PID controller in heating and cooling modes
Project for PR200-24.2

Project overview

The example explains the use of a PID controller in heating and cooling modes. The project contains 2 data processing blocks and 3 screens.

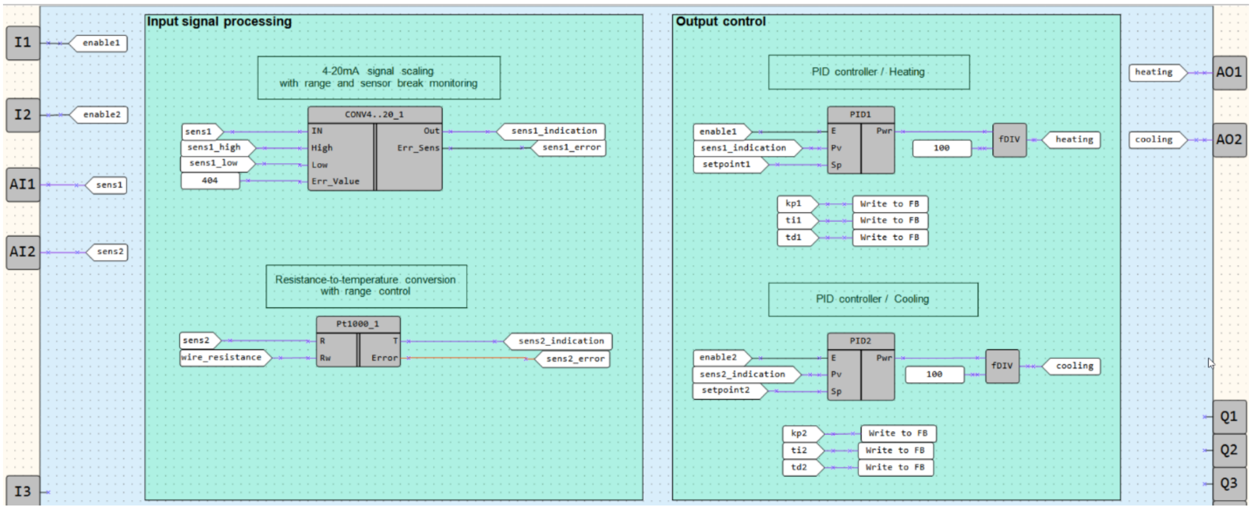

Fig. 1. Program workspace

Data processing blocks:

- Input signal processing
- Output control

Screens:

- *Sensors*
- *Setpoints*
- *PID coefficients*

Table 1. Device inputs/outputs

Name	Type	Description
I1	BOOL	Heater enable (NO contact, latching)
I2	BOOL	Cooler enable (NO contact, latching)
AI1	REAL	1st Temperature sensor (4-20 mA)
AI2	REAL	2nd Temperature sensor (RTD)
AO1	REAL	Heater
AO2	REAL	Cooler

Table 2. Project variables

Name	Type	Description
heating	REAL	Heater / control signal
cooling	REAL	Cooler / control signal
enable1	BOOL	Heater enable
enable2	BOOL	Cooler enable

sens1_error	BOOL	1st sensor / error
sens2_error	INT	2nd sensor / error
kp1	REAL	Heater / proportional gain
kp2	REAL	Cooler / proportional gain
td1	REAL	Heater / differential time
td2	REAL	Cooler / differential time
ti1	REAL	Heater / integral time
ti2	REAL	Cooler / integral time
sens1	REAL	1st sensor / signal (temperature 4-20 mA)
sens1_indication	REAL	1st sensor / signal (temperature °C)
sens1_high	REAL	1st sensor / upper limit
sens1_low	REAL	1st sensor / lower limit
setpoint1	REAL	Heater / setpoint
sens2	REAL	2nd sensor / signal (temperature Ω)
sens2_indication	REAL	2nd sensor / signal (temperature °C)
setpoint2	REAL	Cooler / setpoint
wire_resistance	REAL	2nd sensor / wire resistance (Ω)

