SIEMENS

SIMATIC

PID Self-Tuner V5

Contents

Manual

This manual is part of the documentation package with the order number:

6ES7860-4AA01-0YX0

12/99
C79000-G7076-C825
Edition 03

Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

Warning
indicates that death, severe personal injury or substantial property damage can result if proper precautions are not taken.

Caution!
indicates that minor personal injury or property damage can result if proper precautions are not taken.

Note
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only qualified personnel should be allowed to install and work on this equipment Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Please note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are trademarks of Siemens AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Disclaimer of Liability
We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Siemens AG
Bereich Automatisierungs- und Antriebstechnik
Geschaeftsgebiet Industrie-Automatisierungsysteme
Postfach 4848, D- 90327 Nuernberg

Siemens Aktiengesellschaft

6ES7860-4AA01-0YX0

Excellence in Automation & Drives: Siemens

Contents

1 Getting Started

Aims

You want to control a temperature process which includes heating and cooling with Standard PID Control V5 and want the PID controller parameters to be determined online with PID Self-Tuner V5.

Requirements

The following requirements must be met:

- You have an S7-300/400 station consisting of the power supply unit, a CPU, an analog input module and a digital output module.
- STEP 7 (≥ V3.2) is installed on your programming device.
- The programming device is connected to the CPU.
- Standard PID Control V5 is installed on the programming device.

Installing PID Self-Tuner V5 on the programming device

Procedure:

- Make a copy of each original diskette.
- Install the software from this copy by calling up the installation program SETUP.EXE on diskette 1.

Creating a new project

Create a new project in the SIMATIC Manager and insert a new SIMATIC 300 or SIMATIC 400 station. Hardware configuration is used to configure the station in accordance with your modules.

Copying a working example to your project

In the SIMATIC Manager you can now copy the working example 5 ”Ex5_simple_p” from TunExStd into your project. Download your project into the CPU and familiarize yourself with the example. You can now simulate various scenarios such as the first adaption, adapting, cooling identification, controlling with the zone control system or structure selection.

Wiring the process for the manipulated and process variables

Wire the sensor which measures the process value to be controlled to the analog input module. In your project you must assign the appropriate peripheral input word PIWx to the PV_PER when calling PID_CP (interconnection between the Self-Tuner and the controller). The PVPER_ON parameter must be set to TRUE. Now check your process value in the curve recorder or in a VAT.

Wire the digital output module to the semiconductor relay that controls the heating and the contactor for the cooling. In your project you must lay the outputs QPOS_P and QNEG_P to the corresponding output bit (Ax.y) when calling PID_CP. Check the heating and cooling of the process during manual operation. You can set MAN_ON to TRUE and specify individual manipulated values in MAN in the variable declaration table VAT_SIMPLE.

Process analysis

If you fear that your process may have extremely long dead times, apply a manipulated value step from 0% to, for example, 30% to the heating and use the curve recorder of Standard PID Control to record a step response of the process variable. Check the area of application of the Self Tuner.

Startup/Test

Before starting the first adaption function let the process reach any steady state. This is ideally the ambient temperature in a cooled state.

Set the parameter ADAPT1ST to TRUE in the variable declaration table VAT_SIMPLE. Then enter a positive setpoint step (new setpoint is greater than the old one) at the SP input of the Self-Tuner. Note that at the start a step size > MIN_STEP is required. Set the controller to automatic mode (MAN_ON of the Self-Tuner = FALSE). Controller optimization of the Self-Tuner is now active. Observe the status display at the PHASE parameter. After the process settles to the operating point (PHASE = 4), evaluate the diagnostic display at the STATUS_H parameter. If the adaptation is successful you can test the control response of the controller first set on the basis of smaller setpoint steps at the operating point or by feedforwarding faults.

If the desired result is produced, you can save the controller parameter with SAVE_PAR.

If necessary, you can start further optimization by using ADAPT_ON.

Check for which cases the response to setpoint changes can be improved:

- With or without zone control operation or
- With or without structure selection.

Note

In order to activate the structure changeover function the structure segmentation has to be de-activated (IDB_TUN_CON_P.controller.PFDB_SEL = IDB_TUN_CON_P.controller.DFDB_SEL = FALSE).
If necessary, these bits also have to be reset in the restart routine of the FB_CON_P.

If optimization is unsuccessful, you can use UNDO_PAR to reload the previous controller parameters. These are then active automatically.

If the optimization does not produce satisfactory controller parameters, you can change them by hand and activate them by setting LOAD_PAR.

You can now remove all the parts for simulating the process (FB_PROC_HCP and IDB_PROC_HCP) from your project.

2 Area of application

Advantages and fields of application

With PID Self-Tuner V5 you can trim the PID controller of the following SIMATIC S7 and SIMATIC C7 products to a self-tuning PID controller.

- Standard PID Control (FB PID_CP and FB PID_ES)
- Modular PID Control (FB PID, FB LMNGEN_C and FB LMNGEN_S)
- Controller modules FM 355 and FM 455 (FB PID_CS)
- PID Control (integrated in STEP 7, FB CONT_C, FB CONT_S and FB_PULSEGEN)
- CPU 314 IFM (SFB CONT_C, SFB CONT_S and SFB PULSEGEN)

The Self-Tuner can be used particularly well for

- Temperature control systems, but also for
- Level control systems and
- Flow rate control systems.

In the case of flow rate control systems a distinction has to be made between systems where only the control valve itself is to be controlled or whether there is a time lag process after the control valve. The Self-Tuner cannot be used for the simple control of a control valve (refer to the section on "Processes with a control valve with integral action").

Process requirements

The process has to meet the following requirements:

- Stable time-lag asymptotic transient response
- Time lags not too large
- Adequate linear response across a sufficiently large operating range
- Little disturbance in temperature processes
- Adequate quality of the measured signals in the sense of a sufficiently high signal-to-disturbance ratio.
- Process gains not too high

Transient response

The process must have a stable time-lag asymptotic transient response.

After a step in the manipulated variable the process variable must change to a steady state as shown in the figure below. This therefore excludes processes that have an oscillating response without control as well as processes that are not self-regulating (integrator in the process).

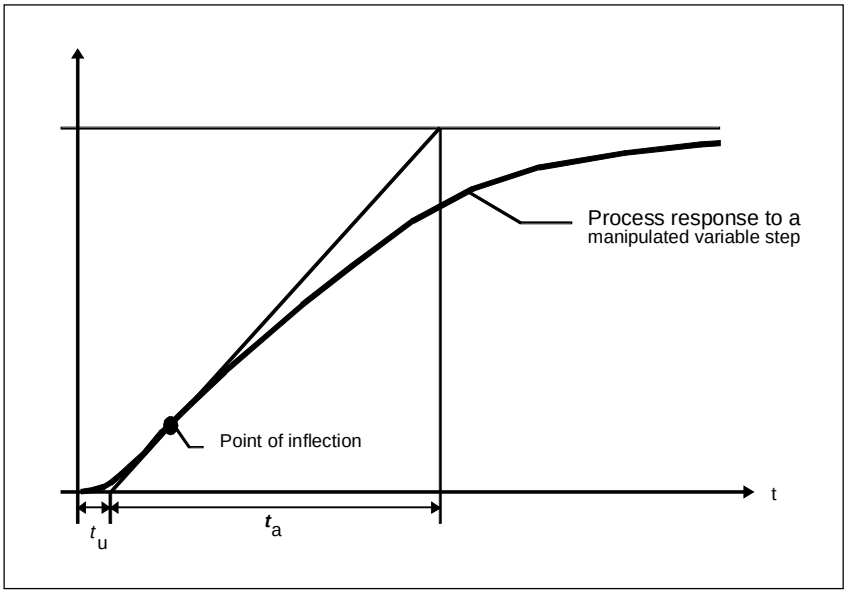


Figure: Process response

Time lags

The process may not have excessive time lags.

The range of application can be specified based on the ratio of the delay time tu and settling time ta. The time lag also encompasses any existing dead times. The first adaption or adaptation is designed for the following range:

tu < 1/10 ta,

Most temperature processes lie within this range. A usable PI controller can still be designed for the range of particularly high time lags:

1/10 ta < tu < 1/3 ta

However, the duration of the learning phase can be increased significantly and overshoots can also occur during the learning phase. In such processes it is advisable to specify the setpoint somewhat lower for the first adaption than the desired operating point and to observe the status bits of the Self-Tuner exactly.

Linearity and operating range

The process must have a sufficiently linear response across a sufficiently large operating range.

This means that non-linear effects within this operating range can be ignored both during identification and during normal controlled operation. However, it is possible to re-identify the process when changing the operating point when the adaptation process is carried out again close to the new operating point and provided that the non-linearity does not occur during adaptation.

If certain static non-linearities (for example valve characteristics) are known, it is always advisable to compensate from the beginning with a polygon in order to linearize the process response.

Disturbances in temperature processes

Disturbances such as thermal transfer to neighboring zones or heat gains or losses during a change in the equipment status must not influence the overall temperature process to any great extent. If appropriate, adaptation at the operating point may be necessary. For example, all the zones of an extruder have to be heated up simultaneously in order to optimize the zones.

Note

- Faults in measuring transducers such as a wire break or a short circuit - are not recognized by the Self-Tuner and have to be eliminated outside the Self-Tuner. If a fault in the measuring transducer occurs during optimization, abort it and repeat it. If a fault in the measuring transducer occurs during closed-loop control, the controller should be switch briefly to manual mode and then back to automatic mode in order to compare its integral action.

Quality of the measured signals

The quality of the measured signals has to be sufficient, in other words the signal-to-disturbance ratio has to be sufficiently high.

Process gains

The process gains may not be too high. Normalization of the process variables is not required. The process gain K can thus include physical units, such as:

$K = \Delta PV / \Delta LMN$, [K]= °C/ %%

The final controller designed is based on a calculation of the process gain K and can thus in principle compensate any values of K. However, K is initially unknown during the learning phase and overshoots cannot be avoided at extreme combinations of gain and learning step changes. The overshoot can also be reduced by reducing the parameter LHLM_TUN in the heating process or by increasing LLLM_TUN in the cooling process.

Processes with a control valve with integral action

In processes with control valves with an integral action further requirements have to be fulfilled in addition to those specified above:

The motor actuating time of the control valve must be smaller than the time required to find a point of inflection after a step change in the manipulated variable (also refer to the process response figure). If this is not the case, the process involved is often a flow rate control system in which only the control valve is effective as the dominating process action. It is therefore not advisable to use the Self-Tuner. You can then set the PI step controller by hand in accordance with the following rule of thumb:

GAIN = 1, TI = Control valve actuating time.

Note

With a step controller without a position feedback signal it must be possible within the process that the control valve is opened completely in order to determine the motor actuating time.

In the case of a step controller with position feedback you can use the LHLM_TUN parameter to determine how far the valve is opened.

3 Description of the function blocks

3.1 FB "TUN_EC"

How the function works

Interconnect the FB TUN_EC block with a continuous PID controller (has already been prepared in the example). The FB TUN_EC disposes of the following functions to control or optimize the controller:

- **Manual mode**
  The Self-Tuner forces the controller into manual mode and transfers its manual parameters to it.
- **Manual/Automatic changeover with prediction-control manipulated-value selection**
  After the controller has been optimized successfully, the Self-Tuner can change over the controller from manual mode to automatic mode by feedforwarding a prediction-control manipulated value in such a manner that the process settles into the setpoint value rapidly. This type of changeover is not smooth. You can also set smooth changing over at the Self-Tuner.
- **Controller optimization**
  The process is first identified when optimizing the controller parameters. A controller is designed based on the process characteristics. The new system parameters are calculated on this basis and these are activated for the controller.
  - **First adaption**
    The first adaption can only be carried out in case of a positive manipulated variable step. It is usually used during the first startup or in case of a serious change in the controlled system characteristics.
  - **On-line adaption**
    The controller can be optimized subsequently on-line during a positive setpoint step when the operating point or the process response is changed.
  - **Cooling identification**
    For control systems which operate with two opposing actuating elements (for example in case of temperature control: heating and cooling actuating elements) the Self-Tuner determines the ratio of the process gains under the influence of a second actuating element in the case of a negative manipulated variable step (staying at the example of a temperature control: under influence of the cooling actuating element).

- **Optimization of the response to setpoint changes**
  The controller is designed for optimum disturbance response. The resultant "stringent" parameters would result in overshoots of 10% to 20% of the step height at setpoint steps.

  The **control zone** represents an initial remedy. In order to optimize the start-up control response at large setpoint steps the Self-Tuner calculates a control zone range during optimization. The controller is active within the zone. Outside the zone the Self-Tuner controls the controller with the maximum or minimum manipulated variable of the controller.

  The **structure segmentation** represents a further remedy, in which the proportional or derivative action is moved into the feedback loop. A change in the setpoint thus only acts on the integral action of the controller.

  If structure segmentation is not possible for the controller or if the control zone fails, the following function can be activated in order to optimize the response to setpoint changes:

  Structure changeover:
  Temporary de-activation of the integral action or prediction-control specification of the manipulated variable at large positive setpoint steps. This function is not effective when the control zone is activated.
- **Saving the controller parameters**
  If you judge the current controller parameters to be utilizable, you can save these before a manual change in parameters specially provided in the instance DB of the Self-Tuner. During controller optimization the saved parameters are overwritten by the values valid before optimization.
- **Restore saved controller parameters**
  This function allows the controller parameters last saved to be re-activated for the controller.
- **Changing the controller parameters**
  If you want to set the controller parameters yourself for the controller, you can assign these through special parameters in the instance DB of the Self-Tuner and transfer them to the controller by means of this function.

Remark: The following section assumes a general temperature process with active heating and cooling for the description of the process.

Modes

Control zone

The Self-Tuner operates with a control zone. This means that the controller is controlled by means of the following algorithm:

- If the error signal is greater than CON_ZONE, the system is heated with the maximum output.
- If the error signal is smaller than (-CON_ZONE) * RATIOFAC, the system is cooled with the maximum output.

During the first setting process or during on-line adaption a control zone CON_ZONE is determined by the Self-Tuner and is activated in case of a suitable process type: CONZ_ON = TRUE. In closed-loop control you can modify the control zone or de-activate it completely (with CONZ_ON = FALSE).

When the control zone is activated (CONZ_ON = TRUE), the structure changeover function STRUC_ON is not effective.

Note

No control zone is determined during a separate controller design for processes with a high time lag, when STATUS_D = 3 after adaptation.

Before activating the control zone by hand ensure that the control zone range is not set too small. If the control zone range is set too small, oscillations occur during the course of the manipulated variable and of the process variable.

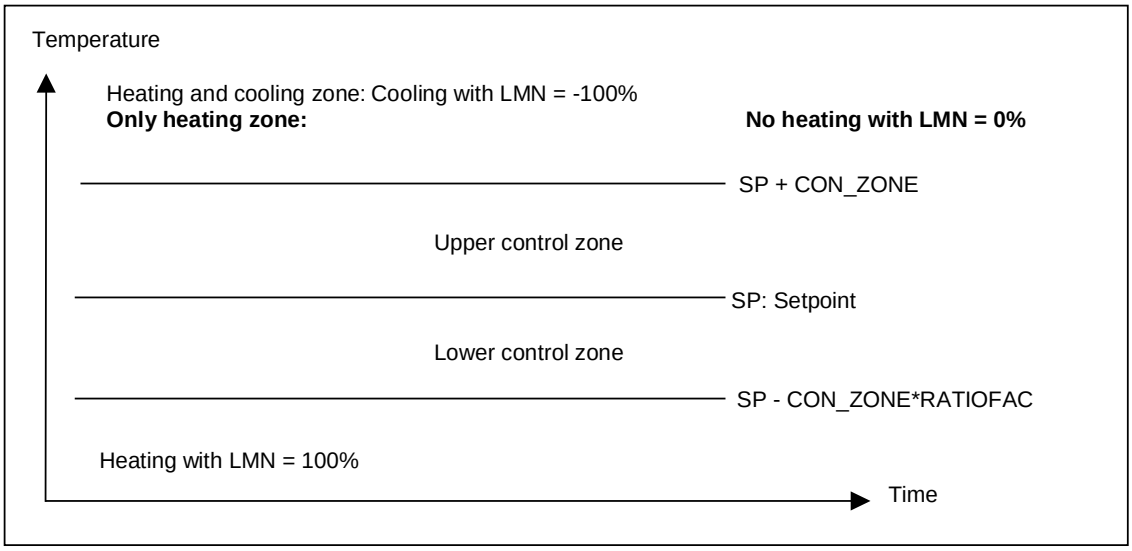


Figure: Control zone

Structure segmentation

Structure segmentation is a function of the controller. In the case of closed-loop control with structure segmentation the proportional or derivative action lies in the feedback loop, so that a change in the setpoint only acts on the integral action of the controller. Manipulated variable steps do not occur during setpoint steps - the manipulated variable takes a ramp course. The disturbance response is not changed as a result.

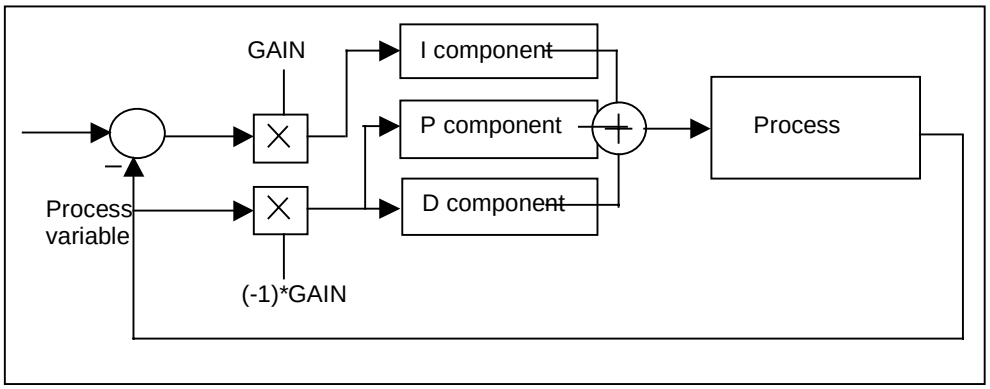


Figure: Closed-loop control with structure segmentation

In the case of closed-loop control without structure segmentation the proportional and derivative actions lie in the error signal just like the integral action. You can select a structure changeover when you are in this operating mode (STRUC_ON=1).

Closed-loop control with structure segmentation is possible in the following control packages:

- Standard PID Control
- Modular PID Control
- FM 355/455
- PCS7 controller

Note

Only closed-loop control without structure segmentation is possible for the simple controller PID Control of Step7-Basis and in the CPU 314 IFM. The structure changeover function should only be selected if there are no other alternatives.

Structure changeover at proportional and derivative actions in the error signal

The structure changeover is de-activated in the default setting. It should only be selected if no other alternatives such as, for example, control zone or structure segmentation, are available.

When the control zone is activated (CONZ_ON = TRUE), the structure changeover function STRUC_ON is not effective.

You can activate the structure changeover function by setting STRUC_ON = TRUE. The block automatically selects between two controlling measures depending on the total loop gain (product of the process gain and the controller gain):

- Integral action temporarily de-activated, P(D) control:

  In the case of a positive setpoint step ≥ MIN_STEP the integral action of the controller is de-activated temporarily and the gain is increased slightly, so that

a pure P(D) controller is used. When the setpoint value is approached the integral action is re-activated and the gain is reduced again (PHASE = 5 in the operation of the Self-Tuner).

- Temporary prediction-control specification of the manipulated variable:

  Processes with a high time lag cannot be controlled well with a P(D) control system. After a positive setpoint step ≥ MIN_STEP the manipulated variable required for the new setpoint in steady state is output as a constant. When the setpoint value is approached the system changes back bumplessly to PI or PID controller operating mode. This results in a slow action which is, however, free of overshooting (PHASE = 5 in the operation of the Self-Tuner).

Note

If you do not achieve good results (slow transient response) with the structure changeover function at positive setpoint steps (for example heating processes) , you can de-activate the structure changeover function by setting STRUC_ON = FALSE, provided that small overshoots may occur.

Modes of the structure	**CONZ_ON**	**PFDB_SEL, DFDB_SEL** [1)]	**STRUC_ON**
Activate the control zone	TRUE	Any	Any
Activate the structure segmentation	Any	TRUE	FALSE
Activate the structure changeover	FALSE	FALSE	TRUE

[1)] Controller parameters PFDB_SEL, DFDB_SEL: These parameters do not exist in the simple controllers of STEP7 CONT_C. This corresponds to the setting PFDB_SEL = DFDB_SEL = FALSE (without structure segmentation).

Information on selecting the operating mode

- The Self-Tuner calculates a control zone and activates it automatically, if the ratio of the two time constants of the process f = TM_LAG1 / TM_LAG2 > 10.
- In the case of steps within the control zone the structure segmentation which is selected by default acts, if allowed by the controller.
- De-activate the structure segmentation function if you want small setpoint steps within the control zone to be settled as fast as possible and are willing to accept small overshoots.
  Example with numbers: Control zone 20°C, then overshoots occur of approx. 10%-20% of the step size (< 20°C), i.e. less than 2-4°C.
- The structure segmentation function prevents overshoots at setpoint steps of all sizes, even if the process does not permit a control zone or if you have de-activated it. However, the rise times are slightly slower than in the control zone.
- Activate the structure changeover function if the process does not allow a control zone and the controller does not allow a structure segmentation. The structure changeover function may not be active at the same time as the structure segmentation function.

Manual mode

If you set the input MAN_ON to TRUE, the output QMAN_ON is set to TRUE and the output MAN_OUT to MAN. This changes the PID controller over to manual operation (PHASE = 5 in the operation of the Self-Tuner).

Manual mode takes priority over all other operating modes.

A first adaption, on-line adaption or a cooling identification is aborted. When manual operation is de-activated (MAN_ON = FALSE), the controller changes over to automatic mode (PHASE = 4 in the operation of the Self-Tuner) and controls with the existing controller parameters.

Manual/automatic changeover with prediction-control manipulated-value selection

When changing over to automatic mode you can specify whether you want to change over bumplessly from the last manual value or with bumps with a (prediction-control) manipulated variable calculated by the Self-Tuner. If PRED_ON = TRUE (default setting) the changeover is carried out with bumps. The changeover with bumps is only effective after an optimization and ensures rapid settling into the desired setpoint, in particular for a PI control system.

