BECKHOFF New Automation Technology

Manual | EN

TwinCAT 3

TC3 Temperature Controller

2021-03-24 | Version: 2

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

NOTE

Damage to the environment or devices

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

ℹ **Tip or pointer**

This symbol indicates information that contributes to better understanding.

2 Product description

The TwinCAT Temperature Controller is a universally applicable PLC function block for monitoring and controlling a wide variety of temperature-dependent processes. The controller can be operated in

- automatic mode (closed loop) and
- manual mode (open loop).

The control value can be accessed in digital or analog form. The digital control value is pulse width modulated (PWM). A two-point or three-point output is also available. The control value is limited to the permitted maximum and minimum values.

The setpoint is also limited to permitted minimum and maximum values, and can also be ramped. A bit is available in the interface to the function block that provides easy switching from the setpoint to a standby setpoint. A soft start can be parameterized to support "heater baking". This involves the setpoint (optionally ramped) being initially set to a low value, remaining there for a certain time, then being changed to the true setpoint (again optionally ramped).

The actual value can be digitally filtered.

The control algorithm is PID-based. An additional pre-controller can be inserted in order to minimize overshoot.

The controller has a variety of parameterizable monitoring functions. There is

- tolerance band monitoring (two different tolerance bands),
- absolute value monitoring,
- encoder monitoring (open, backvoltage, reverse) and
- heating current monitoring (open, short circuit, leakage current).

There is an algorithm for determination of optimal controller parameters that greatly simplifies the process of commissioning the controller. This algorithm evaluates a jump and uses the inflectional tangent method to determine the maximum velocity and delay time of the section. This data allows a controller to be specified according to the rules of Chien, Hrones and Reswick. The parameters for the pre-controller are also determined here. If the controller parameters are already known, then the controller can also be operated using these externally supplied parameters. The controller parameters can be determined separately for the heating and cooling section. A corresponding sequence of the tuning process can be preselected. If no separate parameter set is determined for cooling, it is also possible to use the heating parameter set for the cooling section by means of a freely selectable scaling factor.

Commissioning the Controller in Stages [▸ 16].

Documentation of the function block [▸ 20] and the required structures.

3 Installation

3.1 System requirements

This section describes the minimum requirements needed for engineering and/or runtime systems.

Development environment

A pure development environment describes a computer on which PLC programs are developed but not executed. The following components must be installed on a development computer:

- TwinCAT 3 XAE (Engineering) build 4012 or higher
- TwinCAT 3 function TF4110 temperature controller version 3.3.0.0 or higher
- Please note: A 7-day trial license can be used (multiple times, if required) for the development environment, see Licensing [▶ 11].

Runtime environment

A runtime environment describes a computer on which PLC programs are executed. The following components must be installed on a runtime computer:

- TwinCAT3 XAR build 4012 or higher
- Installation of TC1200 PLC and TF4110 temperature controller
- Licenses for TC1200 PLC and TF4110 temperature controller
  Note: A 7-day trial license key can be used for testing purposes, see Licensing [▶ 11].

Developer environment and runtime on one computer

If runtime and development environments are to run on the same computer (e.g. to test a PLC program before it is loaded on the target computer), the following requirements must be met:

- TwinCAT3 XAE (engineering installation) build 4012 or higher
- Licenses for TC1200 PLC and TF4110 temperature controller
- Please note: A 7-day trial license key can be used for testing purposes, see Licensing [▶ 11].

3.2 Installation

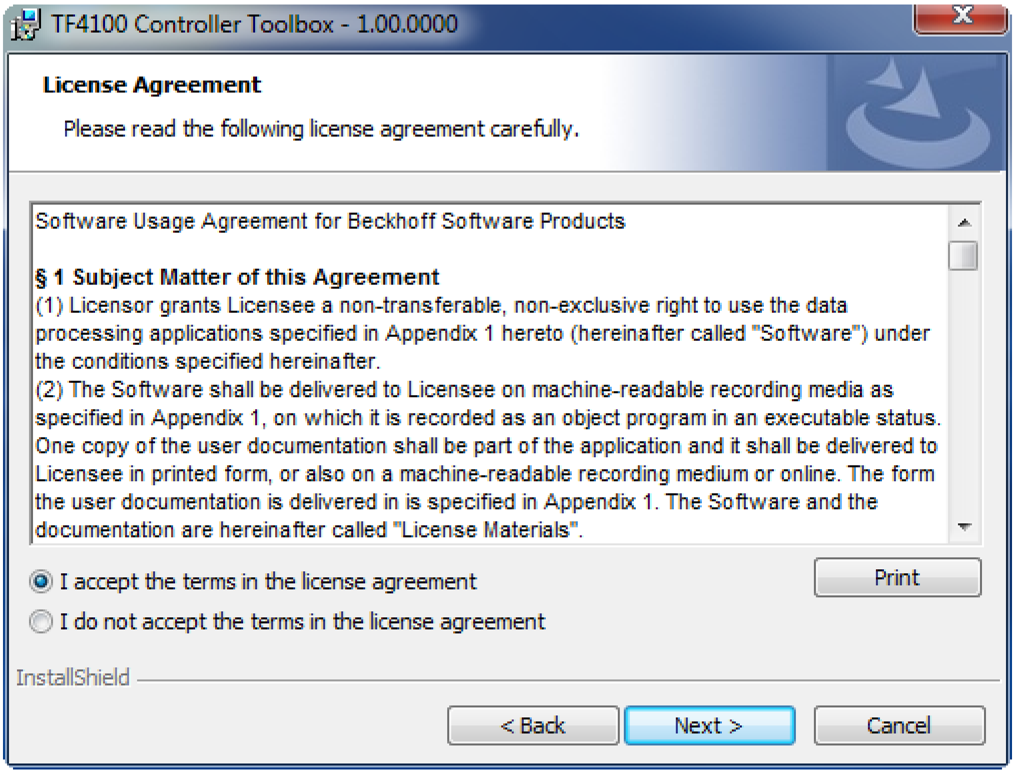
The following section describes how to install the TwinCAT 3 Function for Windows-based operating systems.