Input signal processing

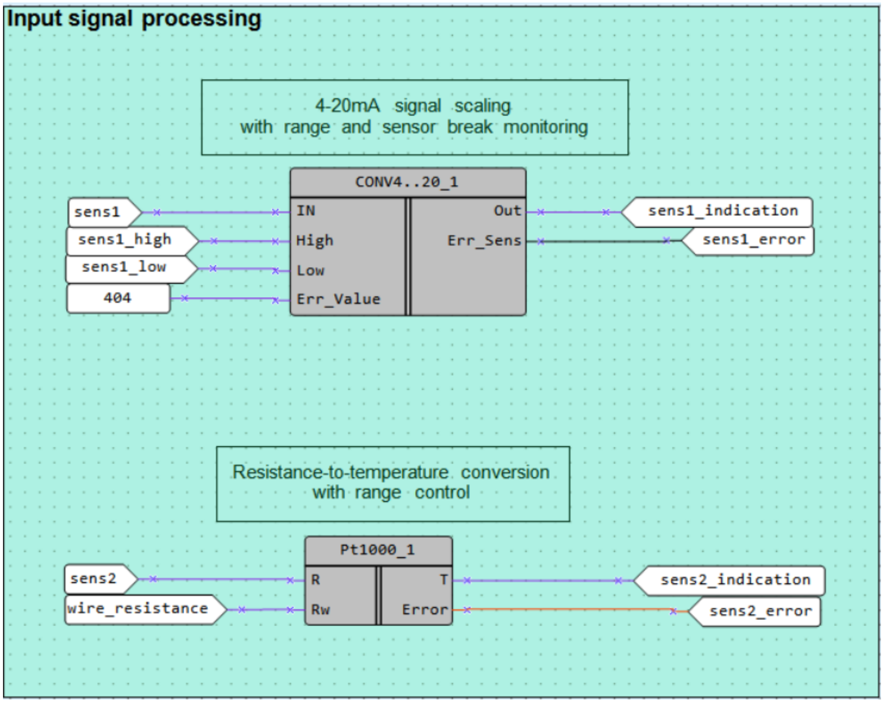

Fig. 2. Input signal processing

The sensor connected to input AI1 is a current sensor with a 4-20 mA output signal. The output signal is converted into temperature using the *CONV4..20* macro, which scales the current signal.

The constant applied to the input *Err_Value* is the value on the macro output *Out* in case of error.

The sensor connected to the AI2 input is a PT1000 resistance thermometer. The output signal is converted to temperature with *PT1000* macro that scales the resistance signal. It also provides the wire resistance compensation over *Rwire* input.