Controller optimization

Optimization modes in the Self-Tuner	ADAPT1ST	ADAPT_ON	COOLID_ON
Deleting of all the old process information and requesting of a first adaption	TRUE	Any Is set by the Self-Tuner if ADAPT1ST=TRUE	Any
First adaption of the PID controller at an unknown process or on-line adaption (adaption) of the PID controller at a process already identified previously. The Self-Tuner decides by itself whether it carries out a first adaption or on-line adaption.	FALSE	TRUE	FALSE
Cooling identification Only results from PHASE=4	FALSE	FALSE First adaption must have been carried out	TRUE

First adaption or on-line adaption (adaption of the controller parameters)

Optimization of the controller parameters (first adaption or on-line adaption) is carried out during the heating process, whereby a process identification with subsequent controller design is carried out.

You can request optimization with ADAPT_ON or ADAPT1ST.

With ADAPT_ON = TRUE the Self-Tuner decides by itself whether it carries out a first adaption or on-line adaption. If the Self-Tuner already has information on the process, it automatically calculates a setpoint step and carries out on-line adaption. If information on the process is not available, first adaption is carried out. On-line adaption can only be carried out after the first adaption has been carried out correctly.

If you set ADAPT1ST = TRUE, the Self-Tuner then sets ADAPT_ON = TRUE and carries out a first adaption by force. During a first adaption the Self-Tuner for the process startup uses the parameter LHLM_TUN for the setpoint step.

The parameter ADAPT1ST is only required if serious errors in an adaptation result in such absurd process parameters that the setpoint step calculated automatically is unsuitable for a renewed adaption.

If ADAPT1ST = TRUE, the initialization routine of the Self-Tuners is carried out. The following initial assignments are carried out:

- The system changes over to PI controller. To this purpose the parameters of PI_CON are transferred to the controller. If the parameters PI_CON and PI_CON_TI both have the value of 0.0, their initial assignments are PI_CON = 1.5 and PI_CON_TI = 3600.0 s.
- PROCESS.GAIN = 999.0
- RATIOFAC = 1.0
- STATUS_H= 0; STATUS_D= 0; STATUS_C= 0

Note

If not desired explicitly, do **not** call up the Self-Tuner during a restart (warm start) of the CPU in its initialization routine (ADAPT1ST = TRUE) as is usual for other blocks, since all the process parameters will otherwise be lost.

The optimization is started with ADAPT_ON or ADAPT1ST = TRUE and a subsequent setpoint step ≥ MIN_STEP in the positive direction.

As a rule the setpoint step jumps from the setpoint of the cold process (ambient temperature) to near the operating point during the first adaption. Any setpoint can be used as the initial state as long as it is ensured that the process is in a steady state. No further setpoint steps may be specified during optimization.

You can start optimization either from automatic mode or from manual mode:

- **Starting optimization from automatic mode**
  In automatic mode set ADAPT_ON or ADAPT1ST = TRUE and specify a setpoint step ≥ MIN_STEP.
- **Starting optimization from manual mode**
  In manual mode set ADAPT_ON or ADAPT1ST = TRUE and specify a setpoint step MIN_STEP. The setpoint step does not become effective until you reset MAN_ON = FALSE afterwards. Optimization then begins.

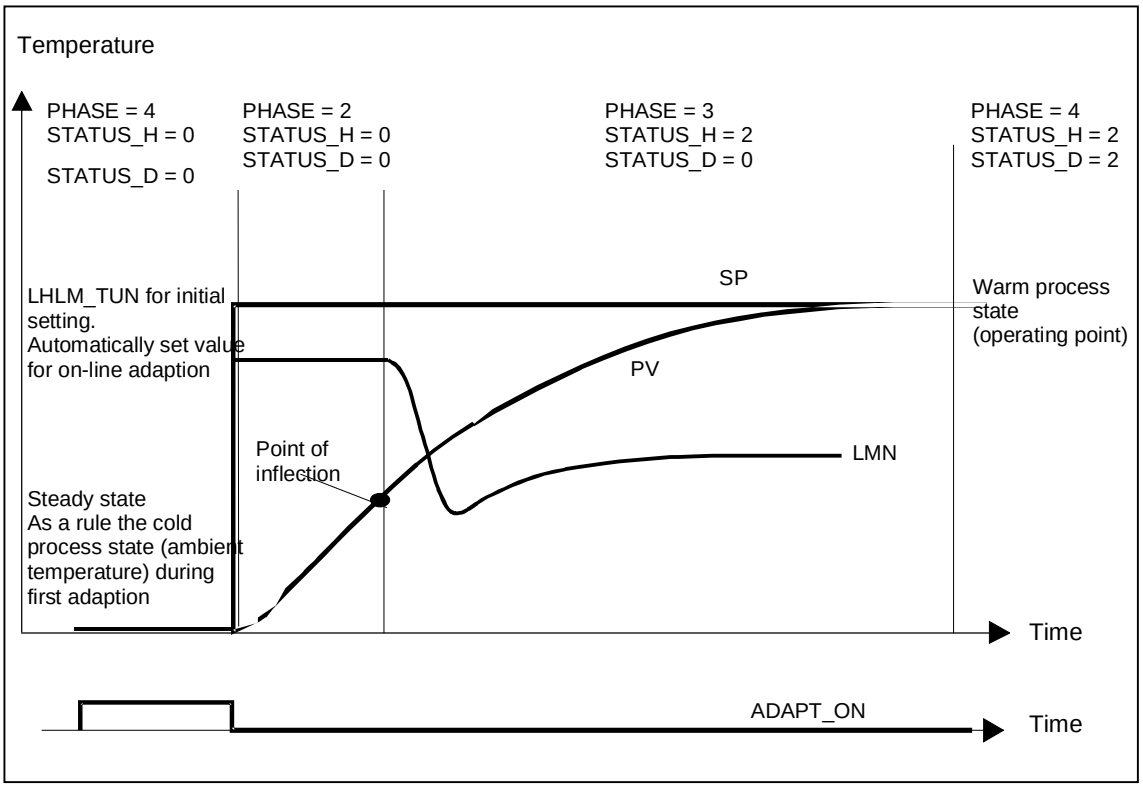


Figure: Phase steps of the first adaption

If you want to abort optimization before the setpoint step, you have to set ADAPT_ON = FALSE. After a setpoint step (PHASE 2 or 3) you have to change over to manual mode (MAN_ON = TRUE) in order to abort optimization.

Setting ADAPT_ON or ADAPT1ST during an active optimization (PHASE 2 or 3) does not cause optimization to be aborted nor are the bits reset. The running optimization is terminated and is re-started after the start conditions (refer to the table "Activating the functions") have been fulfilled.

Phase steps of optimization (first adaption or on-line adaption) during the heating process

- PHASE = 0:

  Zero is initially assigned to the parameter PHASE during the generation of an instance DB for the Self-Tuner. Since TRUE is initially assigned to MAN_ON, the Self-Tuner will change to PHASE 7 (manual mode) after the first run. You can start the first adaption directly from manual operation (ADAPT_ON = TRUE) or change over to automatic mode. When you change over to automatic mode MAN_ON = FALSE for the first time, the parameters are read in from PI_CON. The default values for GAIN and TI are: GAIN = 1.5, TI = 1h.

- PHASE = 2:

  As soon as you specify a setpoint step ≥ MIN_STEP in the positive direction to the operating point of the warm process state, LHLM_TUN is assigned to MAN_OUT during the first adaption, while a value calculated by Self-Tuner is assigned during the on-line adaption. In addition QMAN_ON = TRUE is set. Both values are transferred to the PID controller. The PID controller is controlled with them in manual mode. MIN_STEP should be greater than 10% of the operating range of the setpoint and the process variable.

- PHASE = 3:

  If the point of inflection of the step response is recognized (STATUS_H = 2) or if the process variable has reached 70% of the step size (STATUS_H = 3), a prudently set PI controller is designed. The controller gain GAIN is limited to the range of 0.2 to 20.0. The controller operates immediately as a PI controller and tries to settle the process to a steady state. The Self-Tuner saves the newly determined PI controller parameters in GAIN, TI and PI_CON. The previous values of the PI controller are saved in PI_CON_OLD. If it takes a long time until the steady state is reached (creeping transient response at temperature processes) you can initiate the control design with the current data when the state is almost steady by setting STEADY = TRUE. You can also re-initiate the controller design with the current data at a later stage in Phase 3 or 4 by specifying STEADY = TRUE. This often improves the controller design further.

  If an overshoot occurs or if no point of inflection is found, the cause may be that the setpoint step size LHLM_TUN was selected too large during the first adaption. This does not necessarily produce a bad controller setting. You should select LHLM_TUN approx. 20% smaller at the next first adaption.

  If the block has recognized the steady state or if the period 30*TI (TI: Integral-action time of the PI controller set under PHASE = 3) has passed since the setpoint step, an improved controller design is realized and the system changes over to PHASE = 4. The Self-Tuner saves the newly determined PI/PID controller parameters in GAIN, TI, TD and PI_CON or PID_CON. The previous values of the PID controller are saved in PID_CON_OLD. If possible, a PID controller and a PI controller are designed. If PID_ON=TRUE, the PID controller is activated, otherwise the PI controller. In PHASE 4 you can switch over bumplessly between the two controllers. The controller gain GAIN is

limited in such a manner that the loop gain of the open loop (product of the process gain and the controller gain) lies under 80.0.

- PHASE = 4:

  In this phase the controller controls with its optimized parameters. From this state you can start a cooling identification, an on-line adaption or a first adaption.

Cooling identification

The cooling identification can only be carried out after the first adaption has been carried out correctly. It is started in PHASE 4. When COOLID_ON = TRUE is specified, the controller is set to manual mode and LLLM_TUN is output: MAN_OUT = LLLM_TUN. The Self-Tuner changes to PHASE 3, and COOLID_ON is reset immediately. As soon as a point of inflection is found in the process variable course the Self-Tuner returns to closed-loop control (PHASE 4). The PID Self-Tuner determines a ratio factor RATIOFAC whose value is transferred to the pulse generator module of the controller.

The cooling identification can only be aborted by switching over to manual mode (MAN_ON = TRUE).

Setting COOLID_ON during an active identification or optimization (PHASE 3) does not cause identification or optimization to be aborted, nor are the bits reset. The current identification or optimization is terminated. Only then is identification started or started again.

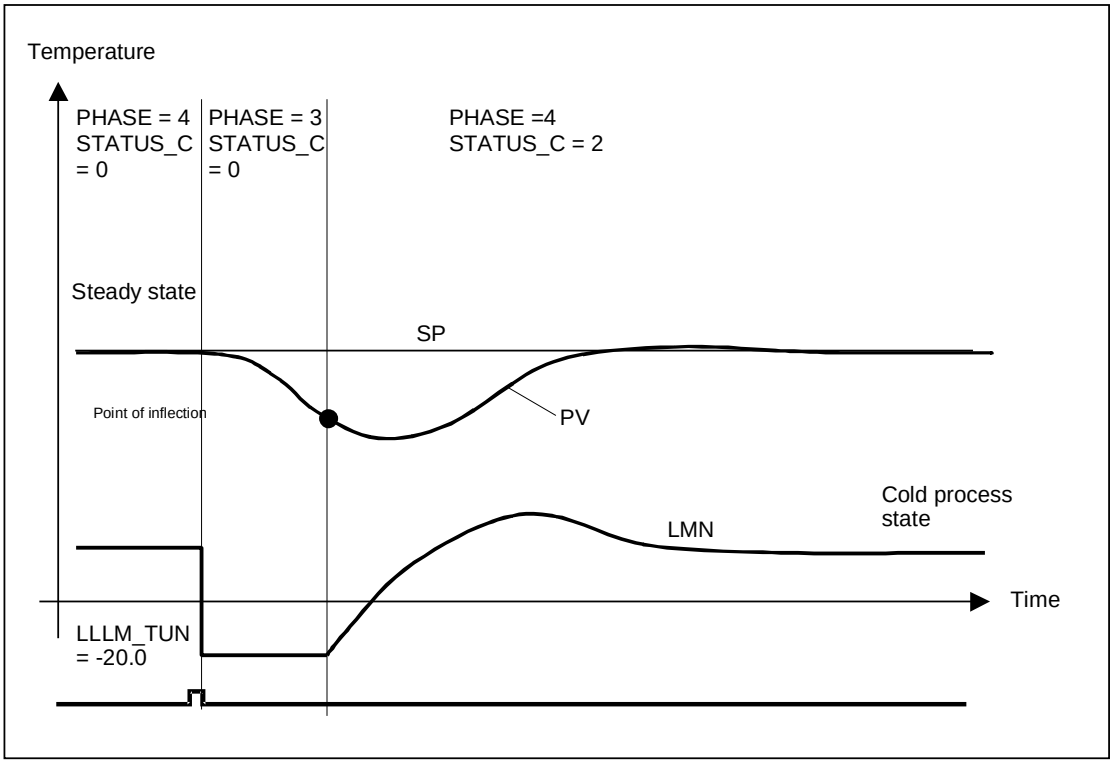


Figure: Cooling identification

Note

- The start bit COOLID_ON can be set simultaneously with ADAPT1ST or ADAPT_ON or while the controller parameters are being optimized. This ensures that the cooling identification can be started immediately after optimization of the controller parameters for heating has been terminated.
- in the case of a plant with several thermally coupled temperature zones (for example plastic machines) the cooling identification should not be started until all (!) the zones) have terminated the first adaption, since the results of the first adaption are otherwise falsified.

Diagnostic values of optimization

The following table describes the results of the diagnostic during the identification of heating and cooling as well as during the controller design.

Diagnostic value	STATUS_H ("heating")	STATUS_D ("design")	STATUS_C ("cooling")
0	Initialization	Initialization	Initialization
1	Point of inflection too low In most cases optimization still results in good controller parameters In order to approach the ideal case increase LHLM_TUN or decrease the setpoint step size.	The process has almost PT1 behavior. You may be able to do without the derivative action of the controller.	Point of inflection not found In most cases optimization still results in good controller parameters. In order to approach the ideal case increase LLLM_TUN (for example from -20.0 to -10.0)
2	Ideal case: Point of inflection found	PI and PID controllers possible, VZ2 behavior	Ideal case: Cooling point of inflection found
3	Point of inflection too high A safety factor is included. In most cases optimization leads to controller parameters which can still be used but are prudent. In order to approach the ideal case decrease LHLM_TUN or increase the setpoint step size.	Separate PID controller design for very high time lags, dead times or high process order In this case the control zone is switched off.	
11, 12, 13	The time lag could not be detected correctly. Operation is continued with an estimate which cannot lead to optimum controller parameters. [1)] Repeat optimization and ensure that disturbances do not occur at the process variable.		Incorrect estimate The ratio factor us set to the default value: RATIOFAC = 1.0 Repeat optimization and ensure that disturbances do not occur at the process variable.

Diagnostic value	STATUS_H ("heating")	STATUS_D ("design")	STATUS_C ("cooling")
4		Incorrect estimate No controller design at on-line adaption. During the first adaption the process is continued with the rough PI controller from PHASE 3. Repeat optimization and ensure that disturbances do not occur at the process variable.	

[1)] If STATUS_H = 11 or 12 and the process has a pure **VZ1 behavior** (no time lag), the Self-Tuner can often still design good parameters. If you are not satisfied with the controller parameters, you can at least use the values from PROCESS.GAIN and PROCESS.TM_LAG1 for a VZ1 model and use your own controller design in accordance with the text book. Since calculation is continued with a roughly estimated time lag, it is possible that STATUS_D cannot be determined to 1 (controller for approximating VZ1 behavior) and that the control zone is de-activated, although a control zone makes sense for VZ1 processes.

Change controller parameters

If you want to change the controller parameters GAIN, TI, TD or TM_LAG by hand after a first adaption or on-line adaption, you can enter your parameters into the output structures PI_CON or PID_CON. If you set the throughput parameter LOAD_PAR to TRUE and the Self-Tuner is in PHASE = 4, either the parameters of PI_CON or of PID_CON are transferred bumplessly to the controller, depending on the PID_ON, and LOAD_PAR is set to FALSE.

If oscillations occur in closed loops or if there are overshoots after setpoint steps, you can reduce the controller gain GAIN (for example, to 80 % of the original value) and increase the integral-action time TI (for example to 150 % of the original value). If the analog manipulated variable of the continuous controller with a pulse generator is converted into binary actuating signals, small sustained oscillations can occur through quantization effects. You can eliminate these by increasing the controller deadband DEADB_W.

Note

The controller parameters are overwritten when the first adaption or adaption is carried out again. The new determined parameters are entered into GAIN, TI, TD and into PI_CON and PID_CON. The old parameters are transferred to PI_CON_OLD and PID_CON.OLD.

Saving the controller parameters

The PID controller parameters found during the first adaption or on-line adaption are entered in the output parameters PI_CON or PID_CON. During saving the PI/PID controller parameters are saved in the output parameters PI_CON_OLD and PID_CON_OLD.

Parameters	PHASE	Behavior
SAVE_PAR = TRUE	All	PID_CON_OLD = PID_CON PI_CON_OLD = PI_CON
UNDO_PAR = TRUE	Only 4	PID_CON_OLD = PID_CON PI_CON_OLD = PI_CON

Typical procedure for manual on-line adaption: Save the current parameters by using SAVE_PAR, change the parameters in PI_CON or PID_CON and activate the new parameters by setting LOAD_PAR. If the new parameters are worse than the old ones, you can re-activate the old ones by using UNDO_PAR.

When optimization is carried out with the Self-Tuner the old controller parameters are saved automatically in PI_CON_OLD or PID_CON_OLD before the controller is designed. If the new parameters are worse than the old ones, you can re-activate the old ones by using UNDO_PAR. These are activated automatically at the controller.

Note

The PID controller parameters are only written back with UNDO_PAR if the controller gain is not equal to zero (PID_CON = PID_CON_OLD, if PID_CON_OLD is not equal to zero). This behavior does not apply to the PI parameters (PID_ON = FALSE).

Setting the sampling time

The sampling time should not exceed 10% of the determined integral-action time of the controller. You can set the sampling time at the CYCLE parameter of the Self-Tuner and of the controller. It has to agree with the time difference between two calls (cycle time of the watchdog interrupt OB under observance of the pulse scalings).

In order for the process parameters to be identified reliably, the smallest time constant of the process (time lag) should be sampled at least 10 to 20 times.

When used with the controller modules FM355/455 the time between the Self-Tuner calls (sampling time) has to be approximately equal to the execution times of the controllers on the FM. The execution time is specified in the parameter assignment tool of the FMs under **Options > Module parameters > Resulting Cycle Time**.

Normalization of the controller gain under Standard PID Control V5

The controller blocks in Standard PID Control V5 operate with a normalized controller gain which is calculated as follows:

GAIN (normalized) = GAIN (physical) * (NM_PVHR - NM_PVLR)/100

This results in a normalization factor which you have to enter at the parameter NORM_FAC of the Self-Tuner:

NORM_FAC = (NM_PVHR - NM_PVLR)/100

This results in the normalized controller gain being applied at the GAIN output of the Self-Tuner. In the examples for Standard PID Control V5 this point is programmed in the start-up branch of the blocks TUN_CON_C, TUN_CON_P and TUN_CON_S.

NORM_FAC remains at its default assignment of 1 in the case of controllers with a physical controller gain.

Note

Normalizations which lie in the signal course before normalization with NM_PVHR or NM_PVLR belong to the process and may not be taken into consideration at the Self-Tuner.

Execution phases of the Self-Tuner

PHASE = 0	Automatic mode: Closed-loop control with preset parameters after generation of an instance DB. No further functions possible. MAN_ON or ADAPT1ST is used to exit the phase.
PHASE = 2	Controller optimization: Search for the point of inflection at a constant manipulated variable
PHASE = 3	Controller optimization: Controlling with a PI coarse controller
PHASE = 4	Automatic mode: Normal closed-loop control with/without control zone
PHASE = 5	Automatic mode: Structure changeover at setpoint step: Integral action de-activated, P(D) control
PHASE = 6	Automatic mode: Structure changeover at setpoint step: Prediction-control specification of the manipulated variable
PHASE = 7	Manual mode

Function activation

Function	Start bit	Enable bit	Start condition: PHASE	Start condition: Setpoint step at SP
Manual mode	---	MAN_ON	0 to 7	---
Manual/automatic changeover with prediction-control manipulated-value selection	---	PRED_ON	Transition 7 -> 4	---
First adaption	ADAPT1ST	---	0, 4	Pos.: > MIN_STEP
First adaption or on-line adaption	ADAPT_ON	---	4	Pos.: > MIN_STEP
Cooling identification	COOLID_ON	---	4	---
Control zone	---	CONZ_ON	4	
Structure changeover	---	STRUC_ON	4	Pos.: > MIN_STEP, Neg.: > MIN_STEP * RATIO_FAC
Save controller parameters	SAVE_PAR	---	2 to 7	---
Load controller parameters back	UNDO_PAR	---	4	---
Change controller parameters	LOAD_PAR	---	4	---

Note

Start bits are reset after the function has been started successfully. Enable bits remain set.

Input parameters

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
SP	REAL	Setpoint	Technical range of values	0.0	The setpoint is entered at the input SP. It is interconnected to the controller via SP_OUT. Note that the setpoint must lie within the setpoint limits of the controller.
PV	REAL	Process variable	Technical range of values	0.0	PV is interconnected to the process variable on the controller.
LMN	REAL	Manipulated variable	-100.0 ... 100.0 (%)	0.0	LMN is interconnected to the manipulated variable on the controller.