✓ The TwinCAT 3 Function setup file was downloaded from the Beckhoff website.

1. Run the setup file as administrator. To do this, select the command **Run as administrator** in the context menu of the file.

   ⇨ The installation dialog opens.

2. Accept the end user licensing agreement and click **Next**.



TF4100 Controller Toolbox - 1.00.0000

License Agreement

Please read the following license agreement carefully.

Software Usage Agreement for Beckhoff Software Products

§ 1 Subject Matter of this Agreement

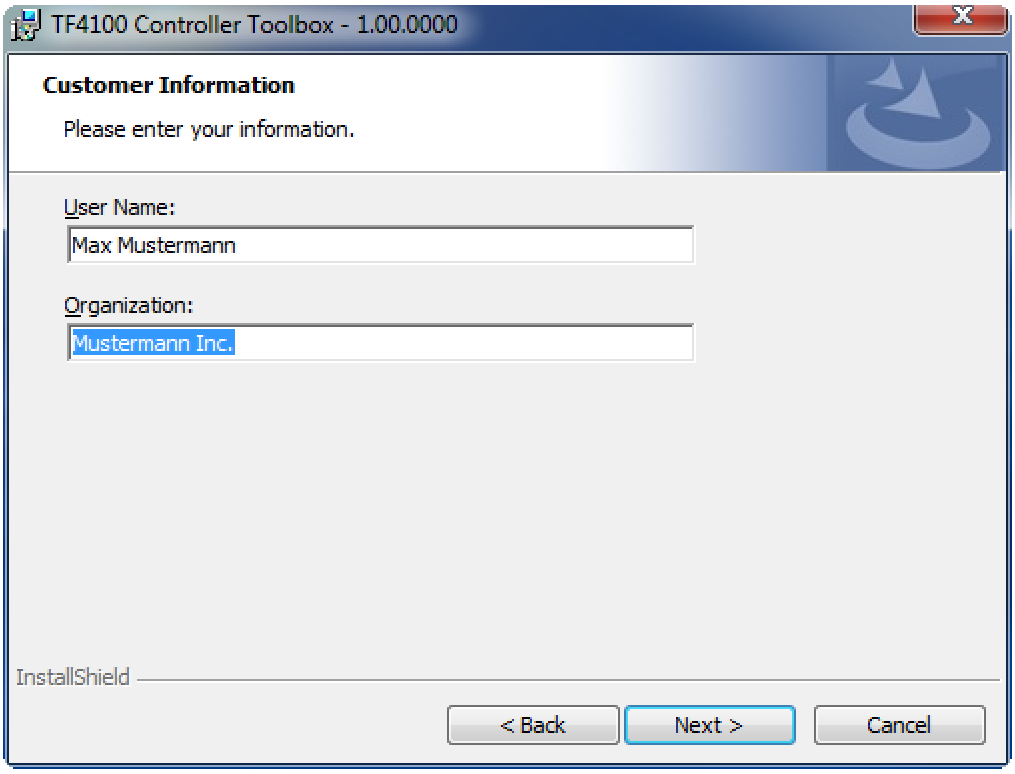
(1) Licensor grants Licensee a non-transferable, non-exclusive right to use the data processing applications specified in Appendix 1 hereto (hereinafter called "Software") under the conditions specified hereinafter.

(2) The Software shall be delivered to Licensee on machine-readable recording media as specified in Appendix 1, on which it is recorded as an object program in an executable status. One copy of the user documentation shall be part of the application and it shall be delivered to Licensee in printed form, or also on a machine-readable recording medium or online. The form the user documentation is delivered in is specified in Appendix 1. The Software and the documentation are hereinafter called "License Materials".

● I accept the terms in the license agreement

○ I do not accept the terms in the license agreement

3. Enter your user data.



TF4100 Controller Toolbox - 1.00.0000

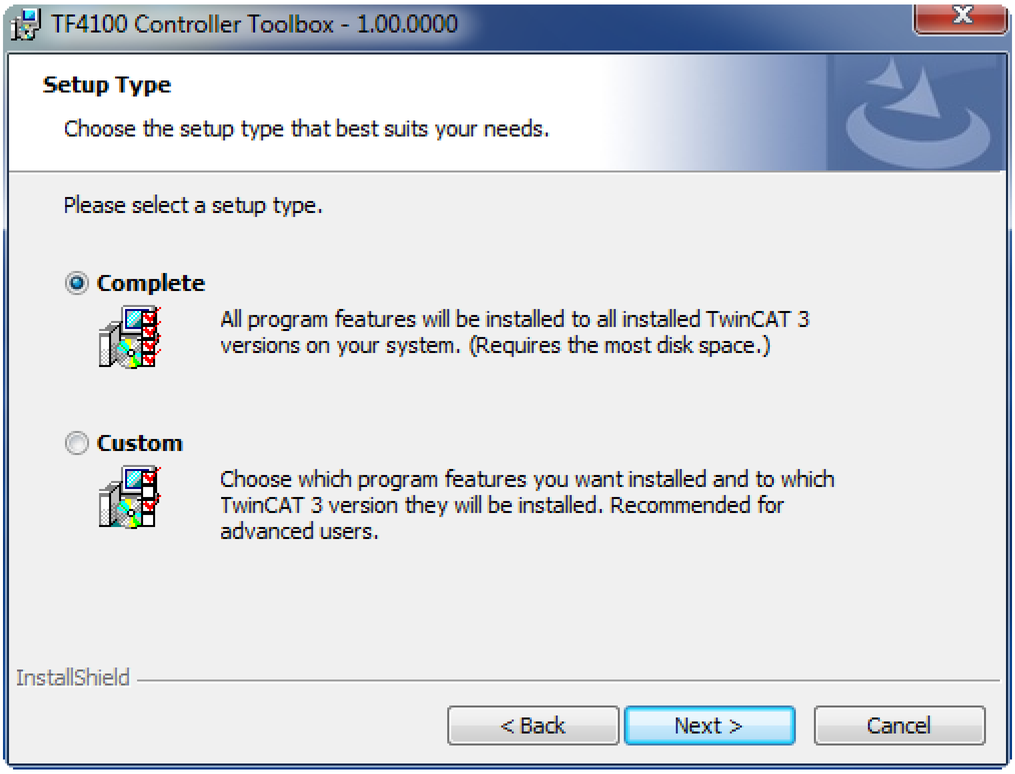
Customer Information

Please enter your information.