Output control

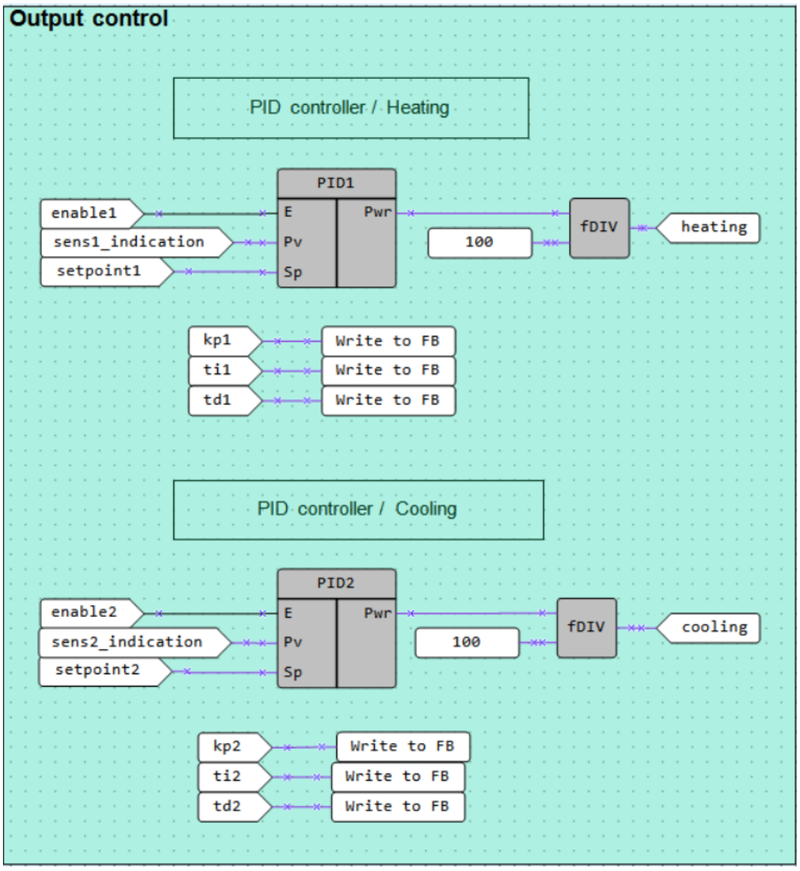

Fig. 3. Output control

The PID temperature control is provided by two *PID* macros. One of them works in heating mode, the other in cooling mode.

The controllers can be enabled/disabled over two latching push-buttons connected to the *I1* and *I2* inputs.

The controller characteristic parameters:

- Setpoint
- Proportional gain
- Differential time
- Integral time

can be read on the device display and changed using the function buttons.

The entered values are written to the *PID* FB over *WriteToFB* blocks.

Since the output signal from the PID controller is the power represented in percent, and the analog output of the device perceives the signal in the range of 0...1, the value of the output power of the PID controller must be divided by 100.

Screens

Table 3. Function buttons

Function buttons	Action
⩔ (down)	Scroll down through screen rows
⩓ (up)	Scroll up through screen rows
ALT + ⩔ (down)	Switch to the next screen
ALT + ⩓ (up)	Switch to the previous screen
ESC	Switch to the first screen

Initially, the screen *Sensors* is displayed (Fig. 4). It shows the status of the both sensors (*normal/error*) and the measured temperature.

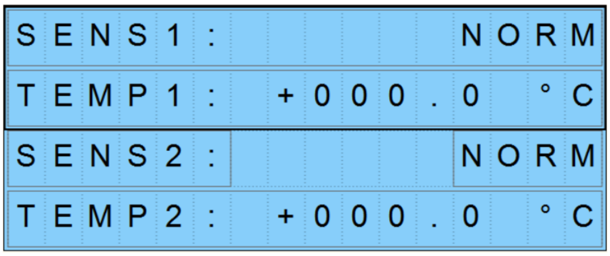

Fig. 4. Screen *Sensors*

The next screen *Setpoints* (Fig.5) shows the parameters *Setpoint* for both controllers.

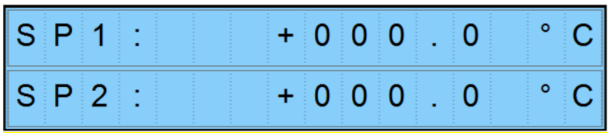

Fig. 5. Screen *Setpoints*

The third screen shows the coefficients of PID controllers (Fig. 6).

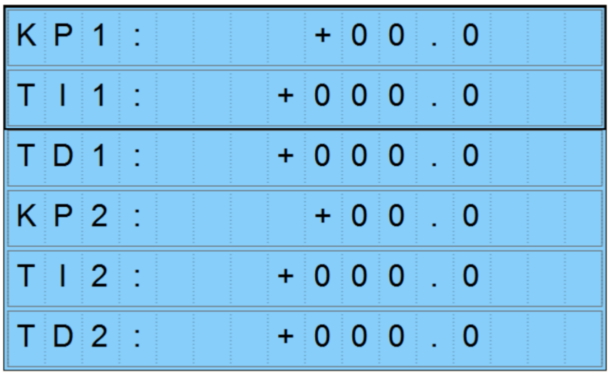

Fig. 6. Screen *PID coefficients*