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
MAN	REAL	Manual value	-100.0 ... 100.0 (%)	0.0	The manual value is entered at the input MAN. If MAN_ON = TRUE, the manual value MAN is transferred to the manipulated variable of the controller. If MAN_ON = TRUE and MAN = 0.0, the heating/cooling function is off. If MAN_ON = FALSE, the manipulated variable of the controller is specified.
MIN_STEP	REAL	Minimal setpoint step	Recommended: ≥ 10 % of the required setpoint-process-variable range	10.0	You can specify the minimal setpoint step size at the input MIN_STEP above which a first adaption or adaption can be carried out. If the structure changeover is activated (STRUC_ON = TRUE), the setpoint step must be greater than MIN_STEP in order for the structure changeover to be activated. MIN_STEP should not be configured to less than 10% of the operating range of the setpoint value or process variable.
LHLM_TUN	REAL	Manipulated value high limit on self-tuning	0.0 ... 100.0 (%)	80.0	The setpoint step for the first adaption is entered at the input LHLM_TUN. During on-line adaption the setpoint step is calculated automatically.
LLLM_TUN	REAL	Manipulated value low limit on self-tuning	-100.0 ... 0.0 (%)	-20.0	The setpoint step for the cooling identification is entered at the input LLM_TUN.
NORM_FAC	REAL	Proportional gain normalizing factor	Dimensionsless	1.0	A normalizing factor is applied at controllers with a normalized controller gain (Standard PID Control V5). The normalizing factor is calculated on the basis of the range limits and is interconnected to the input NORM_FAC (refer to the examples with Standard PID Control V5 in the FBs TUN_CON_C, TUN_CON_P and TUN_CON_S).

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
MAN_ON	BOOL	Manual mode on		TRUE	If MAN_ON = TRUE, the setpoint of the controller is specified via the input MAN. If MAN_ON = FALSE, the manipulated variable of the controller is specified.
PID_ON	BOOL	PID mode on		TRUE	You can specify at the input PID_ON whether the optimized controller is to operate as a PI or as a PID controller. PID controller: PID_ON = TRUE PI controller: PID_ON = FALSE However, it is possible that a PI controller is designed at some types of process although PID_ON = TRUE.
STRUC_ON	BOOL	Variable structure control for setpoint steps		FALSE	If STRUC_ON = TRUE, a structure changeover (from PI(D) to P(D) controlling) ensures at setpoint steps that overshoots are avoided to a great extent. The input STRUC_ON may not be set = TRUE, if the proportional or derivative action is fed into the feedback loop at the controller.
WRITE_DIS	BOOL	Writing to FM is not possible		FALSE	In connection with the controller module the Self-Tuner has to know whether the driver block is ready to send data to the FM. This is the case if LOAD_OP = LOAD_PAR = FALSE. The result of the OR operation of LOAD_OP and LOAD_PAR is interconnected to WRITE_DIS.

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
PRED_ON	BOOL	Manual to automatic changeover with prediction of the manipulated value		TRUE	When changing over to automatic mode you can specify whether you want to change over bumplessly from the last manual value or with bumps with a manipulated variable calculated by the Self-Tuner. If PRED_ON = TRUE (default setting) the changeover is carried out with bumps. The changeover with bumps does not become effective until after an optimization.
CYCLE	TIME	Sample time	1ms	100ms	The sample time has to be configured at the input CYCLE. CYCLE must agree with the sample time of the controller to be set and with the time difference between two calls (cycle time of the watchdog interrupt OB under observance of the pulse scalings).

Output parameters

Parameters	Data type	Comment	Initial	Explanation
MAN_OUT	REAL	Manual value output	0.0	MAN_OUT is connected to the manual input value of the controller. If MAN_ON = TRUE, the Self-Tuner outputs the value which is specified at the input MAN at the output MAN_OUT.
SP_OUT	REAL	Setpoint output	0.0	SP_OUT is connected to the setpoint input of the controller. The Self-Tuner writes the setpoint specified at the input SP to the output SP_OUT. At manual operation MAN_ON = TRUE the setpoint is tracked to the process variable: SP_OUT = PV.
GAIN	REAL	Proportional gain	1.5	GAIN is interconnected to the proportional-action coefficient (controller gain) of the controller.
TI	TIME	Reset time Integral time	1h	TI is interconnected to the integral time (reset time) of the controller.

Parameters	Data type	Comment	Initial	Explanation
TD	TIME	Derivative time	0ms	TD is interconnected to the derivative time (rate time) of the controller.
TM_LAG	TIME	Time lag	1s	TM_LAG is interconnected to the time lag of the controller differentiator.
RATIOFAC	REAL	Ratio factor heating/cooling	1.0	RATIOFAC is interconnected to the ratio factor heating/cooling of the controller or of the FB PULSEGEN. The parameter is only required for cooling.
PHASE	INT	Phase 0 to 7	0	PHASE displays the current execution phase of controller optimization. During the optimization of Standard PID Control V5 the output PHASE is interconnected to the output PHASE of the FB PID_CP or PID_ES.
STATUS_H	INT	Status heating	0	STATUS_H displays optimization results for the search of the point of inflection during the heating process.
STATUS_D	INT	Status controller design	0	STATUS_D displays optimization results for the controller design during the heating process.
STATUS_C	INT	Status cooling	0	STATUS_C displays optimization results for the search of the point of inflection during the cooling process.
QMAN_ON	BOOL	Manual mode on	FALSE	QMAN_ON is interconnected to the manual/automatic changeover of the controller.
QI_SEL	BOOL	Integral action on	TRUE	QI_SEL is interconnected to the on/off switch of the controller integral action I_SEL. In the case of FM355/455 TI has to be switched to zero depending on QI_SEL: If QI_SEL = FALSE, then TI=0.
QD_SEL	BOOL	Derivative action on	FALSE	QD_SEL is interconnected to the on/off switch of the controller derivative action D_SEL. In the case of FM355/455 TD has to be switched to zero depending on QD_SEL: If QD_SEL = FALSE, then TD=0.
QWRITE	BOOL	Self-Tuner writes parameters to the PID controller	FALSE	QWRITE is interconnected to the inputs LOAD_PAR and LOAD_OP of the driver block PID_CS of FM355/455. The result is that the records of the controller channel are downloaded to the FM.
PROCESS	STRUCT			
GAIN	REAL	Process gain	999.0	The process gain is displayed at the output PROCESS.GAIN.
TM_LAG1	REAL	Process slow time constant	0.0	The process slow time constant is displayed at the output PROCESS.TM_LAG1.

Parameters	Data type	Comment	Initial	Explanation
TM_LAG2	REAL	Process fast time constant	0.0	The process fast time constant is displayed at the output PROCESS.TM_LAG2.
PV00	REAL	Process steady state value for zero input	0.0	The steady state of the process is displayed at the output PROCESS. PV00 The steady state is the steady-state process variable at Setpoint = zero. This corresponds to the ambient tempera ture at a cooling plant.
KIG	REAL	Maximal ascent ratio of PV	0.0	The maximal ascent ratio of the process variable during heating is displayed at the output PROCESS. KIG.
KIG_C	REAL	Maximal ascent ratio of PV (cooling)	0.0	The maximal descent ratio of the process variable during cooling is displayed at the output PROCESS. KIG_C.
	END_ST			
PI_CON	STRUCT			
GAIN	REAL	PI proportional gain	1.5	PI_CON.GAIN displays the proportional gain of the PI controller. You can improve the value manually and make it effective by setting LOAD_PAR.
TI	REAL	PI reset time PI integration time	3600.0	PI_CON.TI displays the integration time of the PI controller. You can improve the value manually and make it effective by setting LOAD_PAR.
	END_ST			
PID_CON	STRUCT			
GAIN	REAL	PID proportional gain	0.0	PID_CON.GAIN displays the proportional gain of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
TI	REAL	PID reset time PID integration time	0.0	PID_CON.TI displays the integration time of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
TD	REAL	PID derivative time	0.0	PID_CON.TD displays the derivative time of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
	END_ST			
PI_CON_ OLD	STRUCT			
GAIN	REAL	PI old proportional gain	1.5	The old proportional gain of the PI controller is displayed at the output PI_CON_OLD.GAIN.
TI	REAL	PI old reset time PI old integration time	3600.0	The old integration time of the PI controller is displayed at the output PI_CON_OLD.TI.
	END_ST			

Parameters	Data type	Comment	Initial	Explanation
PID_CON_ OLD	STRUCT			
GAIN	REAL	PID old proportional gain	0.0	The old proportional gain of the PID controller is displayed at the output PID_CON_OLD.GAIN.
TI	REAL	PID old reset time PID old integration time	0.0	The old integration time of the PID controller is displayed at the output PID_CON_OLD.TI.
TD	REAL	PID old derivative time	0.0	The old derivative time of the PID controller is displayed at the output PID_CON_OLD.TD.
	END_ST			
TU	REAL	Time lag	0.0	The determined time delay is displayed at the output TU.
TA	REAL	Recovery time	0.0	The determined recovery time is displayed at the output TA.

Throughput parameters

Parameters	Data type	Comment	Initial	Explanation
ADAPT1ST	BOOL	First adaption	FALSE	When you carry out the first optimization of the process or do not wish to use the results of previous optimizations such as setpoint steps or filter time constants, when you carry out a first adaption. To this purpose set ADAPT1ST = TRUE. Then the Self-Tuner carries out an initialization run and in the process sets the initial values described in the initialization section. The Self-Tuner resets ADAPT1ST back to FALSE. It does not make sense to set ADAPT1Self-Tuner to true at a restart of the CPU in OB100.
ADAPT_ON	BOOL	Switch on on-line adaption with next setpoint step	FALSE	If you want to trigger on-line adaption, set ADAPT_ON = TRUE. If the subsequent setpoint step is greater than MIN_STEP, identification is started (PHASE = 2). The Self-Tuner resets ADAPT_ON back to FALSE.

Parameters	Data type	Comment	Initial	Explanation
STEADY	BOOL	Steady state reached	FALSE	When the process variable has reached a steady state, you can use STEADY to terminate optimization prematurely, although the Self-Tuner is still in PHASE 3. You can also use STEADY to carry out a renewed controller design in steady state, after the Self-Tuner has passed into PHASE 4.
COOLID_ON	BOOL	Cooling identification on	FALSE	If you want to trigger cooling identification, set COOLID_ON = TRUE. The setpoint is set to LLLM_TUN and COOLID_ON is reset to FALSE. In the case of several heating zones all first adaptions or adaptions should have been completed, before a cooling identification is started.
UNDO_PAR	BOOL	Undo change of controller parameters	FALSE	The old controller parameters are transferred to the current controllers parameters and are active immediately depending on PID_ON. PI_CON = PI_CON_OLD PID_CON = PID_CON_OLD The function for undoing changes can only be used during automatic mode (PHASE 4).
SAVE_PAR	BOOL	Save current controller parameters	FALSE	The current controller parameters are transferred to the old controller parameters. PI_CON_OLD = PI_CON PID_CON_OLD = PID_CON
LOAD_PAR	BOOL	Activate changed controller parameters	FALSE	The modified parameters PI_CON and PID_CON are transferred to the controller and are thus active.
CONZ_ON	BOOL	Control zone active	FALSE	If CONZ_ON = TRUE, the controller operates with a control zone. During controller optimization the Self-Tuner decides whether controlling is carried out with or without control zone and determines an optimal control zone. After optimization has been carried out in PHASE 4 you can de-activate the control zone with CONZ_ON = FALSE or re-configure the control zone.

Parameters	Data type	Comment	Initial	Explanation
CON_ZONE	REAL	Control zone	0.0	CON_ZONE specifies the width of the control zone. During controller optimization the Self-Tuner decides whether controlling is carried out with or without control zone and determines an optimal control zone. After optimization has been carried out in PHASE 4 you can de-activate the control zone with CONZ_ON = FALSE or re-configure the control zone.

3.2 FB "TUN_ES"

How the function works

The block FB TUN_ES is interconnected to a PID step controller. The FB TUN_ES disposes of the following functions to control or optimize the controller:

- **Manual mode**
  The Self-Tuner forces the controller into manual mode and transfers its manual parameters to it. In the case of step controllers with position feedback you can specify the manual value in floating point format and manual value signals such as open/close. In the case of step controllers without position feedback you can only specify the manual value signals.
- **Controller optimization**
  When the controller parameters are optimized, the actuating time of the final control element is measured and then the controlled system is identified. A controller is designed based on the system characteristics. The new system parameters are calculated on this basis and activated for the controller.
  - **First adaption**
    The first adaption can only be carried out in case of a positive manipulated variable step. It is usually used during the first startup or in case of a serious change in the controlled system characteristics.
  - **On-line adaption**
    The controller can be optimized subsequently on-line during a positive setpoint step when the operating point or the process response is changed. On-line adaption is only possible at step controllers with position feedback.

- **Optimization of the response to setpoint changes**
  The controller is designed for optimum disturbance response. The resultant "stringent" parameters would result in overshoots of 10% to 20% of the step height at setpoint steps.

  The **structure segmentation** represents an initial remedy, in which the proportional or derivative action is moved into the feedback loop. A change in the setpoint thus only acts on the integral action of the controller. Structure segmentation may only be used for step controllers with position feedback.

  If structure segmentation is not possible for the controller, the following function can be activated in order to optimize the response to setpoint changes:

  Structure changeover:
  Changeover quasi to P(D) closed-loop control or prediction-control specification of the manipulated variable at large positive setpoint steps. In the case of step controllers without position feedback it is only possible to change over quasi to P(D) closed-loop control.
- **Saving the controller parameters**
  If you judge the current controller parameters to be utilizable, you can save these before a manual change of the controller parameters in parameters specially provided in the instance data block of the Self-Tuner. During controller optimization the saved parameters are overwritten by the values valid before optimization.
- **Restore saved controller parameters**
  This function allows the controller parameters last saved to be re-activated for the controller.
- **Changing the controller parameters**
  If you want to set the controller parameters yourself for the controller, you can assign these through special parameters in the instance DB of the Self-Tuner and transfer them to the controller by means of this function.

Remark: The following section assumes a general temperature process with active heating and cooling for the description of the process.

A cooling identification cannot be carried with the TUN_ES block.

The TUN_ES block cannot operate in control zone mode.

Modes

Structure segmentation

Structure segmentation is a function of the controller. In the case of closed-loop control with structure segmentation the proportional or derivative action lies in the feedback loop, so that a change in the setpoint only acts on the integral action of the controller. Manipulated variable steps do not occur during setpoint steps - the manipulated variable takes a ramp course. The disturbance response is not changed as a result.

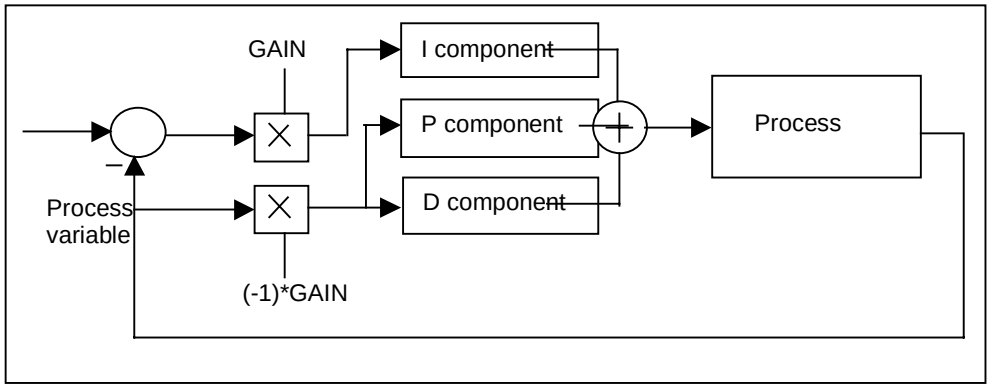


Figure: Closed-loop control with structure segmentation

In the case of closed-loop control without structure segmentation the proportional and derivative actions lie in the error signal just like the integral action. You can select a structure changeover when you are in this operating mode (STRUC_ON=1).

Closed-loop control with structure segmentation is possible in the following control packages:

- Standard PID Control
- Modular PID Control
- FM355/455
- PCS7 controller

Note

- Structure segmentation may not be used if no position feedback of the control valve is available. This means that you must leave the proportional and derivative actions in the error signal.
- In closed-loop control with structure segmentation the dead band in the error signal is almost without effect.
- Only closed-loop control without structure segmentation is possible for the simple controller PID Control of Step7-Basis and in the CPU 314 IFM.

Structure changeover at proportional and derivative actions in the error signal

The structure changeover is de-activated in the default setting. It should only be selected if structure segmentation is not available. You can activate the structure changeover function by setting STRUC_ON = TRUE. The block automatically selects between two controlling measures depending on the total loop gain (product of the process gain and the controller gain):

- Integral action temporarily increased, quasi P(D) control:

  In the case of a positive setpoint step ≥ MIN_STEP the integral action of the controller is increased considerably temporarily and the gain is decreased slightly, so that quasi a pure P(D) controller is used. When the setpoint value is approached the system switches back to the original values of the integral action and the gain is reduced again (PHASE = 5 in the operation of the Self-Tuner).

- Temporary prediction-control specification of the manipulated variable:

  Processes with high time lags cannot be controlled well with a P(D) control system. The manipulated variable required steadily for the new setpoint is therefore output constant after a positive setpoint step ≥ MIN_STEP. When the setpoint value is approached the system changes back bumplessly to PI or PID controller operating mode. This results in a slow action which is, however, free of overshooting (PHASE = 5 in the operation of the Self-Tuner).

Note

In the case of step controlling without position feedback only the first control measure (PASE = 5) is possible.
If you do not achieve good results (slow transient response) with the structure changeover function at positive setpoint steps (for example heating processes) , you can de-activate the structure changeover function by setting STRUC_ON = FALSE, provided that small overshoots may occur.

Modes of the structure	**PFDB_SEL, DFDB_SEL** [1)]	**STRUC_ON**
Activate the structure segmentation	TRUE	FALSE
Activate the structure changeover	FALSE	TRUE

[1)] Controller parameters PFDB_SEL, DFDB_SEL: These parameters do not exist in the simple controllers of STEP7 CONT_S. This corresponds to the setting PFDB_SEL = DFDB_SEL = FALSE (without structure segmentation).

Information on selecting the operating mode

- In the case of setpoint steps structure segmentation prevents overshooting of the control variable. Select structure segmentation is possible.
- Activate the structure changeover function if the controller does not allow structure segmentation. The structure changeover function may not be active at the same time as the structure segmentation function. You can de-activate structure changeover if rapid settling is required and small overshoots are permitted at positive setpoint steps.

Manual mode

The operating mode "controller manual mode" corresponds to PHASE = 7 in the operation of the Self-Tuner. If you use a step controller with position feedback (LMNR_ON = TRUE), you can use LMNS_ON = TRUE or MAN_ON = TRUE to change over to manual mode. If you use a step controller without position feedback (LMNR_ON = FALSE) , you can only use LMNS_ON = TRUE to change over to manual mode. If you set the input MAN_ON to TRUE, the output QMAN_ON is set to TRUE and the output MAN_OUT to MAN. If you set the input LMNS_ON to TRUE, the output QLMNUP is set to LMNUP and the output QLMNDN to LMNDN.

Manual mode takes priority over all other operating modes.

A first adaption or an on-line adaption is aborted. When manual operation is de-activated (MAN_ON = FALSE), the controller changes over to automatic mode (PHASE = 4 in the operation of the Self-Tuner) and controls with the existing controller parameters.

Controller optimization

Optimization modes in the Self-Tuner	ADAPT1ST	ADAPT_ON
Deleting of all the old process information and requesting of a first adaption with measurement of the motor actuating time. Start at a cold process state and setpoint step to LHLM_TUN	TRUE	Any Is set by the Self-Tuner if ADAPT1ST=TRUE
First adaption of the PID controller at an unknown process or on-line adaption (adaption) of the PID controller at a process already identified previously. The Self-Tuner decides by itself whether it carries out a first adaption or on-line adaption. Start at any process state and setpoint step to LHLM_TUN or automatically calculated value. The motor actuating time is not measured. Only possible at step controllers with position feedback.	FALSE	TRUE LMNR_ON from controller and Self-Tuner = TRUE

First adaption or on-line adaption (adaption of the controller parameters)

The controller parameters (first adaption with or without measurement of the motor actuating time or on-line adaption) are optimized during the heating-up process, whereby a process identification with subsequent controller design is carried out.

You can request optimization with ADAPT1ST or ADAPT_ON (only for step controllers with position feedback).

If you set ADAPT1ST=TRUE, the Self-Tuner sets ADAPT_ON=TRUE and closes the control valve (PHASE = 1 in the operation of the Self-Tuner).

If ADAPT1ST = TRUE, the initialization routine of the Self-Tuners is carried out. The following initial assignments are carried out:

- The system changes over to PI controller. To this purpose the parameters of PI_CON are transferred to the controller. If the parameters PI_CON and PI_CON_TI both have the value of 0.0, their initial assignments are PI_CON = 1.5 and PI_CON_TI = 3600.0 s.
- PROCESS.GAIN = 999.0
- STATUS_H= 0; STATUS_D= 0

Note

If not desired explicitly, do **not** call up the Self-Tuner during a restart (warm start) of the CPU in its initialization routine as is usual for other blocks, since all the process parameters will otherwise be lost.

After a setpoint step ≥ MIN_STEP the motor actuating time is measured after which a first adaption is carried out. During a first adaption the Self-Tuner for the process startup uses the parameter LHLM_TUN for the setpoint step. In the case of step controllers with position feedback the value LHLM_TUN is approached directly. In the case of step controllers without position feedback with control valve is opened completely and then traveled closed until the share of the motor actuating time specified by (100 - LHLM_TUN) has expired.

You can only start the optimization with ADAPT_ON in the case of step controllers with position feedback. A first adaption without measurement of the motor actuating time or on-line adaption is started with ADAPT_ON = TRUE and a subsequent setpoint step ≥ MIN_STEP in the positive direction. The Self-Tuner then immediately passes over to PHASE = 2 and carries out a setpoint step to LHLM_TUN for a first adaption or to an automatically calculated setpoint for on-line adaption. The Self-Tuner decides whether a first adaption or an on-line adaption is to be carried out on the basis of old process values. As a rule the setpoint step jumps from the setpoint of the cold process (ambient temperature) to near the operating point during the first adaption. Any setpoint can be used as the initial state as long as it is ensured that it is a steady state. No further setpoint steps may be specified during optimization.

You can start optimization either from automatic mode or from manual mode:

- **Starting optimization from automatic mode**

  In automatic mode set ADAPT_ON or ADAPT1ST = TRUE and specify a setpoint step ≥ MIN_STEP.

Starting optimization from manual mode

In manual mode set ADAPT_ON or ADAPT1ST = TRUE and specify a setpoint step ≥ MIN_STEP. The setpoint step does not become effective until you reset MAN_ON = FALSE afterwards. Optimization then
begins.