User Name: Max Mustermann

Organization: Mustermann Inc.

4. If you want to install the full version of the TwinCAT 3 Function, select **Complete** as installation type. If you want to install the TwinCAT 3 Function components separately, select **Custom**.



TF4100 Controller Toolbox - 1.00.0000

Setup Type

Choose the setup type that best suits your needs.

Please select a setup type.

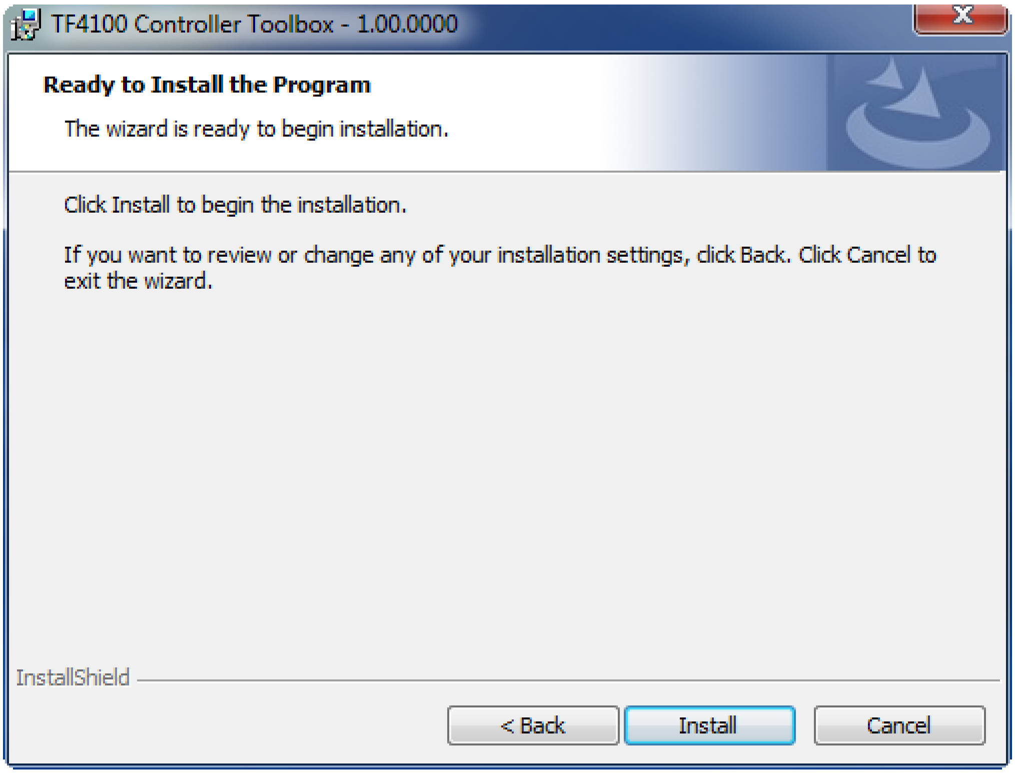
● **Complete**

All program features will be installed to all installed TwinCAT 3 versions on your system. (Requires the most disk space.)

○ **Custom**

Choose which program features you want installed and to which TwinCAT 3 version they will be installed. Recommended for advanced users.

5. Select **Next**, then **Install** to start the installation.



TF4100 Controller Toolbox - 1.00.0000

Ready to Install the Program

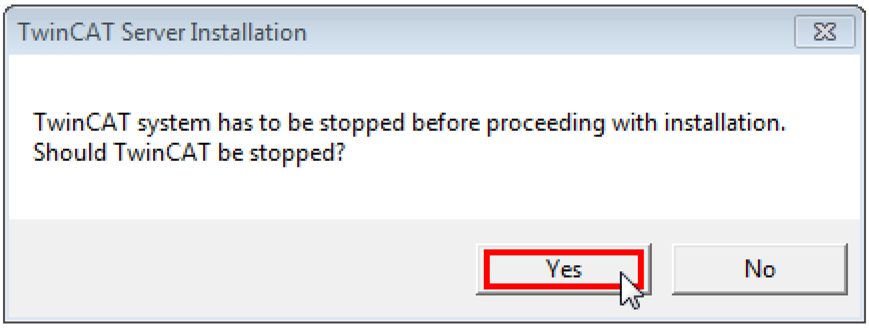
The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

⇨ A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

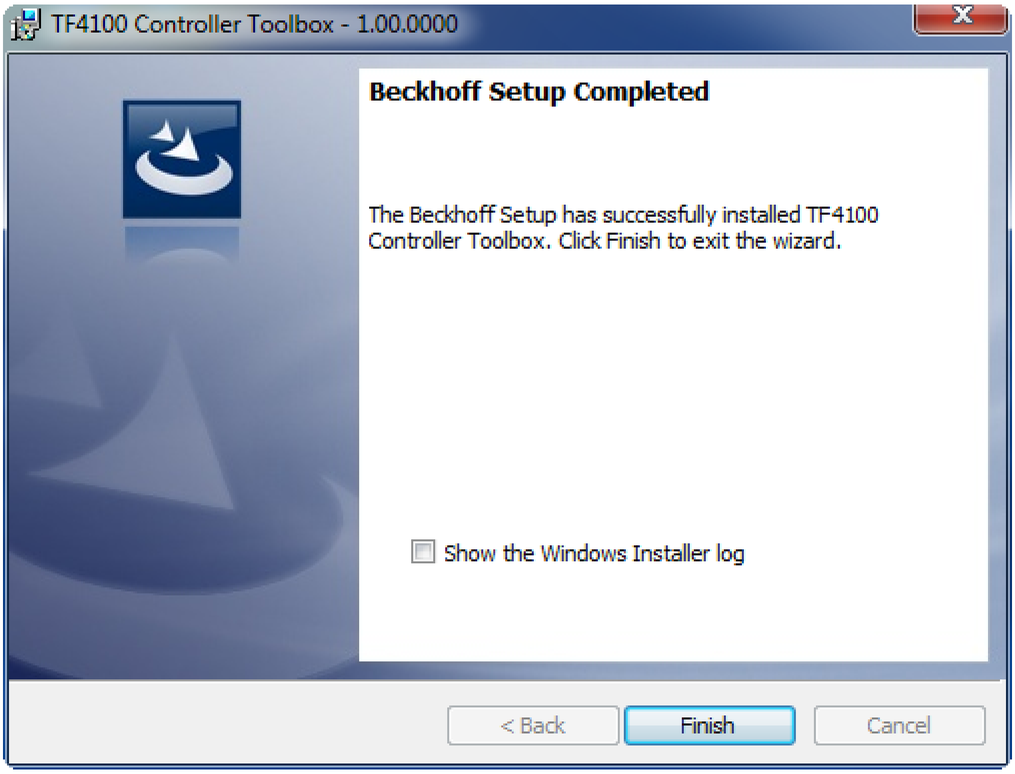
6. Confirm the dialog with **Yes**.



