID (Proportional–Integral–Derivative) controller function block:

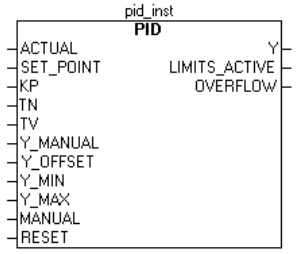

Fig. 18: PID in FBD

Unlike the PD controller, this function block contains a further REAL input TN for the readjusting time in sec (for example "0.5" for 500 msec).

NOTICE!

The PID controller itself measures the elapsed time between two calls, however with a maximum accuracy of milliseconds. This might lead to rough running in case of short cycle times: For example in case of a cycle time of 1ms the PID sometimes might measure 2 ms, sometimes 0 ms. So if possible, for such cases better use PID_FIXCYCLE, where the cycle time can be set precisely. See PID_FIXCYCLE

NOTICE!

Consider that only at a start, a reset or at a change down the controller parameters as used in the manual mode get applied.

Inputs of the function block:

Variable	Data type	Description
ACTUAL	REAL	Current value of the controlled variable
SET_POINT	REAL	Desired value, command variable
KP	REAL	Proportionality coefficient, unity gain of the P-part
		This value mustn't be 0, otherwise the function block will not perform any calculations.
TN	REAL	Reset time, reciprocal unity gain of the I-part
		Given in seconds, for example "0.5" for 500 msec
		This value must be >0, otherwise the function block will not perform any calculations. The smaller TN is, the more the integral part gets included in the value of the manipulated variable. The more TN increases, the less this will be the case.
TV	REAL	Derivative action time, unity gain of the D-part
		Given in seconds, for example "0.5" for 500 msec
Y_MANUAL	REAL	Defines output value Y in case of MANUAL = TRUE
Y_OFFSET	REAL	Offset for the manipulated variable Y
Y_MIN, Y_MAX	REAL	Lower resp. upper limit for the manipulated variable Y. If Y exceeds these limits, output LIMITS_ACTIVE will be set to TRUE and Y will be kept within the prescribed range. This control will only work if Y_MIN<Y_MAX.
MANUAL	BOOL	If TRUE, manual operation will be active, that is the manipulated value will be defined by Y_MANUAL.
RESET	BOOL	TRUE resets the controller; during reinitialization Y = Y_OFFSET.

Outputs of the function block:

Variable	Data type	Description
Y	REAL	Manipulated value, calculated by the function block (see below)
LIMITS_ACTIVE	BOOL	TRUE indicates that Y has exceeded the given limits (Y_MIN, Y_MAX).
OVERFLOW	BOOL	TRUE indicates an overflow (see below)

Y_OFFSET, Y_MIN and Y_MAX serve for transformation of the manipulated variable within a prescribed range.

MANUAL can be used to switch to manual operation; RESET can be used to re-initialize the controller.

In normal operation (MANUAL = RESET = LIMITS_ACTIVE = FALSE) the controller calculates the **controller error e** as difference from SET_POINT – ACTUAL, generates the derivation with respect to time ∂e/ ∂t and stores these values internally.

The output, that is the **manipulated variable Y** unlike the PD controller contains an additional integral part and is calculated as follows:

$$Y = KP \cdot (e + 1/TN \int e dt + TV \cdot \partial e/\partial t) + Y\_OFFSET$$

So besides the **P-part** also the current change of the controller error (**D-part**) and the history of the controller error (**I-part**) influence the manipulated variable.

The PID controller can be easily converted to a PI-controller by setting TV=0.

Because of the additional integral part, an overflow can come about by incorrect parameterization of the controller, if the integral of the error **e** becomes to great. Therefore for the sake of safety a BOOLean output called **OVERFLOW** is present, which in this case would have the value TRUE. This only will happen if the control system is instable due to incorrect parameterization. At the same time, the controller will be suspended and will only be activated again by re-initialization.

As long as the limitation for the manipulated variable (Y_MIN u. Y_MAX) is active, the integral part will be adapted, like if the history of the input values had automatically effected the limited output value. If this behaviour is not wanted, the following workaround is possible: Switch off the limitation at the PID controller (Y_MIN>=Y_MAX) and instead apply the LIMIT operator (IEC standard) on output value Y (see an example in the figure below).

NOTICE!

It is not necessary to readjust the controller parameters (KP, TN, TV) if the cycle time changes.

Temperature control with PID and LIMIT

See in the following figure a simple example of using the PID module for temperature control and in combination with the LIMIT operator. The input of the actual temperature is simulated by giving a constant value via `ActualTemperature`.

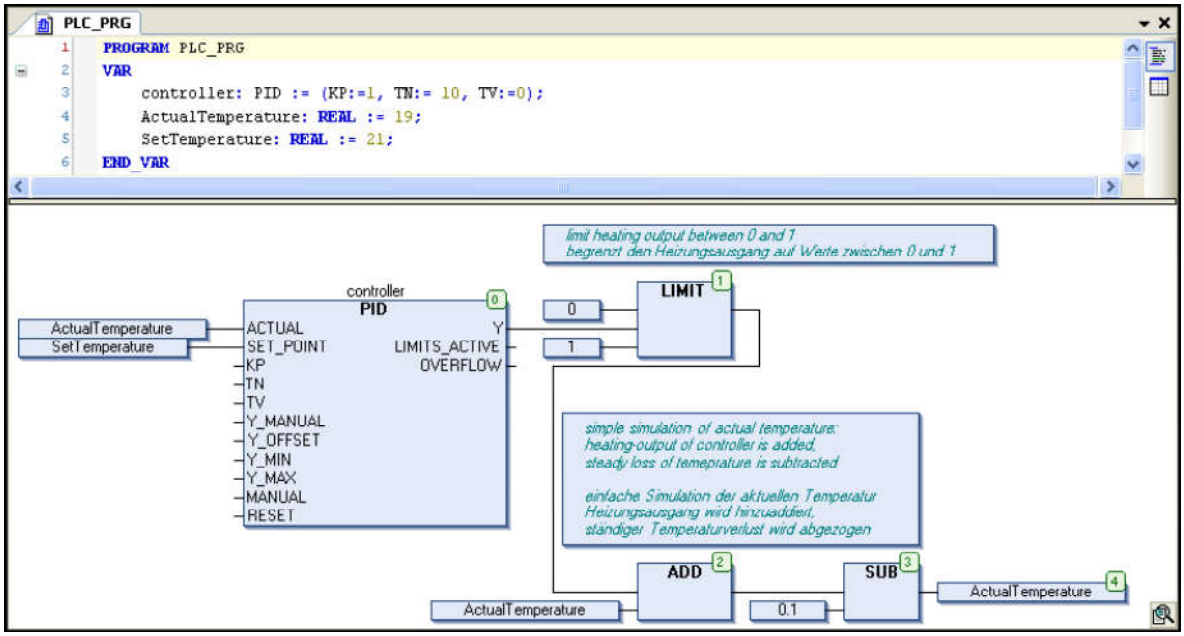

Fig. 19: PID example