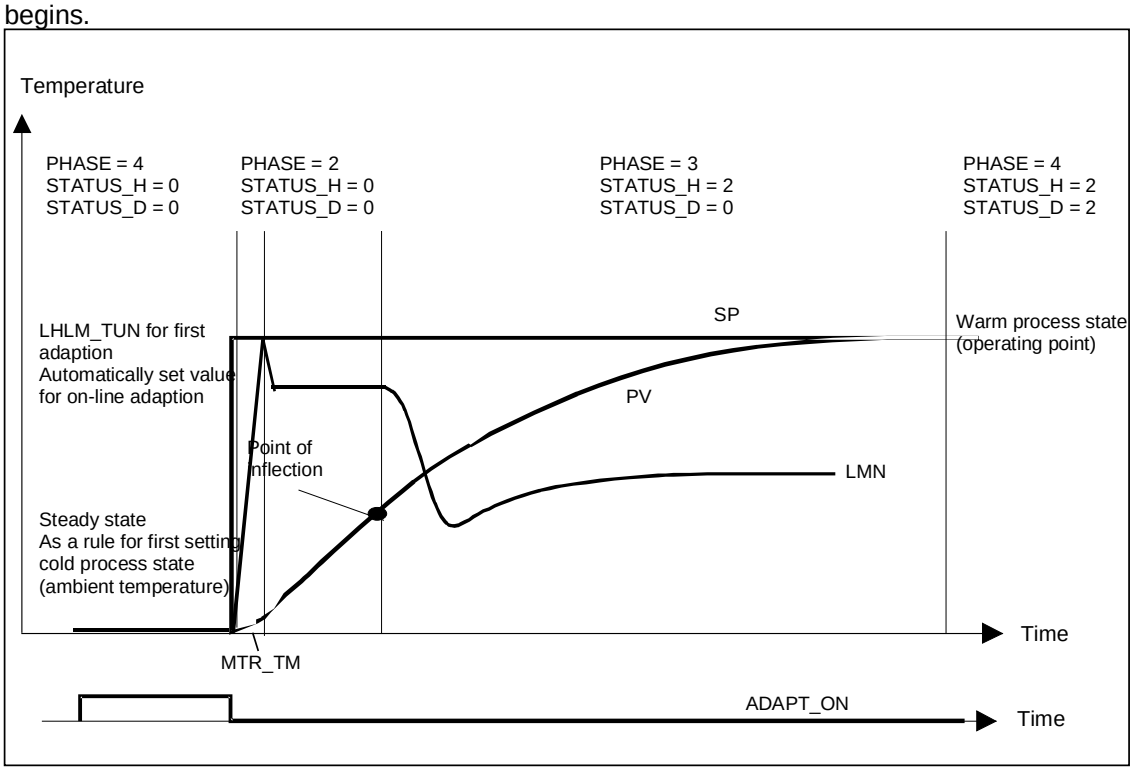


Figure: Phase steps of the first adaption with measurement of the motor actuating time at a step controller with position feedback (start via ADAPT1ST = TRUE)

If you want to abort optimization before the setpoint step, you have to set ADAPT_ON = FALSE. After a setpoint step (PHASE 2 or 3) you have to change over to manual mode (MAN_ON = TRUE) in order to abort optimization.

Setting ADAPT_ON or ADAPT1ST during an active optimization (PHASE 2 or 3) does not cause optimization to be aborted nor are the bits reset. The running optimization is terminated and is re-started after the start conditions (refer to the table "Activating the functions") have been fulfilled.

Phase steps of optimization (first adaption or on-line adaption) during the heating process

- PHASE = 0:

  Zero is initially assigned to the parameter PHASE during the generation of an instance DB for the Self-Tuner. Since TRUE is initially assigned to LMNS_ON, the Self-Tuner will change to PHASE 7 (manual mode) after the first run. You can start the first adaption directly from manual operation (ADAPT_ON = TRUE) or change over to automatic mode. When you change over to automatic mode MAN_ON = FALSE for the first time, the parameters are read in from PI_CON. The default values for GAIN and TI are: GAIN = 1.5, TI = 1h.

- PHASE = 1:

  After optimization has been started with ADAPT1ST = TRUE, the Self-Tuner travels the control valve closed (setpoint = 0 or LMNS_ON = LMNDN = TRUE) and waits for a positive setpoint step ≥ MIN_STEP.

- PHASE = 2:

  As soon as you specify a setpoint step ≥ MIN_STEP in the positive direction to the operating point of the warm process state, LHLM_TUN is assigned to MAN_OUT during the first adaption, while a value calculated by Self-Tuner is assigned during the on-line adaption. MIN_STEP should be greater than 10% of the operating range of the setpoint and the process variable.

  In the case of step control with position feedback (LMNR_ON=TRUE) the value TRUE is assigned to QMAN_ON. The controller controls the valve to the value of MAN_OUT and the FB TUN_ES calculates the motor actuating time MTR_TM.

  In the case of a step control without position feedback QLMNS_ON and QLMNUP are set to TRUE and the valve is traveled to the upper stop. When the upper stop is reached (LMNR_HS =TRUE), the FB TUN_ES calculates the motor actuating time and transfers it to the controller. Then the valve is closed down to the value of MAN_OUT (QLMNDN = TRUE).

- PHASE = 3:

  If the point of inflection of the step response is recognized (STATUS_H = 2) or if the process variable has reached 70% of the step size (STATUS_H = 3), a prudently set PI controller is designed. The controller gain GAIN is limited to the range of 0.2 to 20.0. The controller operates immediately as a PI controller and tries to settle the process to a steady state. The Self-Tuner saves the newly determined PI controller parameters in GAIN, TI and PI_CON. The previous values of the PI controller are saved in PI_CON_OLD. If it takes a long time until the steady state is reached (creeping transient response at temperature processes) you can initiate the control design with the current data when the state is almost steady by setting STEADY = TRUE. You can also re-initiate the controller design with the current data at a later stage in Phase 3 or 4 by specifying STEADY = TRUE. This often improves the controller design further.

If an overshoot occurs or if no point of inflection is found, the cause may be that the setpoint step size LHLM_TUN was selected too large during the first adaption. This does not necessarily produce a bad controller setting. You should select LHLM_TUN approx. 20% smaller at the next first adaption.

If the block has recognized the steady state or if the period 30*TI (TI: Integral-action time of the PI controller set under PHASE = 3) has passed since the setpoint step, an improved controller design is realized and the system changes over to PHASE = 4. The Self-Tuner saves the newly determined PI/PID controller parameters in GAIN, TI, TD and PI_CON or PID_CON. The previous values of the PID controller are saved in PID_CON_OLD. If possible, a PID controller and a PI controller are designed. If PID_ON=TRUE, the PID controller is activated, otherwise the PI controller. In PHASE 4 you can switch over bumplessly between the two controllers. In the case of difficult processes the block always designs a PID controller irrespective of PID_ON. The controller gain GAIN is limited in such a manner that the loop gain of the open loop (product of the process gain and the controller gain) lies under 40.0.

- PHASE = 4:

  In this phase the controller controls with its optimized parameters. From this state you can start an on-line adaption or a first adaption.

Diagnostic values of optimization

The following table describes the results of the diagnostic during the identification of heating as well as during the controller design.

Diagnostic value	STATUS_H ("heating")	STATUS_D ("design")
0	Initialization	Initialization
1	Point of inflection too low In most cases optimization still results in good controller parameters. In order to approach the ideal case increase LHLM_TUN or decrease the setpoint step size.	Only PI controller design since the process almost has PT1 behavior
2	Ideal case: Point of inflection found	Ideal case: PI and PID controllers possible, VZ2 behavior
3	Point of inflection too high A safety factor is included. In most cases optimization leads to controller parameters which can still be used but are prudent. In order to approach the ideal case decrease LHLM_TUN or increase the setpoint step size.	Only PI controller design since there is a very high time lag / dead time or high order

Diagnostic value	STATUS_H ("heating")	STATUS_D ("design")
11, 12, 13	The time lag could not be detected correctly. Operation is continued with an estimate which cannot lead to optimum controller parameters. [1)] Repeat optimization and ensure that disturbances do not occur at the process variable.	
4		Incorrect estimate No controller design at on-line adaption. During the first adaption the process is continued with the rough PI controller from PHASE 3. Repeat optimization and ensure that disturbances do not occur at the process variable.

[1)] If STATUS_H = 11 or 12 and the process has a pure **VZ1 behavior** (no time lag), the Self-Tuner can often still design good parameters. If you are not satisfied with the controller parameters, you can at least use the values from PROCESS.GAIN and PROCESS.TM_LAG1 for a VZ1 model and use your own controller design in accordance with the text book. Since calculation is continued with a roughly estimated time lag, it is possible that STATUS_D cannot be determined to 1 (controller for approximating VZ1 behavior) and that the control zone is de-activated, although a control zone makes sense for VZ1 processes.

Change controller parameters

If you want to change the controller parameters GAIN, TI, TD or TM_LAG by hand after a first adaption or on-line adaption, you can enter your parameters into the output structures PI_CON or PID_CON. If you set the throughput parameter LOAD_PAR to TRUE and the Self-Tuner is in PHASE = 4, either the parameters of PI_CON or of PID_CON are transferred bumplessly to the controller, depending on the PID_ON, and LOAD_PAR is set to FALSE. If the controller gain PID_CON.GAIN is equal to zero, the PI controller parameters are transferred to the controller.

If oscillations occur in closed loops or if there are overshoots after setpoint steps, you can reduce the controller gain GAIN (for example, to 80 % of the original value) and increase the integral-action time TI (for example to 150 % of the original value set). If the analog manipulated variable of the continuous controller with a pulse generator is converted into binary actuating signals, small sustained oscillations can occur through quantization effects. You can eliminate these by increasing the controller deadband DEADB_W.

Note

The controller parameters are overwritten when the first adaption or adaption is carried out again. The new determined parameters are entered into GAIN, TI, TD and into PI_CON and PID_CON. The old parameters are transferred to PI_CON_OLD and PID_CON_OLD.

Saving the controller parameters

The PID controller parameters found during the first adaption or on-line adaption are entered in the output parameters PI_CON or PID_CON. During saving the PI/PID controller parameters are saved in the output parameters PI_CON_OLD and PID_CON_OLD.

Parameters	PHASE	Behavior
SAVE_PAR = TRUE	All	PID_CON_OLD = PID_CON PI_CON_OLD = PI_CON
UNDO_PAR = TRUE	Only 4	PID_CON_OLD = PID_CON PI_CON_OLD = PI_CON

Typical procedure for manual on-line adaption: Save the current parameters by using SAVE_PAR, change the parameters in PI_CON or PID_CON and activate the new parameters by setting LOAD_PAR. If the new parameters are worse than the old ones, you can activate the old ones by using UNDO_PAR.

When optimization is carried out with the Self-Tuner the old controller parameters are saved automatically in PI_CON_OLD or PID_CON_OLD before the controller is designed. If the new parameters are worse than the old ones, you can re-activate the old ones by using UNDO_PAR. These are activated automatically at the controller.

Note

The PID controller parameters are only written back with UNDO_PAR if the controller gain is not equal to zero (PID_CON = PID_CON_OLD, if PID_CON_OLD is not equal to zero).

Setting the sampling time

The sampling time should not exceed 10% of the determined integral-action time of the controller. In addition the sampling time should not exceed 1% of the motor actuating time so that the position feedback message has a resolution of at least 1%. You can set the sampling time at the CYCLE parameter of the Self-Tuner and of the controller. It has to agree with the time difference between two calls (cycle time of the watchdog interrupt OB under observance of the pulse scalings).

When used with the controller modules FM355/455 the time between the Self-Tuner calls (sampling time) has to be approximately equal to the execution times of the controllers on the FM. The execution time is specified in the parameter assignment tool of the FMs under Options > Module parameters > Resulting Cycle Time.

Normalization of the controller gain under Standard PID Control V5

The controller blocks in Standard PID Control V5 operate with a normalized controller gain which is calculated as follows:

GAIN (normalized) = GAIN (physical) * (NM_PVHR - NM_PVLR)/100

This results in a normalization factor which you have to enter at the parameter NORM_FAC of the Self-Tuner:

NORM_FAC = (NM_PVHR - NM_PVLR)/100

This results in the normalized controller gain being applied at the GAIN output of the Self-Tuner. In the examples for Standard PID Control V5 this point is programmed in the start-up branch of the blocks TUN_CON_C, TUN_CON_P and TUN_CON_S.

NORM_FAC remains at its default assignment of 1 in the case of controllers with a physical controller gain.

Execution phases of the Self-Tuner

PHASE = 0	Automatic mode: Closed-loop control with preset parameters after generation of an instance DB. No further functions possible. MAN_ON, LMNS_ON or ADAPT1ST is used to exit the phase.
PHASE = 1	Controller optimization: Heating off and waiting for a setpoint step
PHASE = 2	Controller optimization: Motor actuating measurement, starting up to a constant control valve setting and search for point of inflection
PHASE = 3	Controller optimization: Controlling with a PI coarse controller
PHASE = 4	Automatic mode: Normal closed-loop control with/without control zone
PHASE = 5	Automatic mode, structure changeover at setpoint step: Integral action de-activated, P(D) control
PHASE = 6	Automatic mode, structure changeover at setpoint step: Prediction-control specification of the manipulated variable (only for step controller with position feedback)
PHASE = 7	Manual mode

Function activation

Function	Start bit	Enable bit	Start condition: PHASE	Start condition: Setpoint step at SP
Manual mode	---	MAN_ON LMNS_ON	0 to 7	---
First adaption with measurement of the motor actuating time	ADAPT1ST	---	0, 4	Pos.: > MIN_STEP
First adaption without measurement of the actuating time of the final control element or on-line adaption (only possible for step controller without position feedback)	ADAPT_ON	---	4	Pos.: > MIN_STEP
Structure changeover	---	STRUC_ON	4	Pos.: > MIN_STEP Neg.: > MIN_STEP * RATIO_FAC
Save controller parameters	SAVE_PAR	---	2 to 7	---
Load controller parameters back	UNDO_PAR	---	4	---
Change controller parameters	LOAD_PAR	---	4	---

Note

Start bits are reset after the function has been started successfully. Enable bits remain set.

Input parameters

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
SP	REAL	Setpoint	Technical range of values	0.0	The setpoint is entered at the input SP. It is interconnected to the controller via SP_OUT. Note that the setpoint must lie within the setpoint limits of the controller.
PV	REAL	Process variable	Technical range of values	0.0	PV is interconnected to the process variable on the controller.
MAN	REAL	Manual value	-100.0 ... 100.0 (%)	0.0	The manual value is entered at the input MAN. If MAN_ON = TRUE, the manual value MAN is transferred to the manipulated variable of the controller. If MAN_ON = TRUE and MAN = 0.0, the heating function is off. If MAN_ON = FALSE, the manipulated variable of the controller is specified.
LMNR	REAL	Repeated manipulated value	0.0 ... 100.0 (%)	0.0	LMNR is interconnected to the position feedback of the controller (if it exists).
MIN_STEP	REAL	Minimal setpoint step	Recommended:≥ 10 % of the required setpoint/process variable area	10.0	You can specify the minimal setpoint step size at the input MIN_STEP above which a first adaption or adaption can be carried out. If the structure changeover is activated (STRUC_ON=TRUE), the setpoint step must be greater than MIN_STEP in order for the structure changeover to be activated. MIN_STEP should not be configured to less than 10% of the operating range of the setpoint value or process variable.
LHLM_TUN	REAL	Manipulated value high limit on self-tuning	0.0 ... 100.0 (%)	80.0	The setpoint step for the first adaption is entered at the input LHLM_TUN. During on-line adaption the setpoint step is calculated automatically.

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
NORM_FAC	REAL	Proportional gain normalizing factor	Dimensionsless	1.0	A normalizing factor is applied at controllers with a normalized controller gain (Standard PID Control V5). The normalizing factor is calculated on the basis of the range limits and is interconnected to the input NORM_FAC (refer to the examples with Standard PID Control V5 in the FBs TUN_CON_C, TUN_CON_P and TUN_CON_S).
PULSE_TM	TIME	Minimal pulse time	≥ 1 ms	0 ms	PULSE_TM is interconnected to the minimal pulse time of the controller.
C_LMNUP	BOOL	Controller manipulated signal up		FALSE	C_LMNUP is interconnected to the manipulated variable signal QLMNUP on the controller.
C_LMNDN	BOOL	Controller manipulated signal down		FALSE	C_LMNDN is interconnected to the manipulated variable signal QLMNDN on the controller.
MAN_ON	BOOL	Manual mode on		TRUE	If MAN_ON = TRUE, the setpoint of the controller is specified via the input MAN. If MAN_ON = FALSE, the manipulated variable of the controller is specified.
LMNR_HS	BOOL	High limit signal of repeated manipulated value		FALSE	LMNR_HS is interconnected to the limit signal LMNR_HR on the controller.
LMNS_ON	BOOL	Manipulated signals on		FALSE	If LMNS_ON is set, you can use the parameters LMNUP and LMNDN to operate the actuating signals of the controller.
LMNUP	BOOL	Manipulated signal up		FALSE	You can use LMNUP to operate the actuating signal OPEN of the controller.
LMNDN	BOOL	Manipulated signal down		FALSE	You can use LMNDN to operate the actuating signal CLOSE of the controller.

Parameters	Data type	Comment	Permitted range of values	Initial	Explanation
PID_ON	BOOL	PID mode on		TRUE	You can specify at the input PID_ON whether the optimized controller is to operate as a PI or as a PID controller: PID controller: PID_ON = TRUE PI controller: PID_ON = FALSE However, it is possible that a PI controller is designed at some types of process although PID_ON = TRUE.
STRUC_ON	BOOL	Variable structure control for setpoint steps		FALSE	If STRUC_ON = TRUE, a structure changeover (from PI(D) to P(D) controlling) ensures at setpoint steps that overshoots are avoided to a great extent. The input STRUC_ON may not be set = TRUE, if the proportional or derivative action is fed into the feedback loop at the controller.
WRITE_DIS	BOOL	Writing to FM is not possible		FALSE	In connection with the controller module the Self-Tuner has to know whether the driver block is ready to send data to the FM. This is the case if LOAD_OP = LOAD_PAR = FALSE. The result of the OR operation of LOAD_OP and LOAD_PAR is interconnected to WRITE_DIS.
CYCLE	TIME	Sample time	≥1 ms	100 ms	The sample time has to be configured at the input CYCLE. CYCLE must agree with the sample time of the controller to be set and with the time difference between two calls (cycle time of the watchdog interrupt OB under observance of the pulse scalings).

Output parameters

Parameters	Data type	Comment	Initial	Explanation
MAN_OUT	REAL	Manual value output	0.0	MAN_OUT is connected to the manual input value of the controller. If MAN_ON = TRUE, the Self-Tuner outputs the value which is specified at the input MAN at the output MAN_OUT.
SP_OUT	REAL	Setpoint output	0.0	SP_OUT is connected to the setpoint input of the controller. The Self-Tuner writes the setpoint specified at the input SP to the output SP_OUT. At manual operation MAN_ON = TRUE the setpoint is tracked to the process variable: SP_OUT = PV.
GAIN	REAL	Proportional gain	1.5	GAIN is interconnected to the proportional-action coefficient (controller gain) of the controller.
TI	TIME	Reset time Integral time	1 h	TI is interconnected to the integral time (reset time) of the controller.
TD	TIME	Derivative time	0ms	TD is interconnected to the derivative time (rate time) of the controller.
TM_LAG	TIME	Time lag	1 s	TM_LAG is interconnected to the time lag of the controller differentiator.
MTR_TM	TIME	Motor actuating time	30s	MTR_TM is interconnected to the motor actuating time of the controller.
DEADB_W	REAL	Dead band width	0.0	DEADB_W is interconnected to the dead band width of the controller.
PHASE	INT	Phase 0 to 7	0	PHASE displays the current execution phase of controller optimization. During the optimization of Standard PID Control V5 the output PHASE is interconnected to the output PHASE of the FB PID_CP or PID_ES.
STATUS_H	INT	Status heating	0	STATUS_H displays optimization results for the search of the point of inflection during the heating process.
STATUS_D	INT	Status controller design	0	STATUS_D displays optimization results for the controller design during the heating process.
QMAN_ON	BOOL	Manual mode on	FALSE	QMAN_ON is interconnected to the manual/automatic changeover of the controller.
QLMNS_ON	BOOL	Manipulated signals on		QLMNS_ON is interconnected to the parameter "Manipulated signals on" of the controller
QLMNUP	BOOL	Manipulated signal up		QLMNUP is interconnected to the parameter "Manipulated signal up" of the controller
QLMNDN	BOOL	Manipulated signal down		QLMNDN is interconnected to the parameter "Manipulated signal close" of the controller

Parameters	Data type	Comment	Initial	Explanation
QI_SEL	BOOL	Integral action on	TRUE	QI_SEL is interconnected to the on/off switch of the controller integral action I_SEL. In the case of FM355/455 TI has to be switched to zero depending on QI_SEL: If QI_SEL=FALSE, then TI=0.
QD_SEL	BOOL	Derivative action on	FALSE	QD_SEL is interconnected to the on/off switch of the controller derivative action D_SEL. In the case of FM355/455 TD has to be switched to zero depending on QD_SEL: If QD_SEL=FALSE, then TD=0.
QWRITE	BOOL	Self-Tuner writes parameters to the PID controller	FALSE	QWRITE is interconnected to the inputs LOAD_PAR and LOAD_OP of the driver block PID_CS of FM355/455. The result is that the records of the controller channel are downloaded to the FM.
PROCESS	STRUCT			
GAIN	REAL	Process gain	999.0	The process gain is displayed at the output PROCESS.GAIN.
TM_LAG1	REAL	Process slow time constant	0.0	The process slow time constant is displayed at the output PROCESS.TM_LAG1.
TM_LAG2	REAL	Process fast time constant	0.0	The process fast time constant is displayed at the output PROCESS.TM_LAG2.
PV00	REAL	Process steady state value for zero input	0.0	The steady state of the process is displayed at the output PROCESS. PV00 The steady state is the steady-state process variable at Setpoint = zero. This corresponds to the ambient tempera ture at a cooling plant.
KIG	REAL	Maximal ascent ratio of PV	0.0	The maximal ascent ratio of the process variable during heating is displayed at the output PROCESS.KIG.
MTR_TM	REAL	Motor actuating time	0.0	The motor actuating time is displayed at the output MTR_TM.
	END_ST			
PI_CON	STRUCT			
GAIN	REAL	PI proportional gain	1.5	PI_CON.GAIN displays the proportional gain of the PI controller. You can improve the value manually and make it effective by setting LOAD_PAR.