TwinCAT Server Installation

TwinCAT system has to be stopped before proceeding with installation.
Should TwinCAT be stopped?

7. Select **Finish** to exit the setup.



TF4100 Controller Toolbox - 1.00.0000

Beckhoff Setup Completed

The Beckhoff Setup has successfully installed TF4100 Controller Toolbox. Click Finish to exit the wizard.

□ Show the Windows Installer log

⇨ The TwinCAT 3 Function has been successfully installed and can be licensed (see Licensing [▶ 11]).

3.3 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

Licensing the full version of a TwinCAT 3 Function

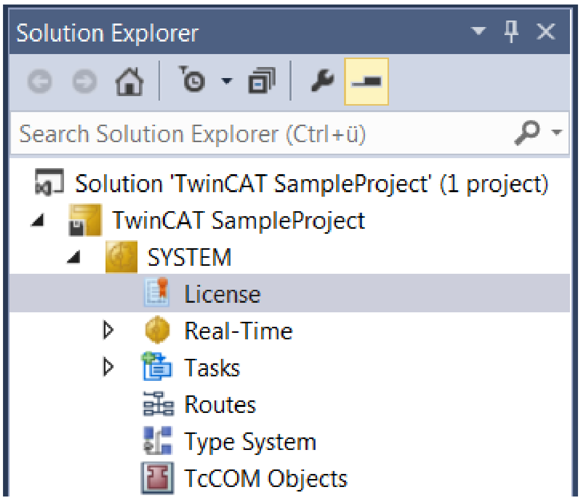
A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "TwinCAT 3 Licensing".

Licensing the 7-day test version of a TwinCAT 3 Function

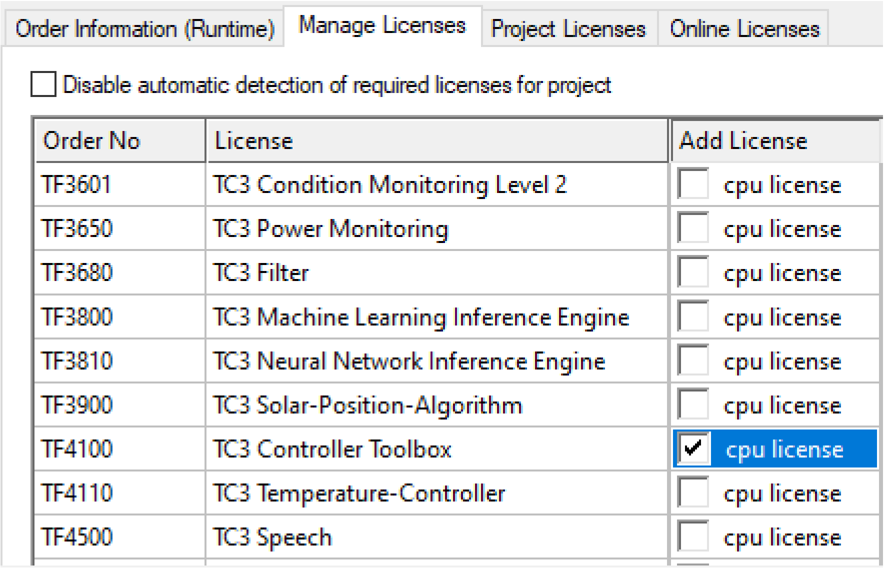
A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.

3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
   ⇨ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.
4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



   ⇨ The TwinCAT 3 license manager opens.
5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF6420: TC3 Database Server").

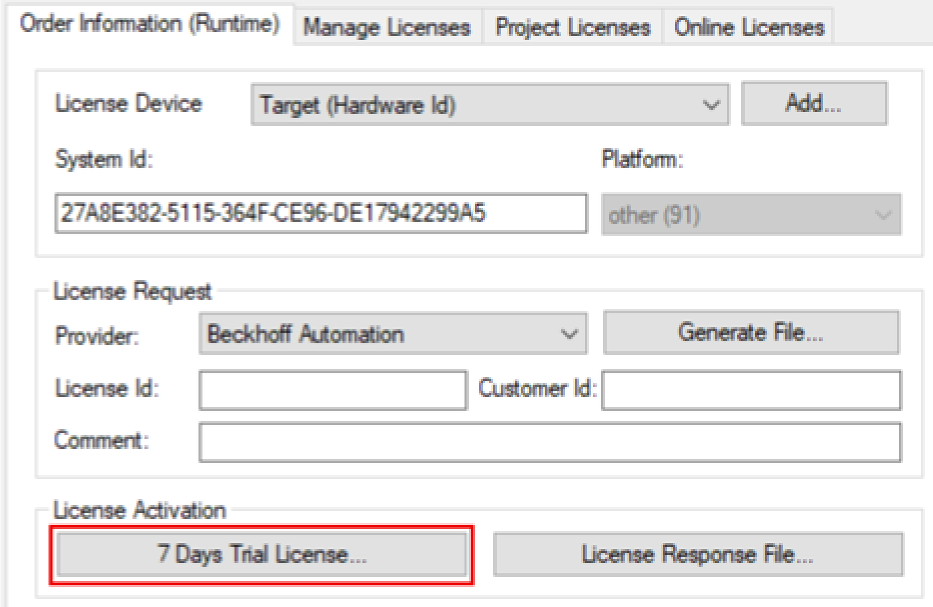


□ Disable automatic detection of required licenses for project

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	□ cpu license
TF3650	TC3 Power Monitoring	□ cpu license
TF3680	TC3 Filter	□ cpu license
TF3800	TC3 Machine Learning Inference Engine	□ cpu license
TF3810	TC3 Neural Network Inference Engine	□ cpu license
TF3900	TC3 Solar-Position-Algorithm	□ cpu license
TF4100	TC3 Controller Toolbox	☑ cpu license
TF4110	TC3 Temperature-Controller	□ cpu license
TF4500	TC3 Speech	□ cpu license

6. Open the **Order Information (Runtime)** tab.
   ⇨ In the tabular overview of licenses, the previously selected license is displayed with the status “missing”**.**

7. Click **7-Day Trial License...** to activate the 7-day trial license.



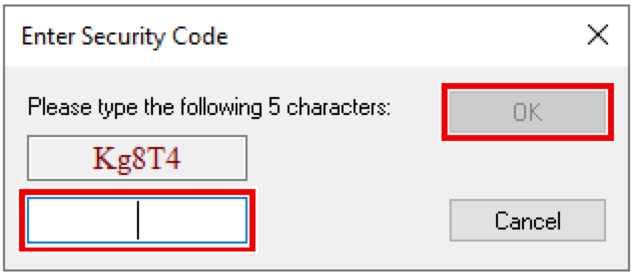
License Device: Target (Hardware Id)

System Id: 27A8E382-5115-364F-CE96-DE17942299A5

Platform: other (91)

Provider: Beckhoff Automation

⇨ A dialog box opens, prompting you to enter the security code displayed in the dialog.



Please type the following 5 characters:

Kg8T4

8. Enter the code exactly as it is displayed and confirm the entry.
9. Confirm the subsequent dialog, which indicates the successful activation.
   ⇨ In the tabular overview of licenses, the license status now indicates the expiry date of the license.
10. Restart the TwinCAT system.

⇨ The 7-day trial version is enabled.

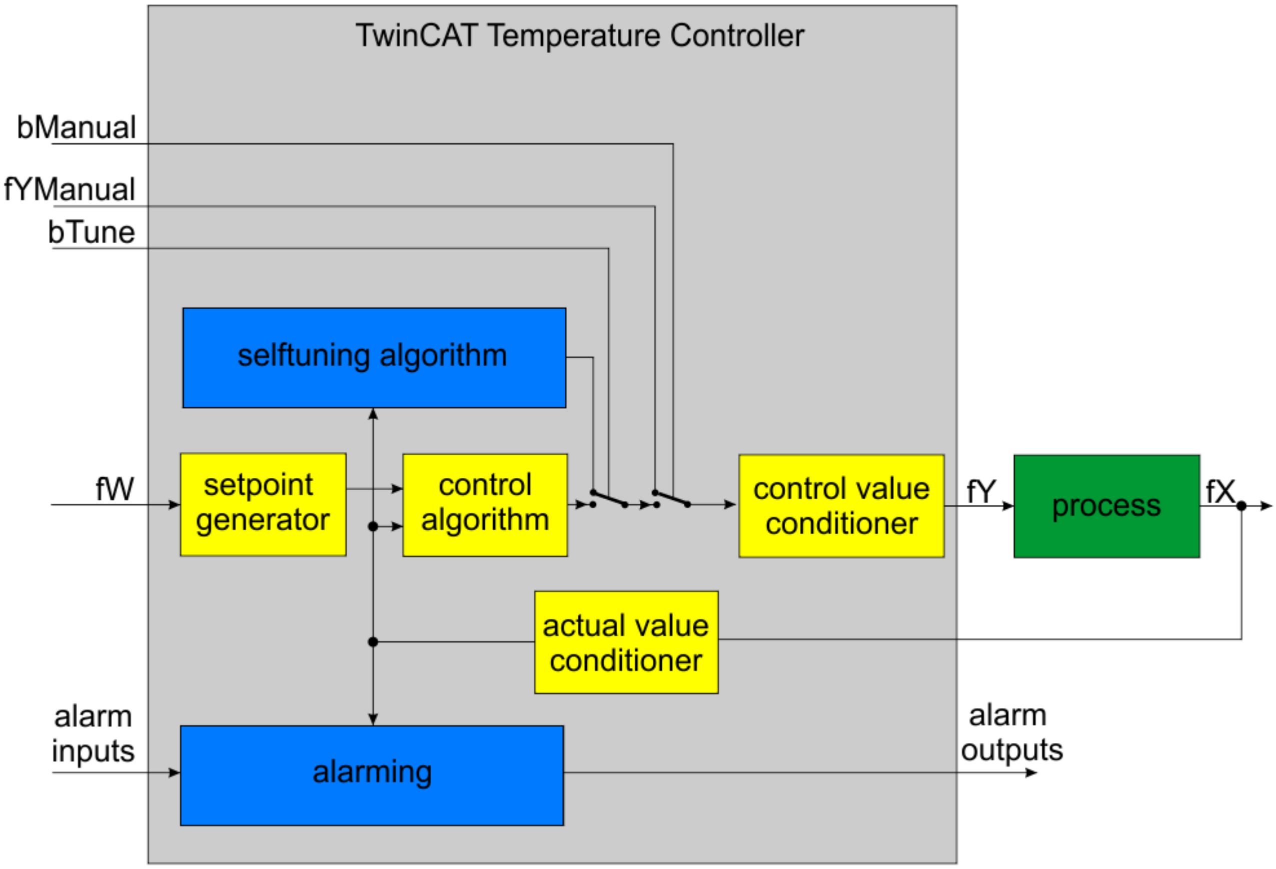
4 Configuration

4.1 Block Diagram

The TwinCAT Temperature Controller consists of a number of function blocks. The following function blocks are involved:

- Self-tuning algorithm (FB_Selftuner)
- Control algorithm (FB_ControlAlgorithm)
- Setpoint generator (FB_SetpointConditioner)
- Control value generator (FB_ControlValueConditioner)
- Alarming (FB_Alarming)

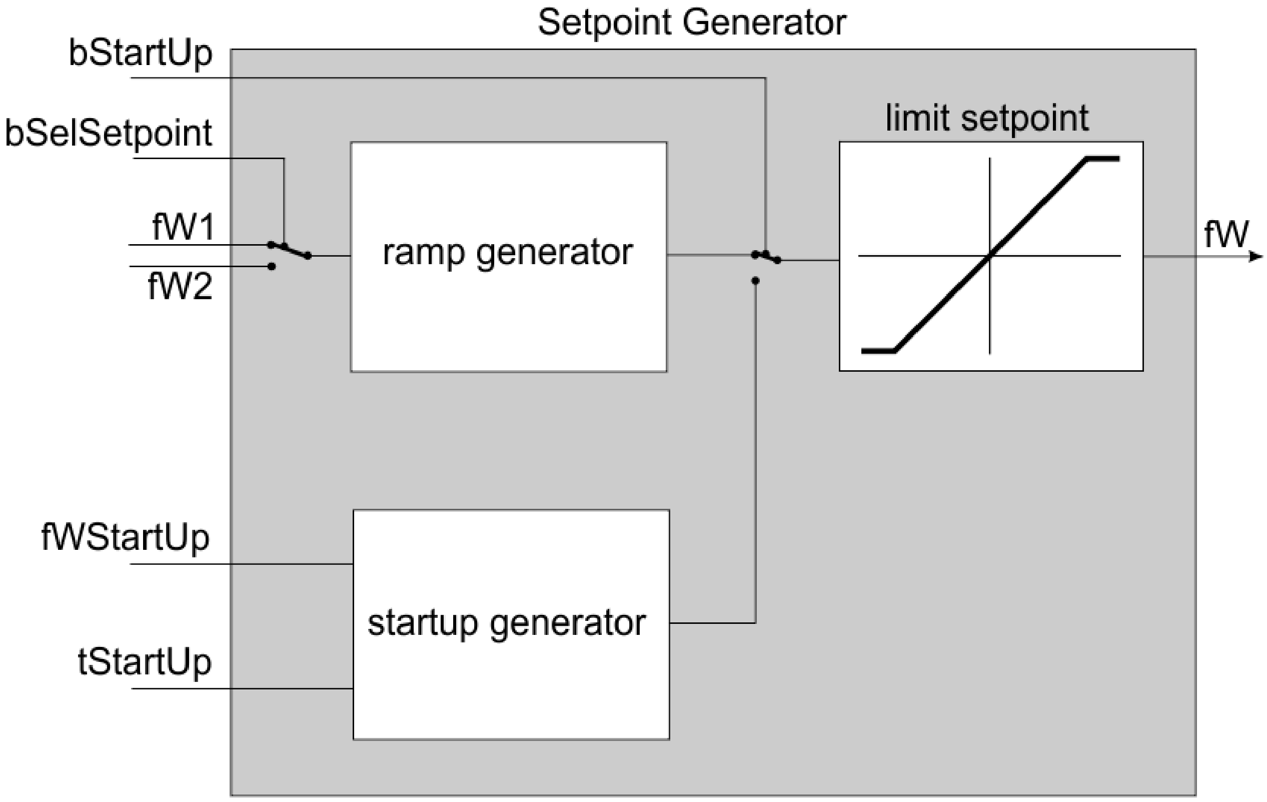
These function blocks in turn call a number of other subsidiary function blocks.