Parameters	Data type	Comment	Initial	Explanation
TI	REAL	PI reset time PI integration time	3600.0	PI_CON.TI displays the integration time of the PI controller. You can improve the value manually and make it effective by setting LOAD_PAR.
	END_ST			
PID_CON	STRUCT			
GAIN	REAL	PID proportional gain	0.0	PID_CON.GAIN displays the proportional gain of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
TI	REAL	PID reset time PID integration time	0.0	PID_CON.TI displays the integration time of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
TD	REAL	PID derivative time	0.0	PID_CON.TD displays the derivative time of the PID controller. You can improve the value manually and make it effective by setting LOAD_PAR.
	END_ST			
PI_CON_OLD	STRUCT			
GAIN	REAL	PI old proportional gain	1.5	The old proportional gain of the PI controller is displayed at the output PI_CON_OLD.GAIN.
TI	REAL	PI old reset time PI old integration time	3600.0	The old integration time of the PI controller is displayed at the output PI_CON_OLD.TI.
	END_ST			
PID_CON_OLD	STRUCT			
GAIN	REAL	PID old proportional gain	0.0	The old proportional gain of the PID controller is displayed at the output PID_CON_OLD.GAIN.
TI	REAL	PID old reset time PID old integration time	0.0	The old integration time of the PID controller is displayed at the output PID_CON_OLD.TI.
TD	REAL	PID old derivative time	0.0	The old derivative time of the PID controller is displayed at the output PID_CON_OLD.TD.
	END_ST			
TU	REAL	Time lag	0.0	The determined time delay is displayed at the output TU.
TA	REAL	Recovery time	0.0	The determined recovery time is displayed at the output TA.

Throughput parameters

Parameters	Data type	Comment	Initial	Explanation
ADAPT1ST	BOOL	First adaption	FALSE	When you carry out the first optimization of the process or do not wish to use the results of previous optimizations such as setpoint steps or filter time constants, when you carry out a first adaption. To this purpose set ADAPT1ST = TRUE. Then the Self-Tuner carries out an initialization run and in the process sets the initial values described in the initialization section. The Self-Tuner resets ADAPT1ST back to FALSE. It does not make sense to set ADAPT1Self-Tuner to true at a restart of the CPU in OB100.
ADAPT_ON	BOOL	Switch on on-line adaption with next setpoint step	FALSE	If you want to trigger on-line adaption, set ADAPT_ON = TRUE. If the subsequent setpoint step is greater than MIN_STEP, identification is started (PHASE = 2). The Self-Tuner resets ADAPT_ON back to FALSE.
STEADY	BOOL	Steady state reached	FALSE	When the process variable has reached a steady state, you can use STEADY to terminate optimization prematurely, although the Self-Tuner is still in PHASE 3. You can also use STEADY to carry out a renewed controller design in steady state, after the Self-Tuner has passed into PHASE 4.
UNDO_PAR	BOOL	Undo change of controller parameters	FALSE	The old controller parameters are transferred to the current controllers parameters and are active immediately depending on PID_ON. PI_CON = PI_CON_OLD PID_CON = PID_CON_OLD The function for undoing changes can only be used during automatic mode (PHASE 4).
SAVE_PAR	BOOL	Save current controller parameters	FALSE	The current controller parameters are transferred to the old controller parameters. PI_CON_OLD = PI_CON PID_CON_OLD = PID_CON
LOAD_PAR	BOOL	Activate changed controller parameters	FALSE	The modified parameters PI_CON and PID_CON are transferred to the controller and are thus active.

4 Examples

4.1 Overview of examples

An example project is provided for each control product with the Self-Tuner.

Control product	Corresponding example product
Standard PID Control V5	TunEx_StdPIDCon
Modular PID Control	TunEx_ModPIDCon
FM355 C/S	TunEx_FM355
FM455 C/S	TunEx_FM455

Each example project contains 5 examples:

- **Ex1_plastic**

  4 heating zones as they occur at an extruder or an injection molding plant are simulated in this example. It contains a control system for the automatic optimization and operation of all 4 heating zones. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generator.

- **Ex2_furnace**

  A furnace as it is used, for example, for sintering processes is simulated in this process. The process has two actuating elements: A semiconductor relay for the heating process and an integral-action control (butterfly) valve for the cooling process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generator in combination with a step controller.

- **Ex3_simple_c**

  This example contains a simple control loop which consists of a PID controller and a VZ2 element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller.

- **Ex4_simple_s**

  This example contains a simple control loop which consists of a PID controller and a VZ2 element with an integrating actuating element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a PID step controller.

- **Ex5_simple_p**

  This example contains a simple control loop which consists of a PID controller and a VZ2 element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generator (semiconductor relay or mechanical contactor).

Note

You still have to copy the controller blocks of the control packages to the examples for the software controllers.
You have to use HW Config to adapt the hardware structure in the examples for the controller modules.

4.2 Interconnection between Self-Tuner and controller

The examples include multi-instance FBs, which contain the controller and the Self-Tuner already interconnected.

FB TUN_CON_C	Self-Tuner with continuous PID controller
FB TUN_CON_P	Self-Tuner with continuous PID controller and pulse generator
FB TUN_CON_S	Self-Tuner with step controller

Interface connection "Continuous controller"

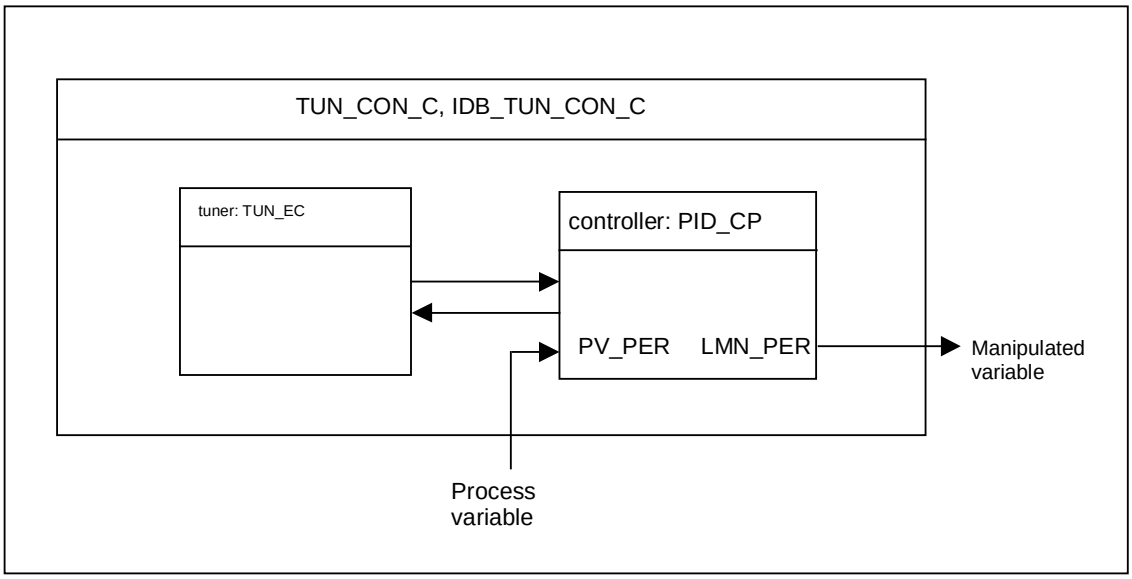


Figure: Multi-instance FB TUN_CON_C (Self-Tuner with a continuous controller)

STL example for supplying process and manipulated variables of the controller

```
CALL #controller
     ...
     PV_PER := PEWx          //Process variable
     LMN_PER := PAWy         //Manipulated variable
     ...
```

Interface connection "Step controller"

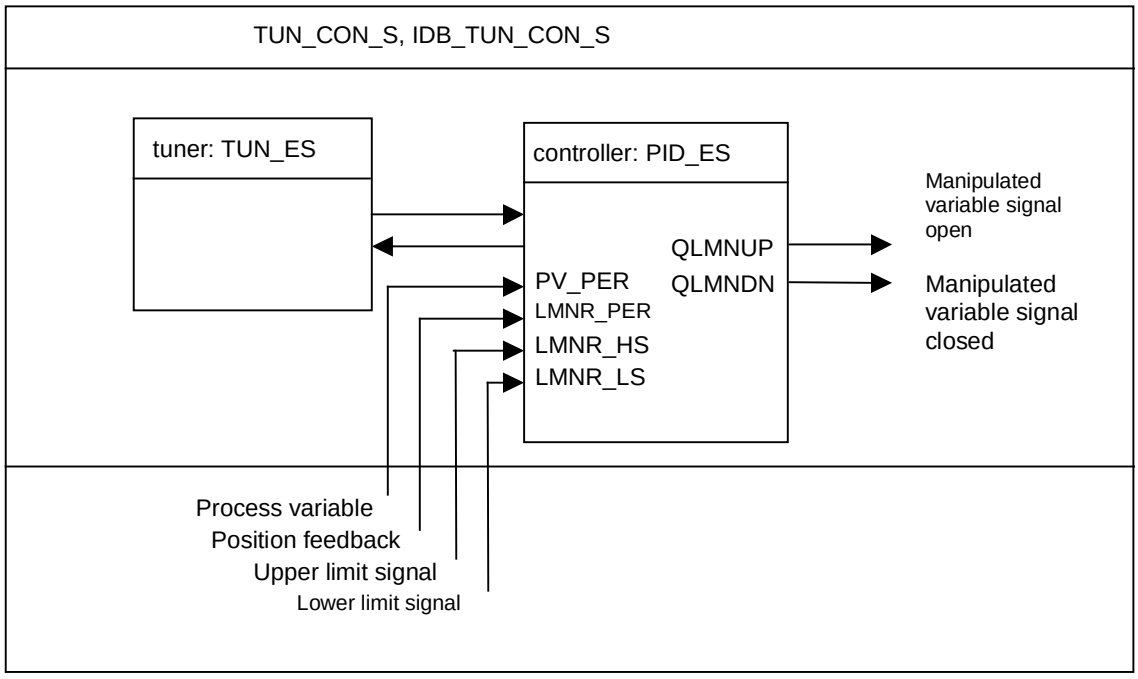


Figure: Multi-instance FB TUN_CON_S (Self-Tuner with a step controller)

STL example for supplying process and manipulated variable signals of the controller

```
            CALL #controller
                 ...
                 PV_PER      := PEWx1      //Process variable
                 LMNR_PER:= PEWx2 //Position feedback value
                 LMNR_HS     := Ex3.y1     //Upper limit signal
                 LMNR_LS     := Ex4.y2     //Lower limit signal
...
QLMNUP  := Ax5.y3        //Actuating signal up
                 QLMNDN      := Ax6.y4     //Actuating signal down
                 ...
```

Interface connection "Continuous controller with pulse generator"

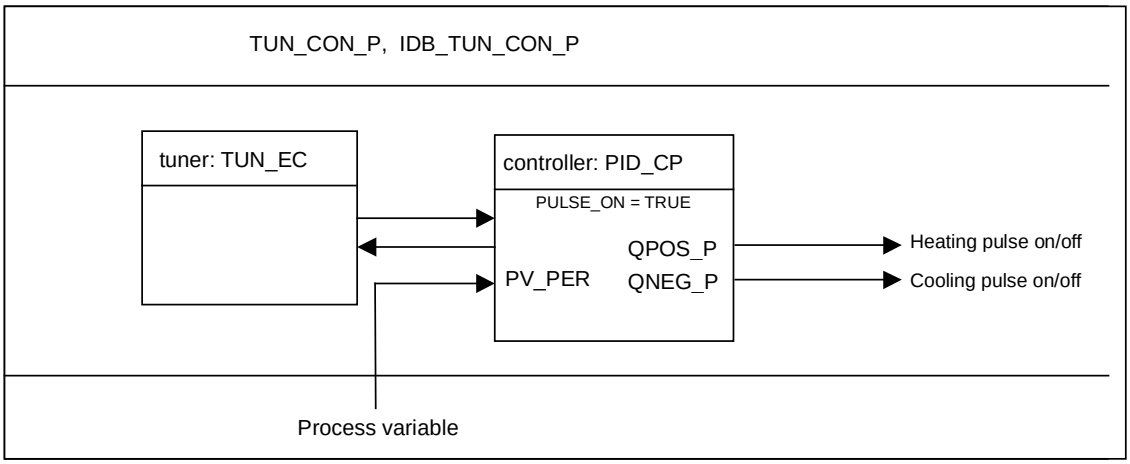


Figure: Multi-instance FB TUN_CON_P (Self-Tuner with a step controller)

STL example for supplying process and manipulated variable signals of the controller

```
            CALL #controller
                 ...
                 PV_PER      := PEWx1       //Process variable
...
QPOS_P   := Ax2.y1       //Heating on/off
                 QNEG_P      := Ax3.y2      //Cooling on/off
                 ...
```

4.3 Example from the plastics industry: Ex1_plastic

Overview

4 heating zones as they occur at an extruder or an injection molding plant are simulated in this example. It contains a control system for automatic optimization (FB AUTO_TUN) and an operation control for all 4 heating zones (FB C_ADMIN). It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generator.

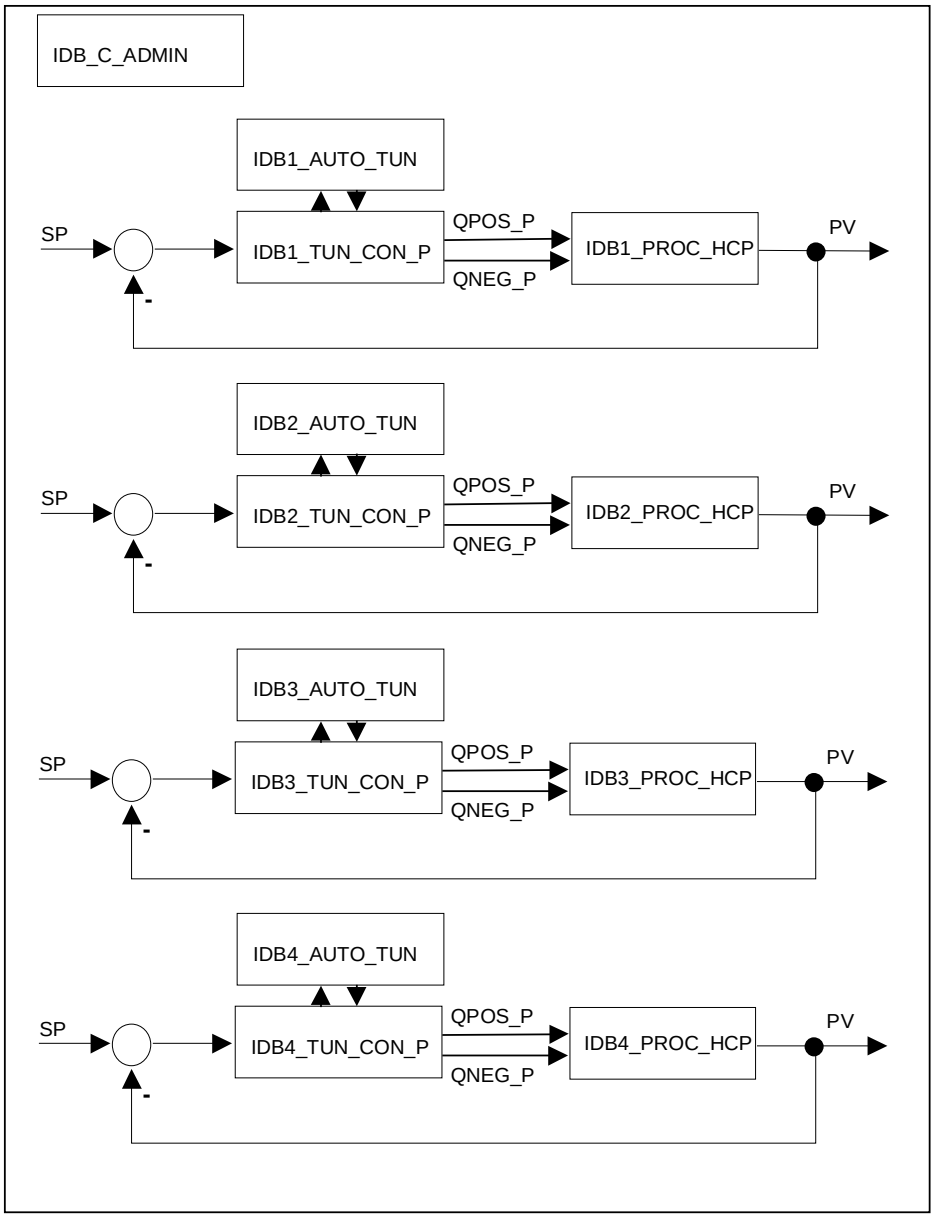


Figure: Block diagram

Program structure

The blocks IDB1_TUN_CON_P, ... IDB4_TUN_CON_P are instance DBs of FB TUN_CON_P, which contains the Self-Tuner, the controller and the pulse generator. The individual instances are called together with the instance DBs of the controlled system IDB1_PROC_HCP, ... IDB4_PROC_HCP in OB35.

The blocks IDB1_AUTO_TUN, ... IDB4_AUTO_TUN are instance DBs of FB AUTO_TUN, which contains the automatic control of the Self-Tuner for optimization.

The block IDB_C_ADMIN is the instance DB of FB C_ADMIN that realizes operation control for all 4 zones with few switches.

Process block for simulating a temperature heating zone

The block simulates a typical temperature process for heating and cooling as it can occur as a control zone in a extruder, an injection molding machine, a tempering machine or as a separate furnace in practice. The block is used in the following examples to build up the process.

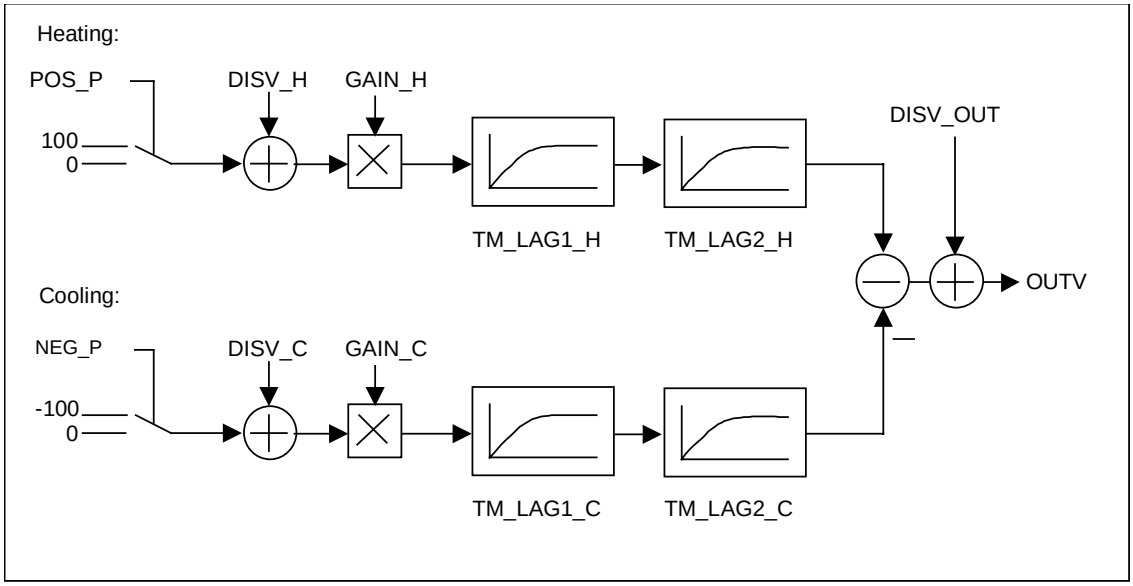


Figure: Block diagram of the process

Parameters

POS_P	Positive pulse	Binary input signal heating
NEG_P	Negative pulse	Binary input signal cooling
DISV_H	Disturbance variable heating	Disturbance variable heating
DISV_C	Disturbance variable cooling	Disturbance variable cooling
DISV_OUT	Disturbance variable by output	Disturbance variable at output (for example ambient temperature)
GAIN_H	Process gain heating	Process gain heating
GAIN_C	Process gain cooling	Process gain cooling
TM_LAG1_H	Time lag 1 heating	Time lag 1 heating
TM_LAG2_H	Time lag 2 heating	Time lag 2 heating
TM_LAG1_C	Time lag 1 cooling	Time lag 1 cooling
TM_LAG2_C	Time lag 2 cooling	Time lag 2 cooling
OUTV	Output variable	Output variable (for example control zone temperature)

The binary input signals are converted into continuous floating-point values (-100, 0, 100). The process values are controlled by two first-order time delays after addition of a disturbance variable (for example ambient temperature) and multiplication with the process gain. This process is carried out separately for heating and cooling.

During initialization the output variable is set to OUTV = DISV_H*GAIN_H - DISV_C*GAIN_C + DISV_OUT.

Automatic control of the Self-Tuner with the FB AUTO_TUN

Input parameters:

SP	Setpoint specification by C_ADMIN
MAN_ON	Activate manual operation by C_ADMIN
MAN	Manual specification by C_ADMIN (in the example = 0.0)
PV	Process variable interconnected to the Self-Tuner
LMN	Manipulated variable interconnected to the Self-Tuner
PHASE	PHASE interconnected to the Self-Tuner
COOL_ON	COOL_ON interconnected to the Self-Tuner
JUMP_EN	All other zones are ready for setpoint specification, is operated by C_ADMIN
COOL_EN	All other zones are ready for cooling identification, is operated by C_ADMIN
END_EN	All other zones are ready to terminate optimization, is operated by C_ADMIN
STAT_TM	Time for steady-state condition
STAT_D_SP	Setpoint-process-variable range for steady state condition
STAT_D_LMN	Manipulated variable range for steady-state condition
DELTA_SP	Automatic setpoint step specification for optimization at operating point
CYCLE	Sampling time interconnected to the Self-Tuner

Output parameters:

STEP	Optimization step display
QTUN_END	Optimization terminated
QTUNBREAK	Optimization interrupted
SP_OUT	Setpoint specification, interconnected to the Self-Tuner
QMAN_ON	Activate manual mode, interconnected to the Self-Tuner
MAN_OUT	Manual value, interconnected to the Self-Tuner
QADAPTON	Adaption request, interconnected to the Self-Tuner
QCOOL_ON	Start cooling identification, interconnected to the Self-Tuner
QCOM_RST	First adaption request, interconnected to the Self-Tuner
QJUMP_EN	Ready for setpoint jump, is evaluated by C_ADMIN
QCOOL_EN	Ready for cooling identification, is evaluated by C_ADMIN
QEND_EN	Ready for optimization end, is evaluated by C_ADMIN

Throughput parameters:

TUN_O	Optimization request by C_ADMIN
RESET	First adaption request by C_ADMIN

Automatic optimization is carried out in the following steps:

- **STEP=0**

  Normal control operation with waiting for an optimization request
  If TUN_ON is set, defaults are assigned to some outputs and the next stage is activated.

- **STEP=1**

  For first adaption the system switches over to manual mode with MAN=0.0.
  The next stage is activated immediately.

- **STEP=2**: Waiting for steady state

  The steady state is recognized when the setpoint and the process variable remain within the steady-state range with a width of 2*STAT_D_SP for the duration of STAT_TM and the manipulated variable lies within the steady state range of 2*STAT_D_LMN. The next stage is activated if JUMP_EN=1, meaning that all other zones are also stationary. In the case of a first adaption the ADAPT1ST bit is set, otherwise the ADAP_ON bit of the Self-Tuner is set.

- **STEP=3**: Setpoint step specification

  In the case of a first adaption (RESET = 1) the system identifies with a setpoint step from the ambient temperature to the set setpoint profile. If the control system is still in manual mode, the system waits until the user changes over to automatic mode. If the setpoint is then still within the steady-state range, for example at an adaption at the operating point (PV-STAT_D_SP < SP < PV+STAT_D_SP), the value DELTA_SP is added automatically to the current setpoint. The next stage is then activated.

- **STEP=4**: Waiting for the end of heating optimization (PHASE=4)

  If the Self-Tuner is in PHASE=4 (heating optimization completed and if all other zones have also completed heat optimization (COOL_EN=1), the system jumps to Stage 5 in the case of zones with cooling COOL_ON=1 - in the case of zones without cooling to Stage 9.

- **STEP=5**: Confirmation that cooling identification has been started (PHASE=3)

  The system jumps immediately to Stage 6 when PHASE=3 has been reached.

- **STEP=6**: Waiting for the end of cooling identification (PHASE=4)

  After PHASE=4 has been reached, the system jumps to Stage 9, also in the case of cooling zone.

- **STEP=7**:

  Free for the user

- **STEP=8**:

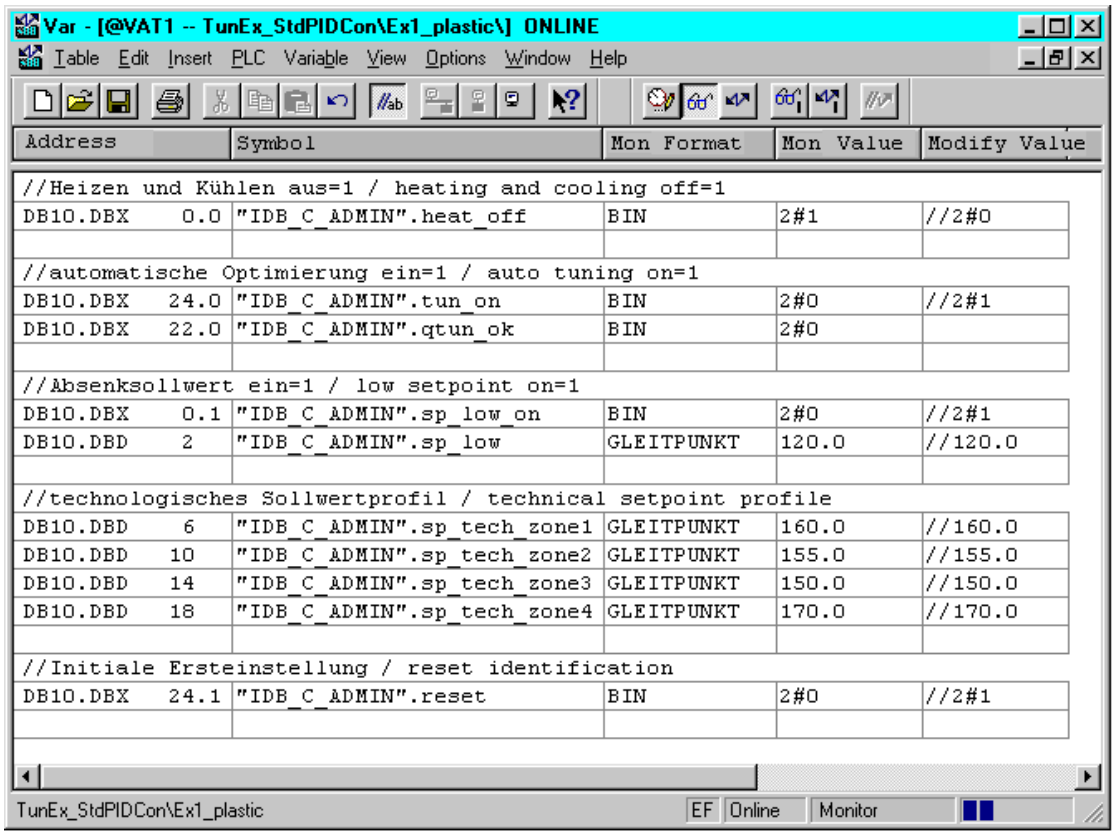
  Free for the user

- **STEP=9**: Waiting for the end of all zones (END_EN=1)

  The system jump backs to Stage 0 when all the zones have reached Stage 9. An automatically applied setpoint DELTA_SP is canceled in the process (setpoint step to the set setpoint profile).

Operator control and monitoring

A group operation of all 4 zones is possible by means of the block C_ADMIN. You can carry out the operation in the variable declaration table VAT_PLASTIC.



Address	Symbol	Mon Format	Mon Value	Modify Value
//Heizen und Kühlen aus=1 / heating and cooling off=1				
DB10.DBX 0.0	"IDB_C_ADMIN".heat_off	BIN	2#1	//2#0
//automatische Optimierung ein=1 / auto tuning on=1				
DB10.DBX 24.0	"IDB_C_ADMIN".tun_on	BIN	2#0	//2#1
DB10.DBX 22.0	"IDB_C_ADMIN".qtun_ok	BIN	2#0	
//Absenksollwert ein=1 / low setpoint on=1				
DB10.DBX 0.1	"IDB_C_ADMIN".sp_low_on	BIN	2#0	//2#1
DB10.DBD 2	"IDB_C_ADMIN".sp_low	GLEITPUNKT	120.0	//120.0
//technologisches Sollwertprofil / technical setpoint profile				
DB10.DBD 6	"IDB_C_ADMIN".sp_tech_zone1	GLEITPUNKT	160.0	//160.0
DB10.DBD 10	"IDB_C_ADMIN".sp_tech_zone2	GLEITPUNKT	155.0	//155.0
DB10.DBD 14	"IDB_C_ADMIN".sp_tech_zone3	GLEITPUNKT	150.0	//150.0
DB10.DBD 18	"IDB_C_ADMIN".sp_tech_zone4	GLEITPUNKT	170.0	//170.0
//Initiale Ersteinstellung / reset identification				
DB10.DBX 24.1	"IDB_C_ADMIN".reset	BIN	2#0	//2#1

Figure: Variable declaration table VAT_PLASTIC

All the heating or cooling systems are switched off with the switch heat_off. If optimization is aborted, the parameter IDBx_AUTO_TUN.QTUNBREAK, $1 \leq x \leq 4$ is set. If one of these parameters is set, the heating is switched off in the FB C_ADMIN and the user has to carry out optimization.

If tun_on is set and heat_off is reset, the following steps are carried out automatically successfully:

- Waiting until a steady state is reached
- Apply setpoint step (during a first adaption the set setpoint profile is applied, during an on-line adaption IDBx_AUTO_TUN.DELTA_SP, $1 \leq x \leq 4$ to which the current setpoint is added.)
- In the case of zones with active cooling a cooling identification is triggered automatically
- After controller optimization has been terminated (PHASE=4), a setpoint return is carried out to the set setpoint profile in the case of on-line adaption.
- After all zones have been optimized, IDB_C_ADMIN.tun_on is reset. Optimization was successful if IDB_C_ADMIN.qtun_ok is set.

If the switch IDB_C_ADMIN.sp_low_on is set, a constant setpoint profile with IDB_C_ADMIN.sp_low is set. If the switch is set to zero, the system controls again to the process-related setpoint profile.

The adaption can be aborted by setting IDB_C_ADMIN.heat_off or by resetting IDB_C_ADMIN.tun_on.

If IDB_C_ADMIN.reset is set, the heating and cooling systems are switched off until all the zones have cooled down to the ambient temperature. In the simulation you can accelerate the cooling down process by means of a restart (warm start). A first adaption with setpoint step to the set setpoint profile is then carried out. The switch IDB_C_ADMIN.reset should be set, if optimization is carried out for the first time on the concrete plant or when all the data lie at their default values after a general reset and downloading via EEPROM.

Taking the example into operation

- **Software controller**

  Copy the controller blocks into the program Ex1_plastic of your example project.

Control product	Blocks
Standard PID Control V5	FB PID_CP
Modular PID Control	FB DEADBAND, FB PID and FB LMNGEN_C Do not overwrite the FB 43 PULSEGEN with the FB PULSEGEN from Modular PID Control.

  Use HW Config to set the cycle time of the OB35 to 20ms. If a time-out occurs in the watchdog interrupt level, increase the cycle time. In this case the simulation is then executed more slowly. If you control a real process, the cycle time of the OB35 must agree with the sampling times IDB1_TUN_CON_P.CYCLE_P, ... IDB4_TUN_CON_P.CYCLE_P and IDB1_AUTO_TUN, ... IDB4_AUTO_TUN.

- **Controller module FM355/455**

  Use HW Config to adapt the configuration of the hardware structure from the example to your real plant. Use the cycle time of the OB35 and the complete FM parameter configuration.

Use the SIMATIC Manager to download the program Ex1_plastic to the CPU and carry out a restart (warm start).

Under "Simatic STEP 7" use the Start button to start the tool belonging to the controller used and open the blocks IDB1_TUN_CONT_P, ... IDB4_TUN_CONT_P on-line. At the curve recorders set Acquisition cycle= 1s and Length of the time axis= 500s under "Settings ...". Set the ranges of the setpoint and process variable to 0.0 to 200.0. Set the range of the manipulated variable to –100.0 to 300.0.

Open the variable declaration table "VAT_PLASTIC" and there set tun_on = TRUE as well as heat_off = FALSE. A fully automatic optimization of all 4 zones is carried out.

You can now continue to operate the program as described in the section "Operator control and monitoring".

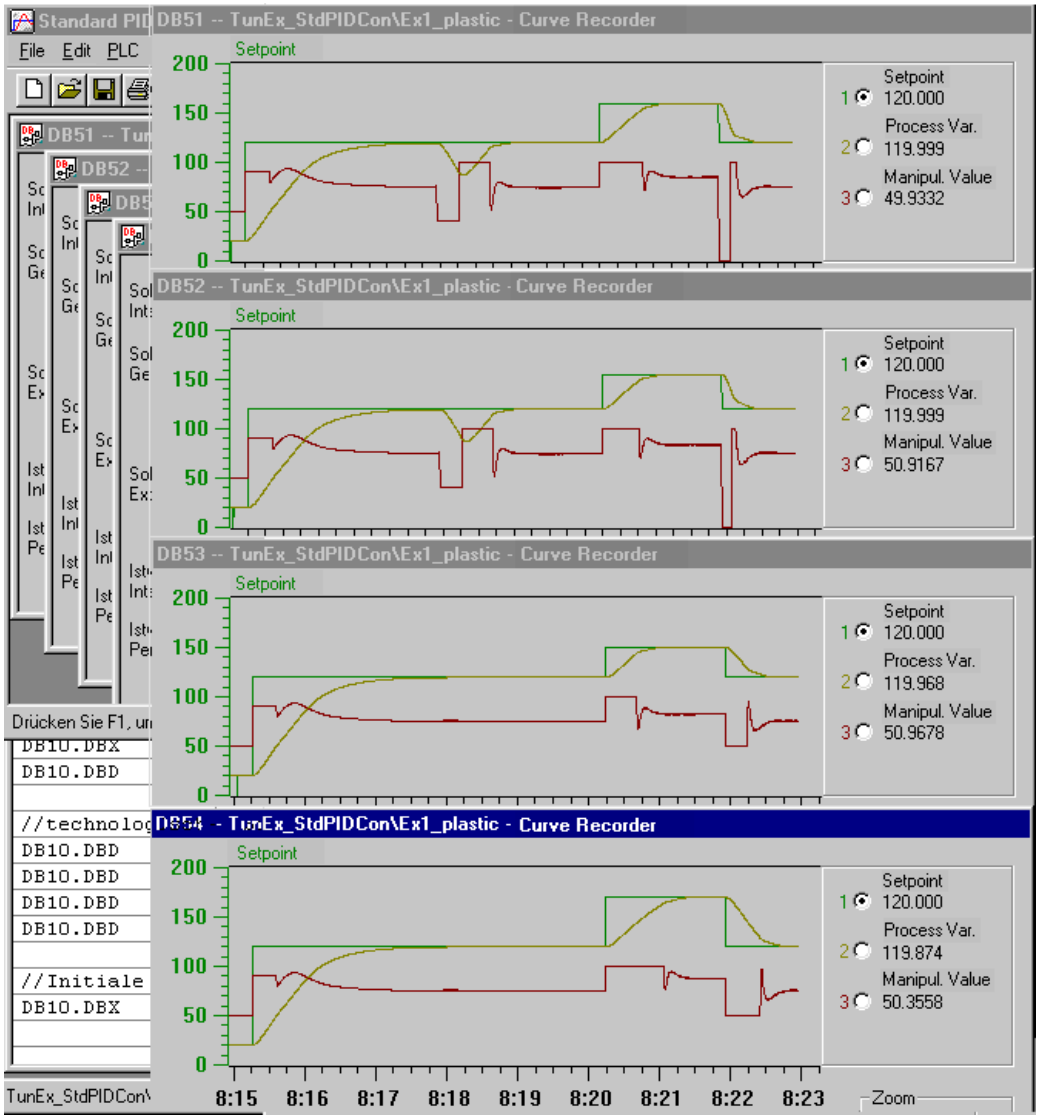


Figure: Controller optimization with Standard PID Control V5

The above figure shows the controller optimization of all 4 zones when heating to the operating point. The first adaption was carried out from the ambient temperature 20 ^{0}C to 120 ^{0}C. After all the zones with the heating identification were completed, the cooling identification at the cooling zones was started automatically. After controller optimization the system heated up to the process-related setpoint profile and then cooled down to the lowering setpoint.

4.4 Example of a furnace control system: Ex2_furnace

Overview

A furnace as it is used, for example, for sintering processes is simulated in this process. The process has two actuating elements: A semiconductor relay for the heating process and an integral-action control (butterfly) valve for the cooling process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generator in combination with a step controller.

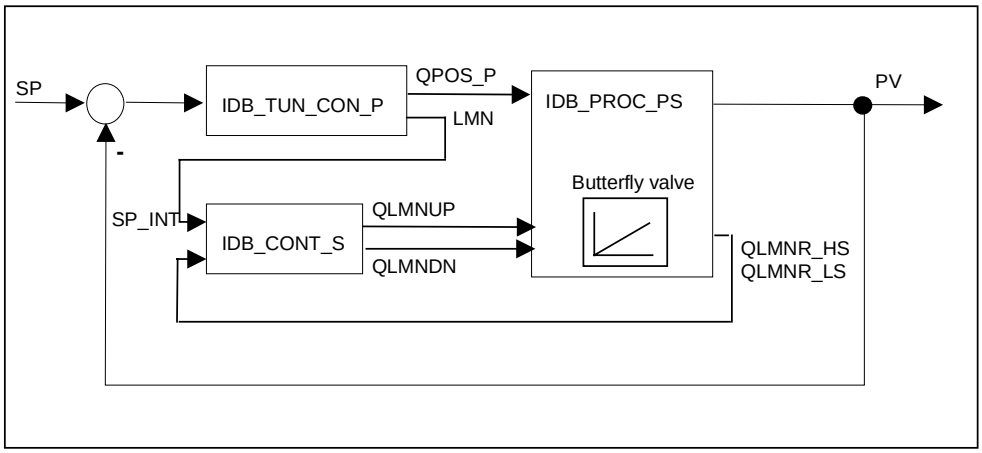


Figure: Block diagram

Program structure

The block IDB_TUN_CON_P is an instance DB of FB TUN_CON_P, which contains the Self-Tuner, the controller and the pulse generator. The control valve for cooling is controlled by a step controller. In the examples it is always the CONT_S from the Standard Library of Step7. The step controller is used as a pure positioner. This means that the manipulated variable of the continuous controller is passed to the setpoint of the step controller. This is configured as a pure P controller with

- GAIN = -RATIOFAC (Heating/cooling gain ratio) and
- TI = 0 (Integral action)

Process block for simulating a temperature process with different actuating elements

The block simulates a temperature process for heating and cooling with different actuating elements. During heating the process expects a pulse-shaped input signal, as is the case for a semiconductor relay or mechanical contactor. During cooling an integrating actuating element (butterfly valve) acts on the process.

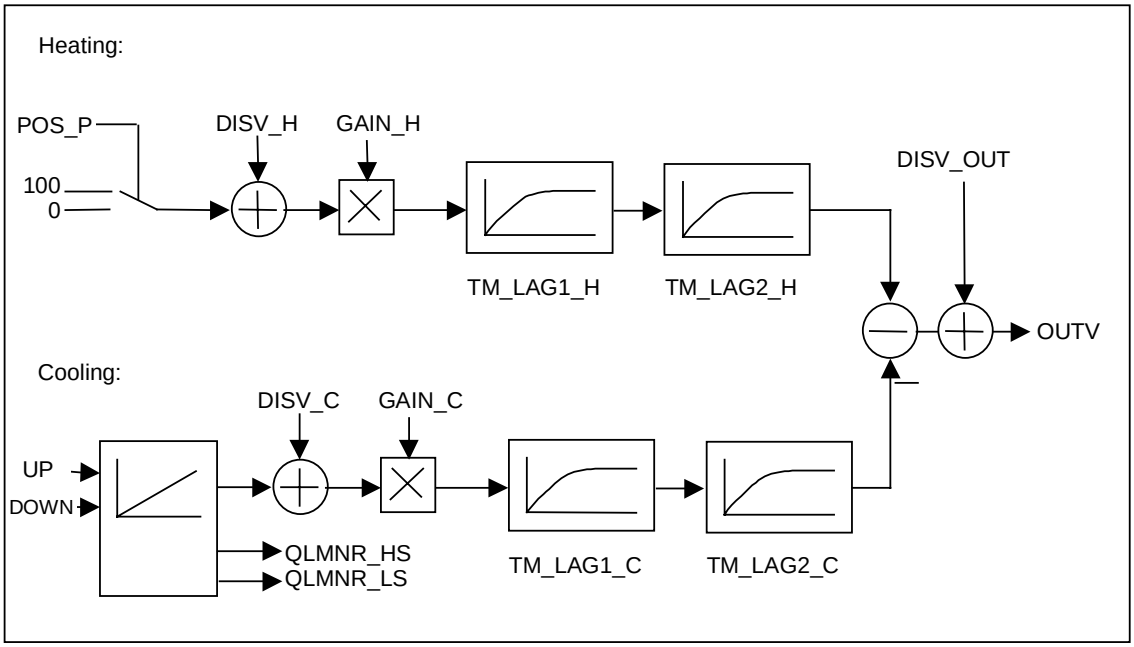


Figure: Block diagram of the process

Parameters

POS_P	Positive pulse	Binary input signal heating
UP	Cooling up	Cooling valve open
DOWN	Cooling down	Cooling valve closed
DISV_H	Disturbance variable heating	Disturbance variable heating
DISV_C	Disturbance variable cooling	Disturbance variable cooling
DISV_OUT	Disturbance variable by output	Disturbance variable at output (for example ambient temperature)
GAIN_H	Process gain heating	Process gain heating
GAIN_C	Process gain cooling	Process gain cooling
MTR_TM	Motor actuating time	Motor actuating time of the cooling valve
TM_LAG1_H	Time lag 1 heating	Time lag 1 heating
TM_LAG2_H	Time lag 2 heating	Time lag 2 heating
TM_LAG1_C	Time lag 1 cooling	Time lag 1 cooling
TM_LAG2_C	Time lag 2 cooling	Time lag 2 cooling
OUTV	Output variable	Output variable (for example control zone temperature)

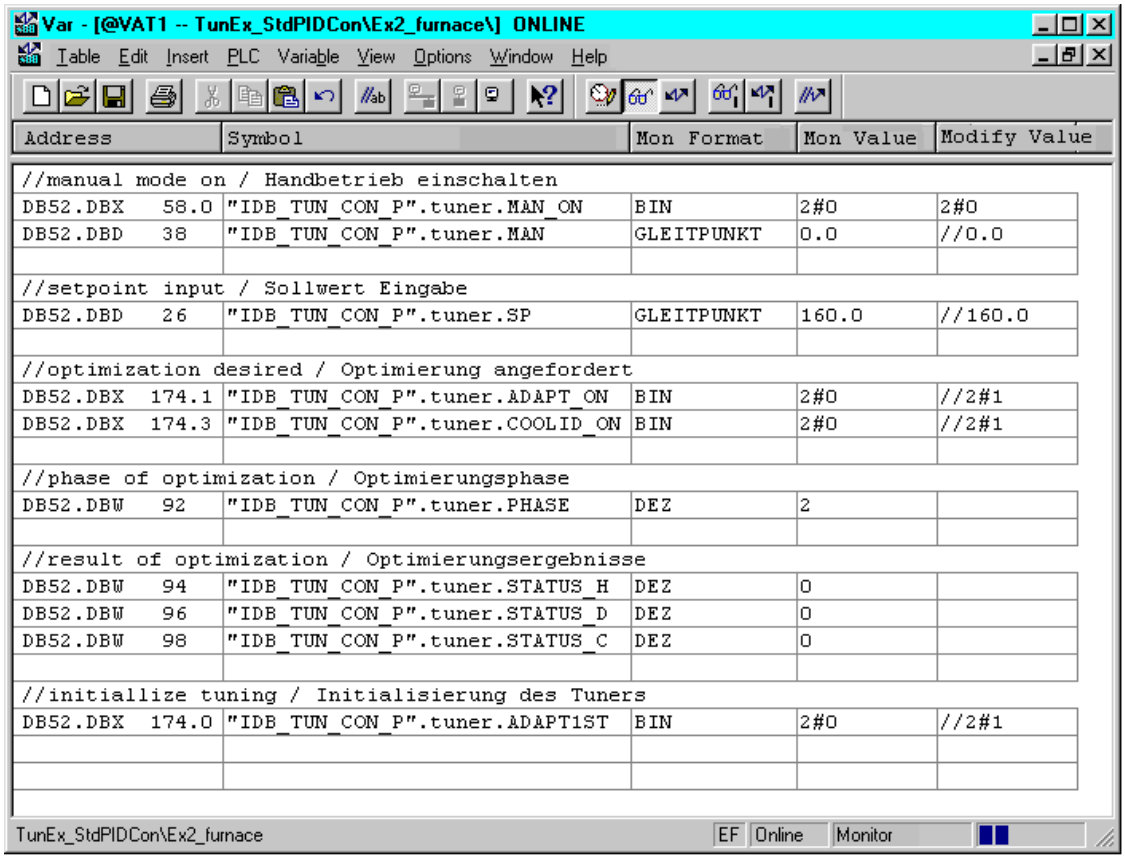
LMNR	Repeated loop manipulated value	Position feedback of the cooling valve
QLMNR_HS	High limit signal of repeated manipulated value	High limit signal of the cooling valve
QLMNR_LS	Low limit signal of repeated manipulated value	Low limit signal of the cooling valve

The binary input signal for heating is converted into a continuous floating-point value (0, 100). The process values are controlled by two first-order time delays after addition of a disturbance variable and multiplication with the process gain. During cooling the binary input signals are controlled via an integrator with the motor actuating time as the integrating time constant.