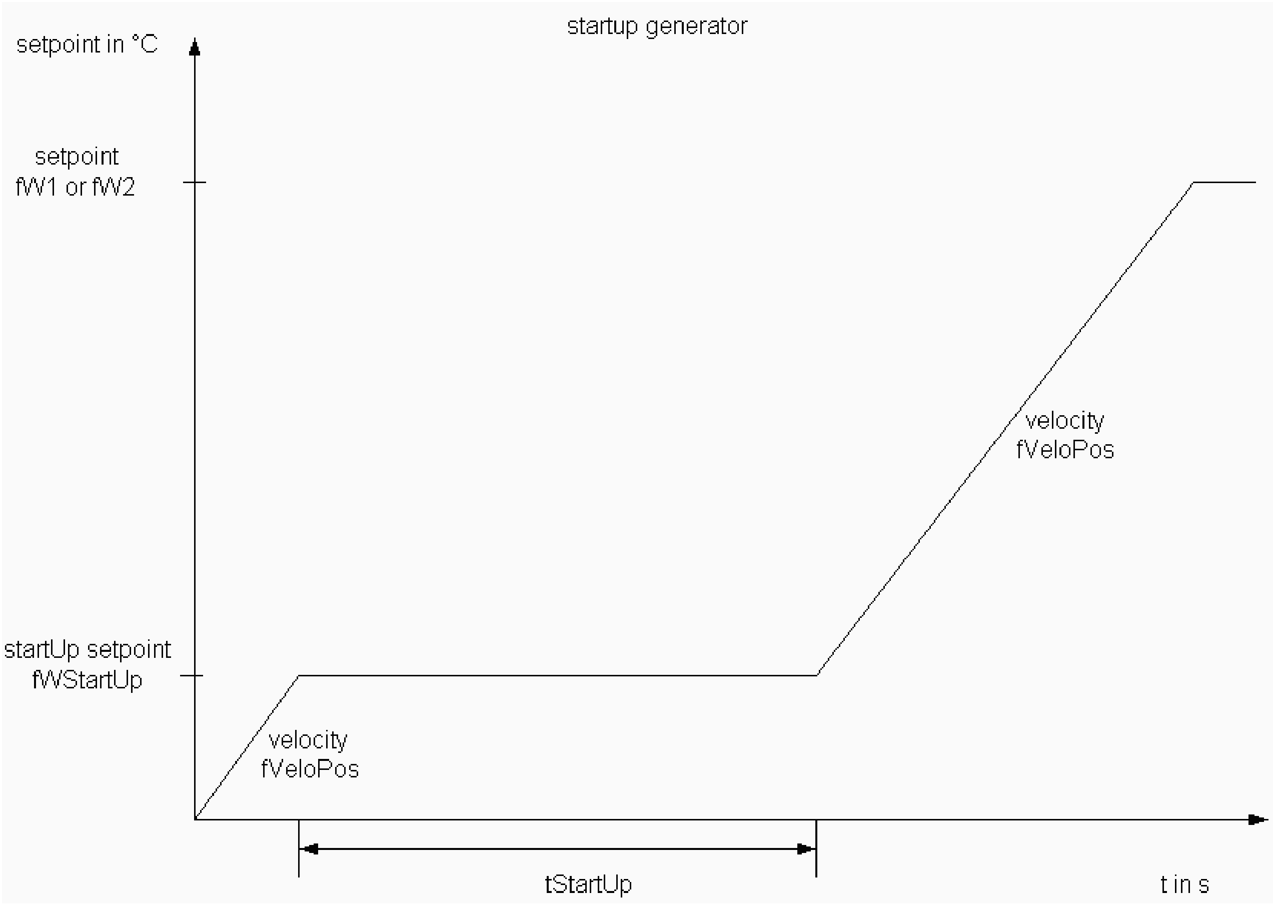
4.2 Generating the Set Value

One bit switches between the setpoints. In addition to the actual setpoint, there is also a standby setpoint. The standby setpoint can be used to reduce the temperature during operating pauses to a lower value in order to save power. If necessary the steps in the setpoint can be ramped. The parameter set for the setpoints includes a rate of rise and a rate of fall.

The setpoints are restricted to their limits.

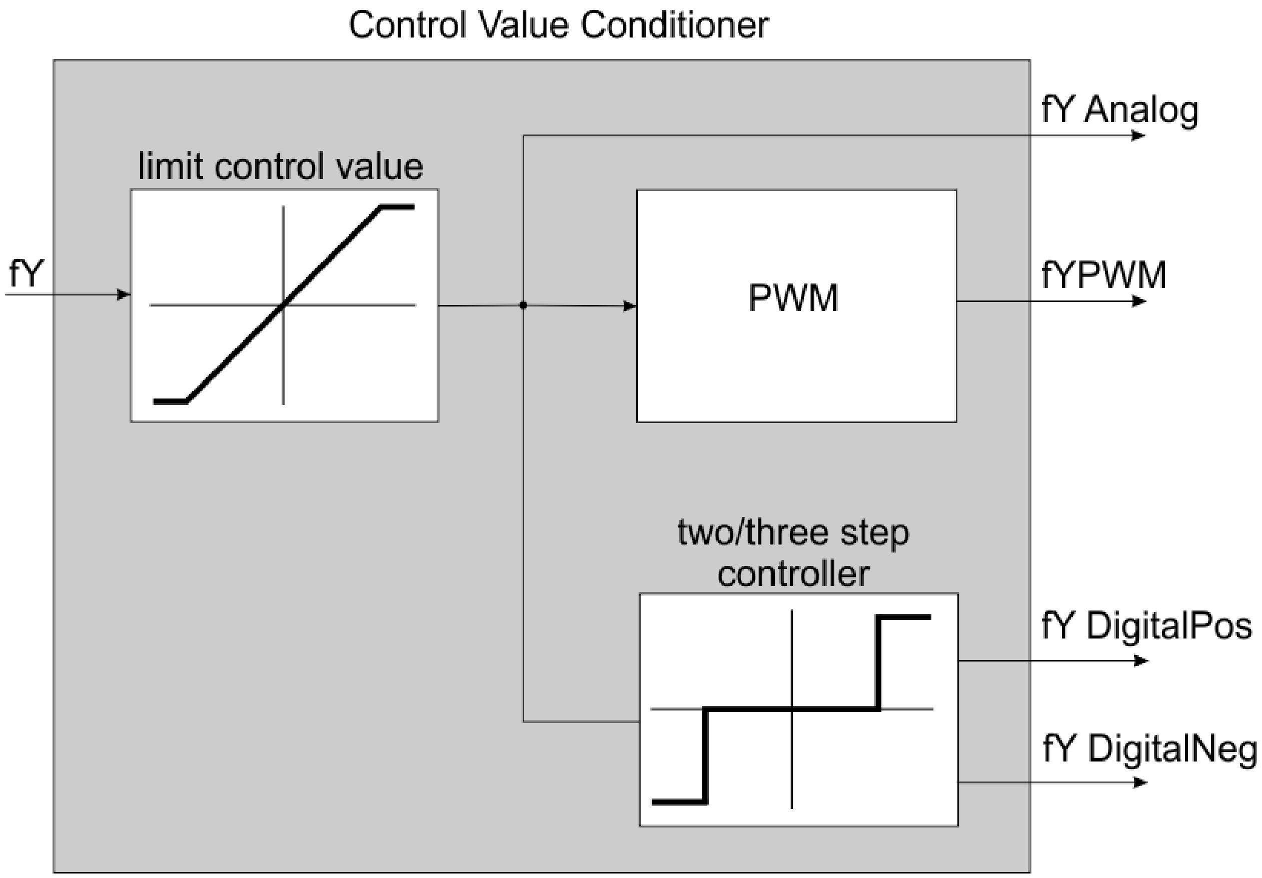


In order to permit "heater baking", a soft start can be parameterized. In this case, the temperature is first ramped up from ambient to a low setpoint (`fWStartUp`). This temperature is then maintained for a period of time (`tStartUp`), and only after that has elapsed does the ramp up to the actual setpoint begin.



4.3 Generating the Control Value

The control value (CV) calculated by the controller is first limited to fall within a valid range. The values of the limits are passed to the controller function block via the control value structure. The control value is made available in three different ways. The control value can be picked up in analog form. However, the more common output is probably the digital output as a pulse width modulated signal. The cycle time required for the pulse width modulation is supplied to the controller in the control value structure. Additionally, a two-point output (for heating or cooling) and a three-point output (for heating and cooling) can be connected.