During initialization the output variable is set to OUTV = DISV_H*GAIN_H - DISV_C*GAIN_C + DISV_OUT.

Operator control and monitoring

You can carry out the operation in the variable declaration table VAT_FURNACE.



Var - [@VAT1 -- TunEx_StdPIDCon\Ex2_furnace\] ONLINE

Table Edit Insert PLC Variable View Options Window Help

Address	Symbol	Mon Format	Mon Value	Modify Value
//manual mode on / Handbetrieb einschalten				
DB52.DBX 58.0	"IDB_TUN_CON_P".tuner.MAN_ON	BIN	2#0	2#0
DB52.DBD 38	"IDB_TUN_CON_P".tuner.MAN	GLEITPUNKT	0.0	//0.0
//setpoint input / Sollwert Eingabe				
DB52.DBD 26	"IDB_TUN_CON_P".tuner.SP	GLEITPUNKT	160.0	//160.0
//optimization desired / Optimierung angefordert				
DB52.DBX 174.1	"IDB_TUN_CON_P".tuner.ADAPT_ON	BIN	2#0	//2#1
DB52.DBX 174.3	"IDB_TUN_CON_P".tuner.COOLID_ON	BIN	2#0	//2#1
//phase of optimization / Optimierungsphase				
DB52.DBW 92	"IDB_TUN_CON_P".tuner.PHASE	DEZ	2	
//result of optimization / Optimierungsergebnisse				
DB52.DBW 94	"IDB_TUN_CON_P".tuner.STATUS_H	DEZ	0	
DB52.DBW 96	"IDB_TUN_CON_P".tuner.STATUS_D	DEZ	0	
DB52.DBW 98	"IDB_TUN_CON_P".tuner.STATUS_C	DEZ	0	
//initiallize tuning / Initialisierung des Tuners				
DB52.DBX 174.0	"IDB_TUN_CON_P".tuner.ADAPT1ST	BIN	2#0	//2#1

TunEx_StdPIDCon\Ex2_furnace EF Online Monitor

Figure: Variable declaration table VAT_FURNACE

The control system can be changed over to manual mode at the switch MAN_ON. The manual value can be specified at MAN. The control system is in manual mode after a restart (warm start).

If you want to optimize the control system, set the bit ADAPT_ON and enter a setpoint at SP. If the control system is still set to manual mode, switch it over to automatic mode.

You can monitor the course of optimization at the parameter PHASE.

The result of optimization is contained in the status words STATUS_H , STATUS_D and STATUS_C.

If you set ADAPT1ST, the Self-Tuner is initialized and the ADAPT_ON bit is set by the block At the beginning of identification the manipulated variable is not calculated automatically, the configured value of LHLM_TUN is used.

Taking the example into operation

- **Software controller**

  Copy the controller blocks into the program Ex2_furnace of your example project.

Control product	Blocks
Standard PID Control V5	FB PID_CP
Modular PID Control	FB DEADBAND, FB PID and FB LMNGEN_C Do not overwrite the FB 43 PULSEGEN with the FB PULSEGEN from Modular PID Control.

  Use HW Config to set the cycle time of the OB35 to 20ms. If a time-out occurs in the watchdog interrupt level, increase the cycle time. In this case the simulation is then executed more slowly. If you control a real process, the cycle time of the OB35 must agree with the sampling times IDB_TUN_CON_P.CYCLE_P and IDB_CONT_S.CYCLE.

- **Controller module FM355/455**

  Use HW Config to adapt the configuration of the hardware structure from the example to your real plant. Use the cycle time of the OB35 and the complete FM parameter configuration.

Use the SIMATIC Manager to download the program Ex2_furnace to the CPU and carry out a restart (warm start).

Under "Simatic STEP 7" use the Start button to start the tool belonging to the controller used and open the block IDB_TUN_CONT_P on-line. Set the ranges of the setpoint and process variable to 0.0 to 200.0. Set the range of the manipulated variable to –100.0 to 300.0. Select the disturbance variable with the range limits -100.0 and 300.0 as the fourth curve. Here you can observe the degree to which the butterfly control valve is opened.

Open the variable declaration table ”VAT_FURNACE” and set ADAP_ON = 1, the setpoint SP = 160.0 as well as MAN_ON = 0.

You can now continue to operate the program as described in the section "Operator control and monitoring".

Figure: Controller optimization with Standard PID Control V5

The above figure shows the controller optimization when heating to the operating point. The first adaption was carried out from the ambient temperature 20 ^{0}C to 160 ^{0}C. After heating identification cooling identification was started. The ventilation valve was opened up to 20%. After controller optimization the setpoint steps were carried out to 180 ^{0}C and returned to 160 ^{0}C .

4.5 Simple example: Ex3_simple_c

Overview

This example contains a simple control loop which consists of a PID controller and a VZ2 element as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller.

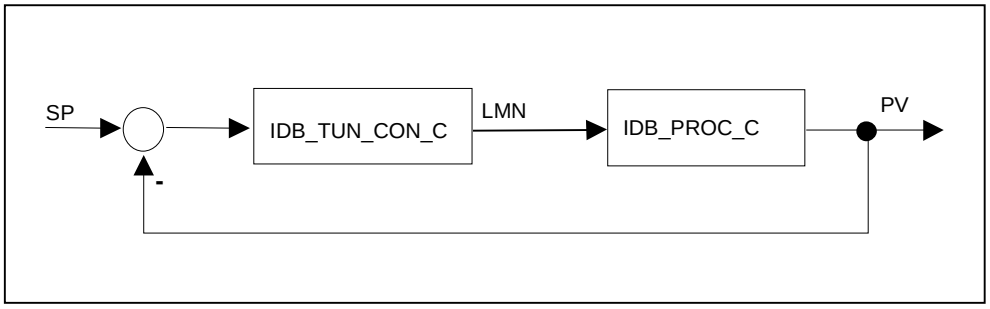


Figure: Block diagram

Program structure

The block IDB_TUN_CON_C is an instance DB of FB TUN_CON_C, which contains the Self-Tuner and the controller. The Self-Tuner, controller and process are called in OB35.

Process block for simulating a temperature process

The block simulates a process with a third-order time delay. For temperature processes select VZ2 behavior with a high and a low time constant (TM_LAG1 = 15*TM_LAG2 and TM_LAG3 = 0s).

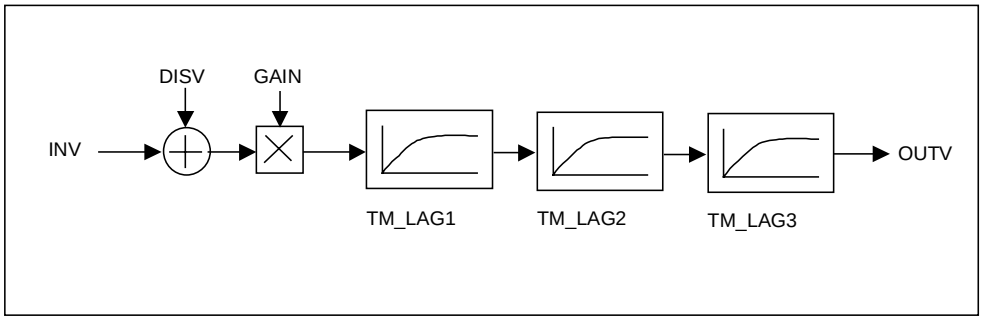


Figure: Block diagram of the process

Parameters

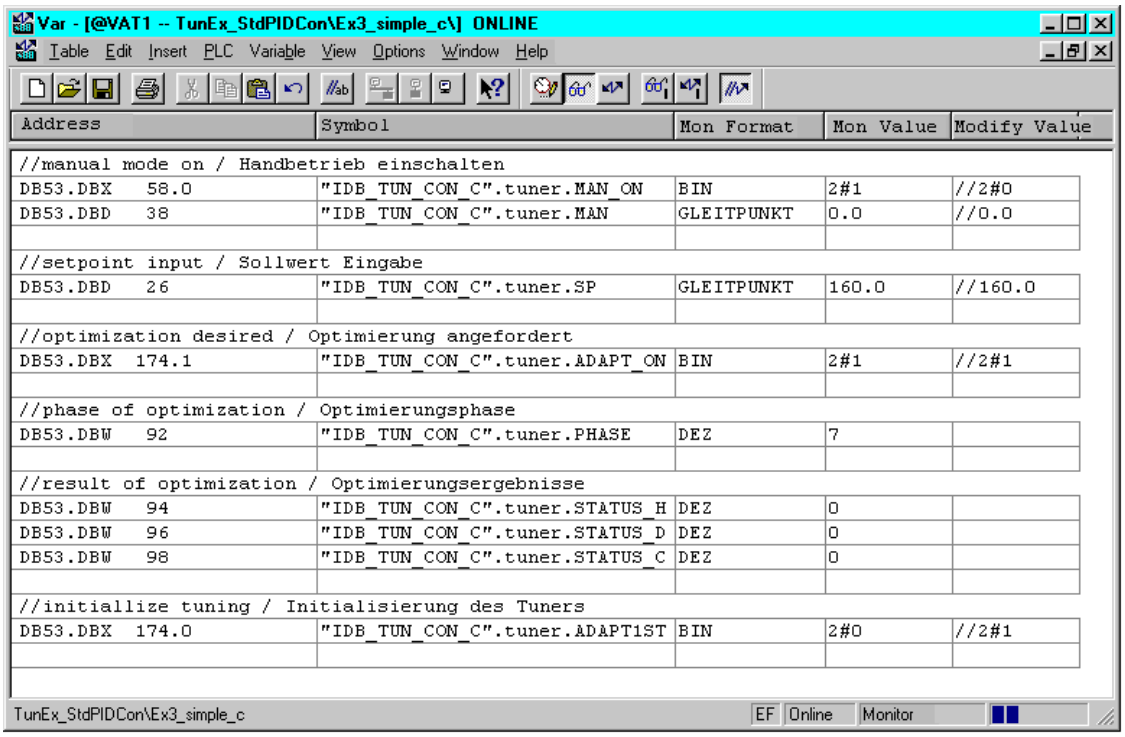
INV	input variable	Input variable (controller setpoint)
DISV	Disturbance variable	Disturbance variable
GAIN	Process gain	Process gain
TM_LAG1	Time lag 1	Time lag 1 (at temperature processes:
TM_LAG2	Time lag 2	Time lag 2 TM_LAG1 = 15*TM_LAG2)
TM_LAG3	Time lag 3	Time lag 3 (=0 at temperature processes)
OUTV	Output variable	Output variable (for example temperature)

The process values are controlled by three first-order time delays after addition of the analog input signal and a disturbance variable and multiplication with the process gain.

During initialization the output variable is set to OUTV = DISV*GAIN.

Operator control and monitoring

You can carry out the operation in the variable declaration table VAT_SIMPLE.



Address	Symbol	Mon Format	Mon Value	Modify Value
//manual mode on / Handbetrieb einschalten				
DB53.DBX 58.0	"IDB_TUN_CON_C".tuner.MAN_ON	BIN	2#1	//2#0
DB53.DBD 38	"IDB_TUN_CON_C".tuner.MAN	GLEITPUNKT	0.0	//0.0
//setpoint input / Sollwert Eingabe				
DB53.DBD 26	"IDB_TUN_CON_C".tuner.SP	GLEITPUNKT	160.0	//160.0
//optimization desired / Optimierung angefordert				
DB53.DBX 174.1	"IDB_TUN_CON_C".tuner.ADAPT_ON	BIN	2#1	//2#1
//phase of optimization / Optimierungsphase				
DB53.DBW 92	"IDB_TUN_CON_C".tuner.PHASE	DEZ	7	
//result of optimization / Optimierungsergebnisse				
DB53.DBW 94	"IDB_TUN_CON_C".tuner.STATUS_H	DEZ	0	
DB53.DBW 96	"IDB_TUN_CON_C".tuner.STATUS_D	DEZ	0	
DB53.DBW 98	"IDB_TUN_CON_C".tuner.STATUS_C	DEZ	0	
//initiallize tuning / Initialisierung des Tuners				
DB53.DBX 174.0	"IDB_TUN_CON_C".tuner.ADAPT1ST	BIN	2#0	//2#1

Figure: Variable declaration table VAT_SIMPLE

The control system can be changed over to manual mode at the switch MAN_ON. The manual value can be specified at MAN. The control system is in manual mode after a restart (warm start).

If you want to optimize the control system, set the bit ADAPT_ON and enter a setpoint at SP. If the control system is still set to manual mode, switch it over to automatic mode.

You can monitor the course of optimization at the parameter PHASE.

The result of optimization is contained in the status words STATUS_H , STATUS_D and STATUS_C.

If you set ADAPT1ST, the Self-Tuner is initialized and the ADAPT_ON bit is set by the block At the beginning of identification the manipulated variable is not calculated automatically, the configured value of LHLM_TUN is used.

Taking the example into operation

- **Software controller**

  Copy the controller blocks into the program Ex3_simple_c of your example project.

Control product	Blocks
Standard PID Control V5	FB PID_CP
Modular PID Control	FB PID and FB LMNGEN_C

  Use HW Config to set the cycle time of the OB35 to 100ms (default setting). If a time-out occurs in the watchdog interrupt level, increase the cycle time. In this case the simulation is then executed more slowly. If you control a real process, the cycle time of the OB35 must agree with the sampling times IDB_TUN_CON_C.CYCLE_P.

- **Controller module FM355/455**

  Use HW Config to adapt the configuration of the hardware structure from the example to your real plant. Use the cycle time of the OB35 and the complete FM parameter configuration.

Use the SIMATIC Manager to download the program Ex3_simple_c to the CPU.

Under "Simatic STEP 7" use the Start button to start the tool belonging to the controller used and open the block ”IDB_TUN_CONT_C" on-line. At the curve recorder set Acquisition cycle= 1s and Length of the time axis= 500s under "Settings ...". Set the ranges of the setpoint and process variable to 0.0 to 200.0. Set the range of the manipulated variable to –100.0 to 300.0.

Open the variable declaration table ”VAT_ SIMPLE” and there set ADAPT1ST = TRUE, the setpoint SP to 160.0 as well as MAN_ON to FALSE. Optimization is carried out.

You can now continue to operate the program as described in the section "Operator control and monitoring".

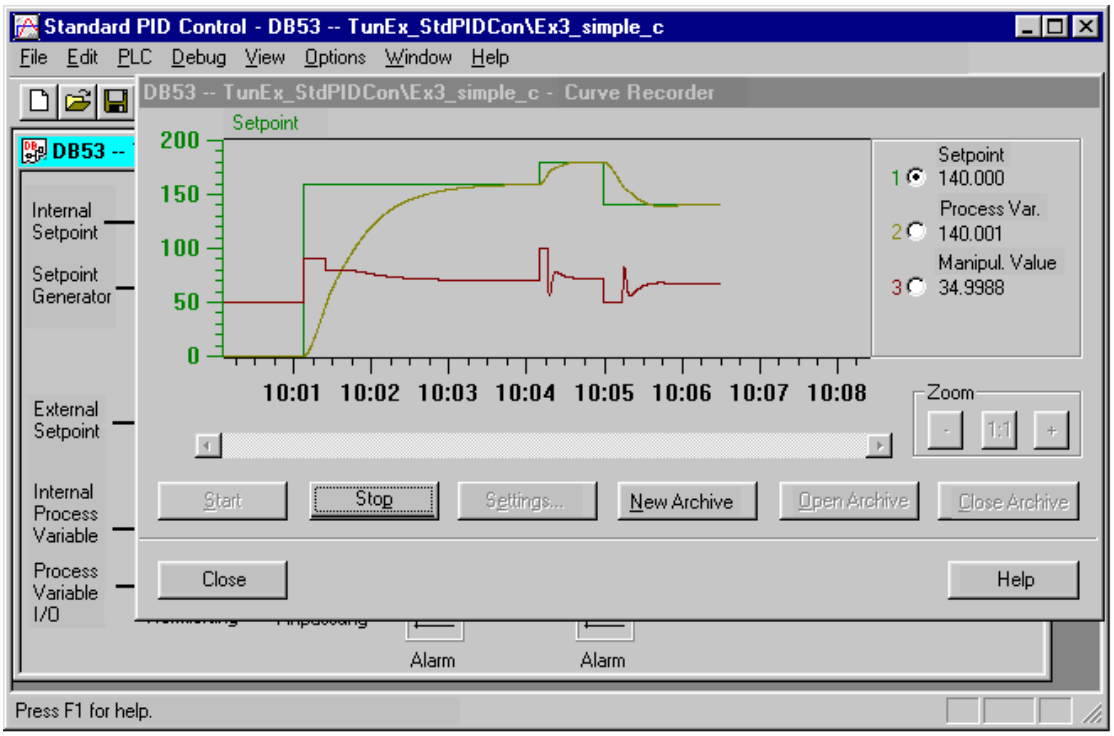


Figure: Controller optimization with Standard PID Control V5

The above figure shows the controller optimization when heating to the operating point. The first adaption was carried out from the ambient temperature 20 ^{0}C to 160 ^{0}C. After the heating identification the setpoint steps were carried out to 180 ^{0}C and returned to 160 ^{0}C .

4.6 Simple example: Ex4_simple_s

Overview

The example contains a simple control loop which consists of a PID controller and a VZ2 element with an integrating actuating element as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a PID step controller.

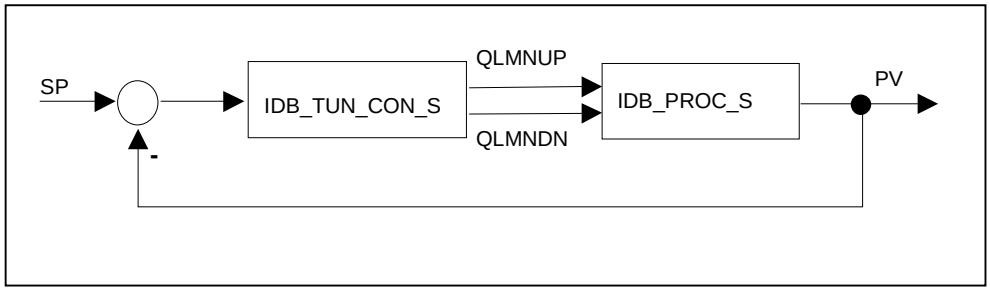


Figure: Block diagram

Program structure

The block IDB_TUN_CON_S is an instance DB of FB TUN_CON_S, which contains the Self-Tuner and the controller. The Self-Tuner, controller and process are called in OB35.

Process block for simulating a temperature process

The block simulates a process with a third-order time delay. For temperature processes select VZ2 behavior with a high and a low time constant (TM_LAG1 = 15*TM_LAG2 and TM_LAG3 = 0s).

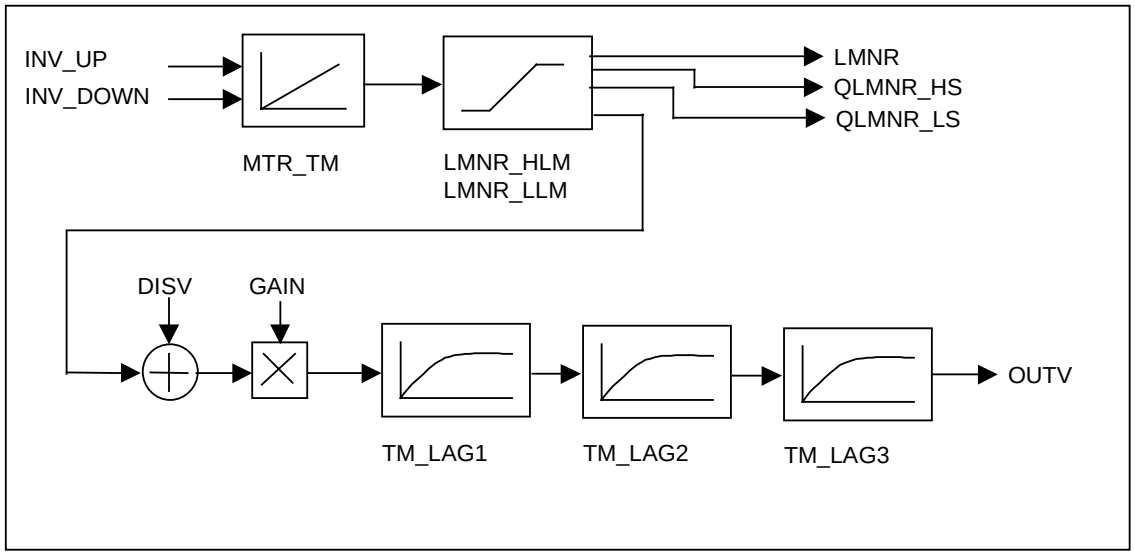


Figure: Block diagram of the process

Parameters

INV_UP	Input variable up	Actuating signal up
INV_DOWN	Input variable down	Actuating signal down
DISV	Disturbance variable	Disturbance variable
GAIN	Process gain	Process gain
MTR_TM	Motor actuating time	Actuating time of the final control element
LMNR_HLM	Repeated manipulated value high limit	High limit of final control element
LMNR_LLM	Repeated manipulated value low limit	Low limit of final control element
TM_LAG1	Time lag 1	Time lag 1
TM_LAG2	Time lag 2	Time lag 2 (at temperature processes: TM_LAG1 = 15*TM_LAG2)
TM_LAG3	Time lag 3	Time lag 3 (=0 at temperature processes)
OUTV	Output variable	Output variable (for example temperature)
LMNR	Repeated manipulated value	Position feedback value
QLMNR_HS	High limit signal of repeated manipulated value	High limit of position feedback value
QLMNR_LS	Low limit signal of repeated manipulated value	Low limit of position feedback value

Depending on the input signals INV_UP and INV_DOWN the position feedback LMNR is calculated by means of an integrator. The position feedback is limited to LMNR_HLM and LMNR_LLM. When the limit is reached the position feedback signals are set to QLMNR_HS or QLMNR_LS.

The process values are controlled by three first-order time delays after addition of a disturbance variable and subsequent multiplication with the process gain.

During initialization the output variables OUTV and LMNR are set to zero.

Operator control and monitoring

You can carry out the operation in the variable declaration table VAT_SIMPLE.

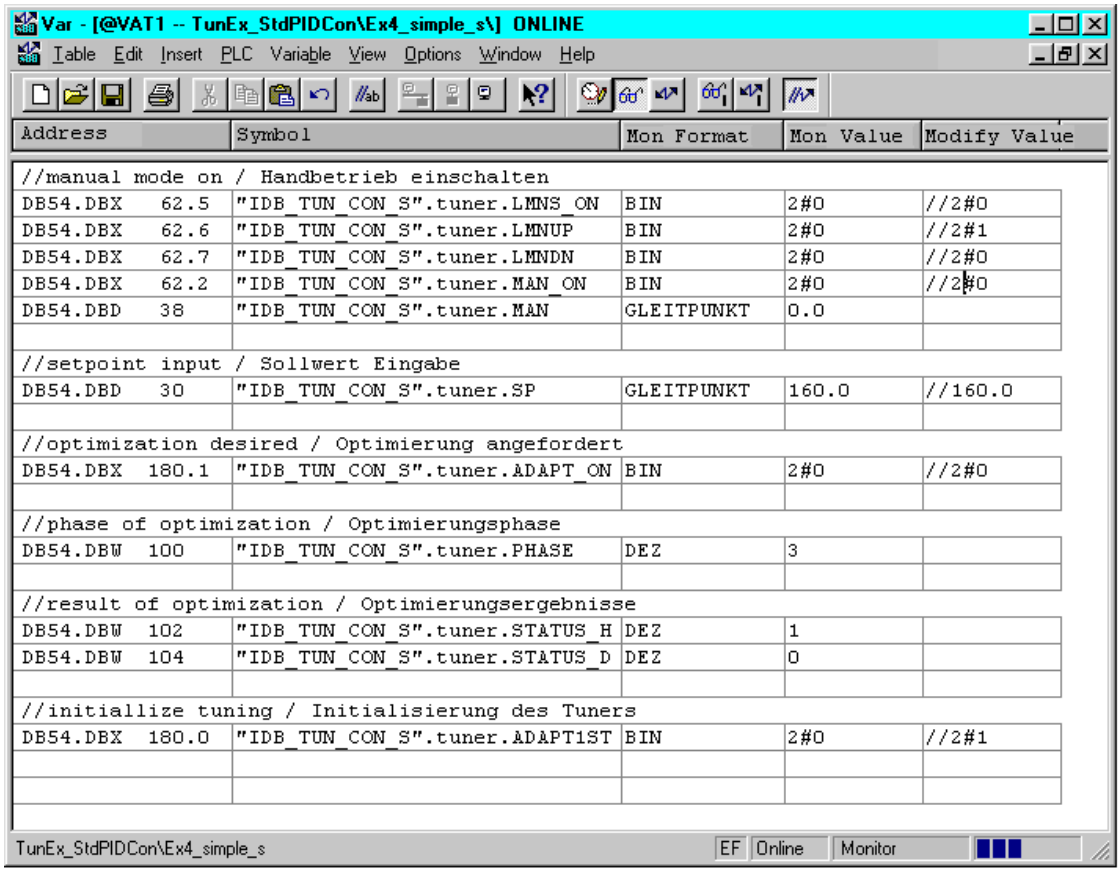


Figure: Variable declaration table VAT_SIMPLE

The control system can be changed over to manual mode at the switch MAN_ON. The manual value can be specified at MAN. The control system is in manual mode after a restart (warm start). If LMNS_ON is set, the outputs QLMNUP and QLMNDN can be controlled by hand at the inputs LMNUP or LMNDN.

If you want to optimize the control system, set the bit ADAPT_ON and enter a setpoint at SP. If the control system is still set to manual mode, switch it over to automatic mode. MAN_ON **and** LMNS_ON must be set to FALSE.

You can monitor the course of optimization at the parameter PHASE.

The result of optimization is contained in the status words STATUS_H , STATUS_D and STATUS_C.

If you set ADAPT1ST, the Self-Tuner is initialized and the ADAPT_ON bit is set by the block At the beginning of identification the manipulated variable is not calculated automatically, the configured value of LHLM_TUN is used.

Taking the example into operation

- **Software controller**

  Copy the controller blocks into the program Ex4_simple_s of your example project.

Control product	Blocks
Standard PID Control V5	FB PID_ES
Modular PID Control	FB DEADBAND, FB PID and FB LMNGEN_S

  Use HW Config to set the cycle time of the OB35 to 20ms. If a time-out occurs in the watchdog interrupt level, increase the cycle time. In this case the simulation is then executed more slowly. If you control a real process, the cycle time of the OB35 must agree with the sampling time IDB_TUN_CON_S.CYCLE_P.

- **Controller module FM355/455**

  Use HW Config to adapt the configuration of the hardware structure from the example to your real plant. Use the cycle time of the OB35 and the complete FM parameter configuration.

Use the SIMATIC Manager to download the program Ex4_simple_s to the CPU.

Under "Simatic STEP 7" use the Start button to start the tool belonging to the controller used and open the block ”IDB_TUN_CONT_S" on-line. At the curve recorder set Acquisition cycle= 1s and Length of the time axis= 500s under "Settings ...". Set the ranges of the setpoint and process variable to 0.0 to 200.0. Set the range of the manipulated variable to –100.0 to 300.0.

Open the variable declaration table ”VAT_ SIMPLE” and there set ADAPT1ST = TRUE, the setpoint SP to 160.0 as well as MAN_ON and LMNS_ON to FALSE. Optimization is carried out.

You can now continue to operate the program as described in the section "Operator control and monitoring".

Figure: Controller optimization with Standard PID Control V5

The above figure shows the controller optimization when heating to the operating point. The first adaption was carried out from the ambient temperature 20 ^{0}C to 160 ^{0}C. After the heating identification the setpoint steps were carried out to 180 ^{0}C and returned to 160 ^{0}C .

4.7 Simple example: Ex5_simple_p

Overview

This example contains a simple control loop which consists of a PID controller with pulse generation and a VZ2 element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generation.

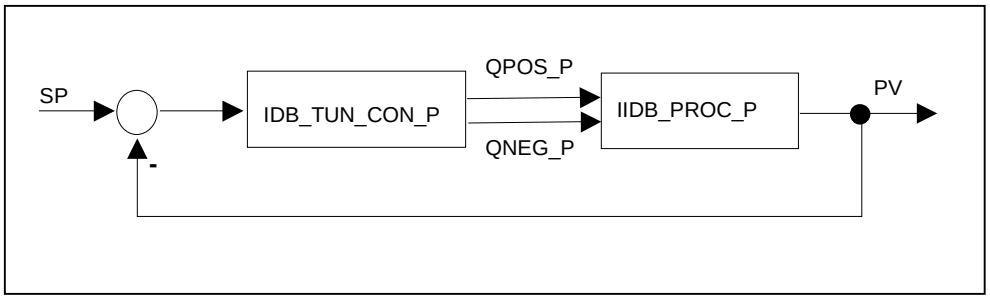


Figure: Block diagram

Program structure

The block IDB_TUN_CON_P is an instance DB of FB TUN_CON_P, which contains the Self-Tuner and the controller with pulse generation. The Self-Tuner, controller and process are called in OB35.

Process block for simulating a temperature heating zone

The block simulates a typical temperature process for heating and cooling as it can occur as a control zone in a extruder, an injection molding machine, a tempering machine or as a separate furnace in practice.

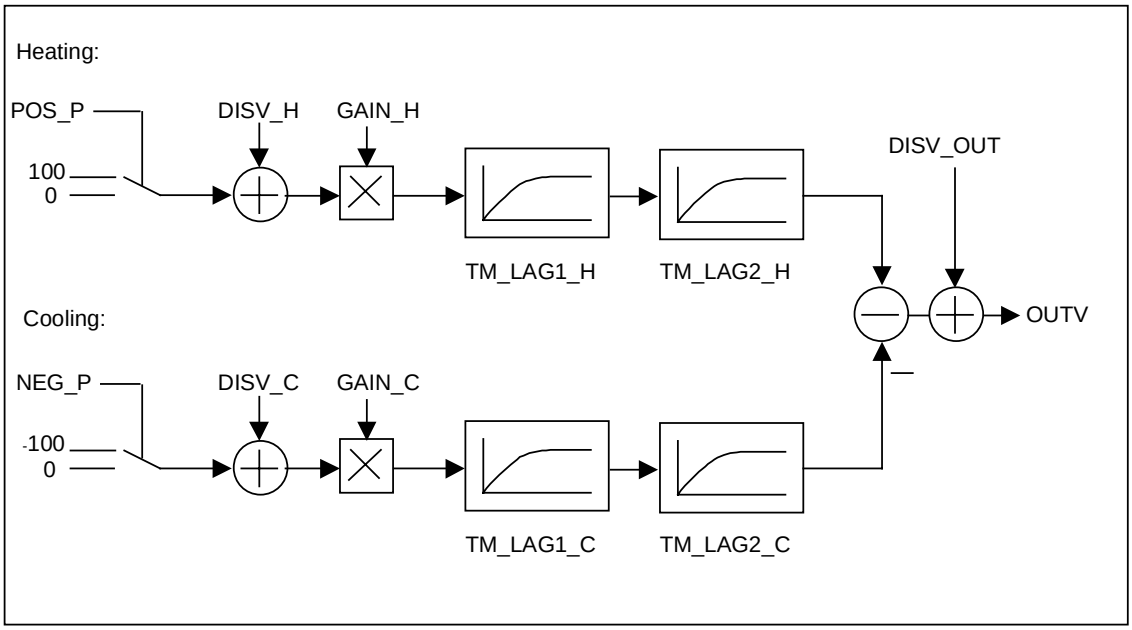


Figure: Block diagram of the process

Parameters

POS_P	Positive pulse	Binary input signal heating
NEG_P	Negative pulse	Binary input signal cooling
DISV_H	Disturbance variable heating	Disturbance variable heating
DISV_C	Disturbance variable cooling	Disturbance variable cooling
DISV_OUT	Disturbance variable by output	Disturbance variable at output (for example ambient temperature)
GAIN_H	Process gain heating	Process gain heating
GAIN_C	Process gain cooling	Process gain cooling
TM_LAG1_H	Time lag 1 heating	Time lag 1 heating
TM_LAG2_H	Time lag 2 heating	Time lag 2 heating
TM_LAG1_C	Time lag 1 cooling	Time lag 1 cooling
TM_LAG2_C	Time lag 2 cooling	Time lag 2 cooling
OUTV	Output variable	Output variable (for example control zone temperature)

The binary input signals are converted into continuous floating-point values (-100, 0, 100). The process values are controlled by two first-order time delays after addition of a disturbance variable (for example ambient temperature) and multiplication with the process gain. This process is carried out separately for heating and cooling.

During initialization the output variable is set to OUTV = DISV_H*GAIN_H - DISV_C*GAIN_C + DISV_OUT.

Operator control and monitoring

You can carry out the operation in the variable declaration table VAT_SIMPLE.

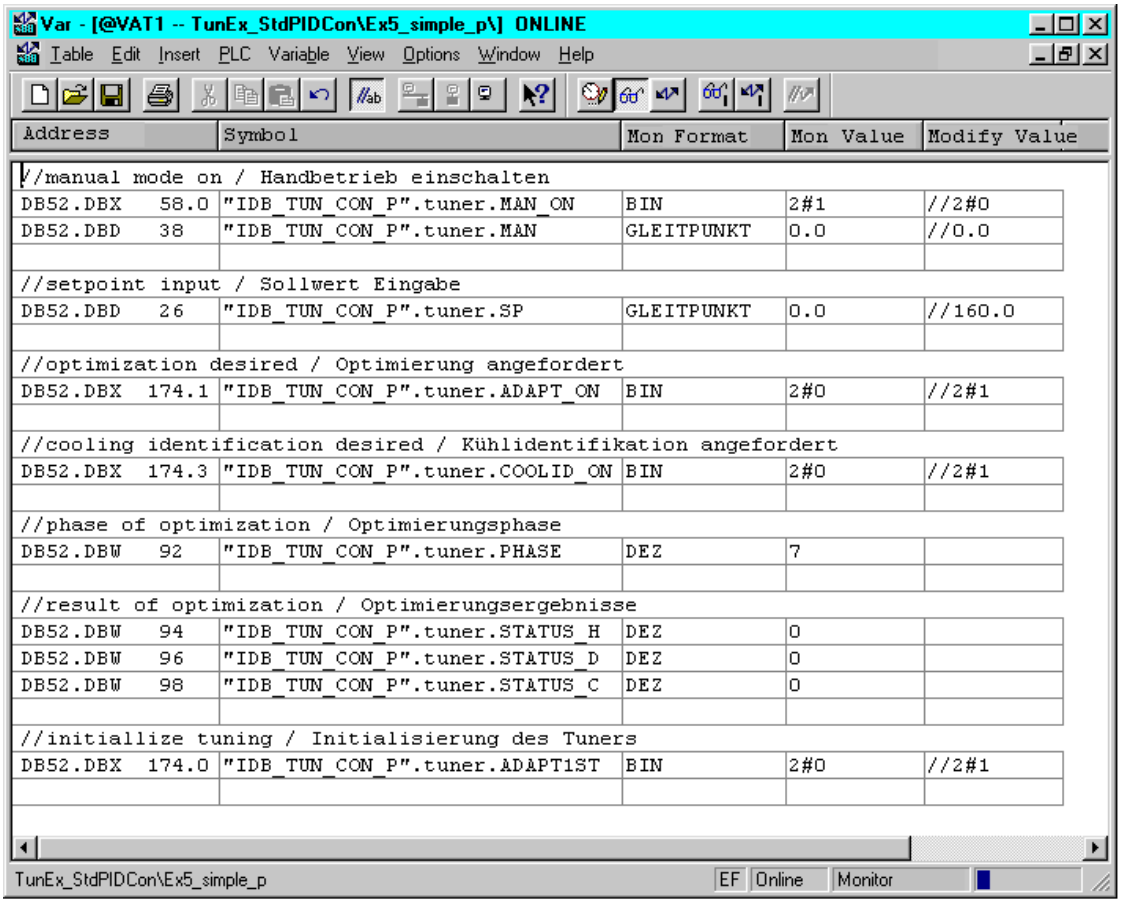


Figure: Variable declaration table VAT_SIMPLE

The control system can be changed over to manual mode at the switch MAN_ON. The manual value can be specified at MAN. The control system is in manual mode after a restart (warm start).

If you want to optimize the control system, set the bit ADAPT_ON and enter a setpoint at SP. If the control system is still set to manual mode, switch it over to automatic mode.

You can monitor the course of optimization at the parameter PHASE.

The result of optimization is contained in the status words STATUS_H , STATUS_D and STATUS_C.

If you set ADAPT1ST, the Self-Tuner is initialized and the ADAPT_ON bit is set by the block At the beginning of identification the manipulated variable is not calculated automatically, the configured value of LHLM_TUN is used.

Taking the example into operation

- **Software controller**

  Copy the controller blocks into the program Ex5_simple_p of your example project.

Control product	Blocks
Standard PID Control V5	FB PID_CP
Modular PID Control	FB DEADBAND, FB PID and FB LMNGEN_C Do not overwrite the FB 43 PULSEGEN with the FB PULSEGEN from Modular PID Control.

  Use HW Config to set the cycle time of the OB35 to 20ms. If a time-out occurs in the watchdog interrupt level, increase the cycle time. In this case the simulation is then executed more slowly. If you control a real process, the cycle time of the OB35 must agree with the sampling time IDB_TUN_CON_S.CYCLE_P.

- **Controller module FM355/455**

  Use HW Config to adapt the configuration of the hardware structure from the example to your real plant. Use the cycle time of the OB35 and the complete FM parameter configuration.

Use the SIMATIC Manager to download the program Ex5_simple_p to the CPU.

Under "Simatic STEP 7" use the Start button to start the tool belonging to the controller used and open the block IDB_TUN_CONT_P on-line. At the curve recorder set Acquisition cycle= 1s and Length of the time axis= 500s under "Settings ...". Set the ranges of the setpoint and process variable to 0.0 to 200.0. Set the range of the manipulated variable to –100.0 to 300.0.

Open the variable declaration table ”VAT_ SIMPLE” and there set ADAPT1ST = TRUE, the setpoint SP to 160.0 as well as MAN_ON to FALSE. Optimization is carried out.

You can now continue to operate the program as described in the section "Operator control and monitoring".

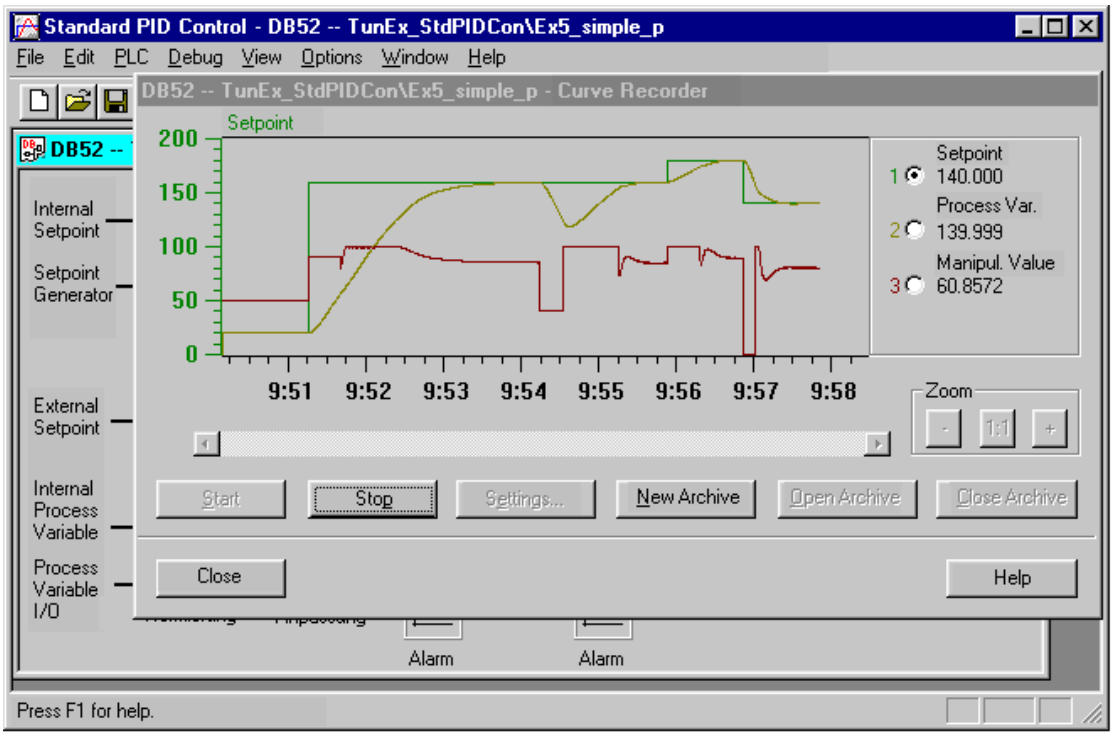


Figure: Controller optimization with Standard PID Control V5

The above figure shows the controller optimization when heating to the operating point. The first adaption was carried out from the ambient temperature 20 ^{0}C to 160 ^{0}C. After the heating identification the setpoint steps were carried out to 180 ^{0}C and returned to 160 ^{0}C .

5 Technical Specifications

5.1 Technical Specifications

Run times

The specified values in the following table are run times in ms.

	TUN_EC, FB 50	TUN_ES, FB 51
CPU 313	1.5	1.5
CPU 314	1.5	1.5
CPU 315 CPU 315-2DP	1.3 1.3	1.3 1.3
CPU 412-1 CPU 413-1 CPU 413-2DP	0.18 0.18 0.18	0.18 0.18 0.18
CPU 414-1 CPU 414-2DP	0.12 0.12	0.12 0.12
CPU 416-1 CPU 416-2DP	0.06 0.06	0.06 0.06

Memory assignment

	Size in the load memory (in bytes)	Size in the work memory (in bytes)	Local data used (in bytes)
TUN_EC	6542	5956	84
TUN_ES	6332	5714	78
Instance DB for TUN_EC	644	294	–
Instance DB for TUN_ES	638	288	–

Index