4.4 Commisioning the Controller in Stages

The following steps must be taken:

1. Insert the controller library into the project via the library manager.
   - Insert the Tc2_TempController in the library manager.
2. Program at least one instance of the controller.
   - To do this, create an instance of the controller function block **FB_TempController**.
   - Also create an instance of the structure **ST_ControllerParameter**.
   - Insert library.
3. Create external wiring.

Name		Description
eCtrlMode	Connection necessary	Switches the controller to an operation mode (active, passive, tuning).
bSelSetpoint	Connection optional	Selects one of two possible setpoints. FALSE selects the normal setpoint, while TRUE selects the standby setpoint.
fW1	Connection necessary	Setpoint
fW2	Connection optional	Standby setpoint is usually smaller than fW1. fSelSetpoint can be used to switch between fW1 and fW2.
fX	Connection necessary	Actual value, expected as LREAL number. The actual value may first have to be converted to LREAL outside the function block.
fYManual	Connection optional	Control value in manual operation
bOpenThermocouple	Connection optional	The thermocouple is open if TRUE. Must be reported by the hardware (e.g. KL3xxx or EL3xxx).
bReverseThermocouple	Connection optional	TRUE indicates that the thermocouple has been connected with the wrong polarity. Must be indicated by the hardware.
bBackVoltage	Connection optional	TRUE indicates that the input voltage at the thermocouple is too high. Must be indicated by the hardware.
bLeakage	Connection optional	TRUE indicates that leakage current has been detected at the heating element. Must be indicated by the hardware.
bShortCircuit	Connection optional	TRUE indicates that a short circuit has been detected at the heating element. Must be indicated by the hardware.
bOpenCircuit	Connection optional	TRUE indicates that an open circuit has been detected at the heating element. Must be indicated by the hardware.
sControllerParameter	Connection necessary	General parameters (sampling time etc.) are passed to the function block in this structure.
sParaControllerExternal	Connection optional	An external controller parameter set is passed to the function block in this structure.

4. Perform the necessary parameterization of the controller via the structure.
   - The parameters can be specified through initial values, or by assignment.
   - If the parameters are assigned by initial values, it could look like this, for example:

```
(* parameters *)
sControllerParameter : ST_CTRL_TempCtrlParameter :=
(
(* base *)
tCtrlCycleTime := t#1000ms,
tTaskCycleTime := t#10ms,

fYMin := -100,
fYMax := 100,
tPWMCycleTime := t#100ms ,
fYManual := 20,
bFilter := FALSE,
tFilter := t#100ms,
bDeadband := FALSE,
fEDeadband := 1.0, (* deadband *)
fWMin := 15,
fWMax := 60,
fWStartUp := 20.0,
tStartUp := t#160s,
fWVeloPos := 0.01,
fWVeloNeg := 0.01,
bStartUpRamping := FALSE,
fWStartUpVeloPos := 0.1,
fWStartUpVeloNeg := 0.1,
iMode := eCTRL_ControlMode_HEATING,
dwAlarmSupp := 16#FF_FF_FF_FF,
bSelCtrlParameterSet:= FALSE,

(* tuninig *)
iTuningMode := eCTRL_TuneMode_heating,
fYTuneHeating := 100.0,
fYTuneCooling := -100.0,
fEndTunePercentHeating := 80.0, (* switch to closed loop control when X > 0.8*W *)
fEndTunePercentCooling := -70.0, (* switch to closed loop control when X < 0.2*W *)

iReactionOnFailure := eCTRL_ReactionOnFailure_StopController,
TempLow := -50.0,
TempLowLow := -100.0,
TempHigh := 100.0,
TempHighHigh := 155.0,
TempAbsoluteHigh := 150.0,
TempAbsoluteLow := -95.0,
bEnablePreController := FALSE,
bEnableZones := FALSE,
bEnableCVFilter := FALSE,
iFilterType := eCTRL_FilterType_AVERAGE,
iControllerType := eCTRL_ControllerType_PID
);
```

Assignment in the code can look like the following in ST:

```
sControllerParameter.tPWMCycleTime  := t#100ms;
```

5. Set the controller sampling time, the task cycle time and the PWM cycle time.
   - The controller's sampling time must be adapted to the section. It should be selected to be equal to or less than one tenth of the loop's dominant time constants.
   - The task cycle time is specified by the PLC task from which the controller function block has been called. This value can be read from the task configuration (PLC Control: Resources Task Configuration). The PWM cycle time is usually equal to the controller cycle time. If the task cycle time is 10ms and the PWM cycle time (=controller sampling time) is chosen to be 100ms, then a total of 10 levels (PWM cycle time / task cycle time) are available.
6. Parameterization of TwinCAT Scope.
   - To check the results, make a scope recording of the tuning process and the closed-loop control behavior.
   - To do this, start and parameterize the TwinCAT Scope View.
   - Record the following channels: setpoint (fW1 or fW2), actual value (fX) and the analog control value (fYAnalog).
7. Switch off alarms during the commissioning phase.
   - The alarms can be temporarily switched off during the commissioning phase.
   - Set a corresponding bit mask in the dwAlarmSupp Dword.
   - If a bit is set in this Dword, the corresponding alarm is disabled. The assignment of the individual alarms is described here [▶ 31].

Caution: After initial commissioning, switch all required alarms back on.

8. Start the controller with tuning.
   - If the controller parameters are to be determined by tuning, the control mode must be set to `eCTRL_MODE_TUNE`.

- A fixed waiting time of 20s first elapses. During this waiting time the temperature is monitored to ensure that it remains within a +-1°C band. If the temperature goes outside this band, the waiting starts again. The process is then subjected to a step excitation with a control value of fYTune. The process then reacts with the step response. As long as 80% of the setpoint is not reached, the process parameters are determined using the inflectional tangent method. For safety reasons, after 80% of the setpoint has been reached, control is switched over to closed loop control. If the temperature reaches the 80% mark too quickly (with no clear inflection) then the value of fYTune is to be reduced. The parameters determined in this way are used for the PID controller, and are provided in a structure at the output of the controller.

NOTE

Setting the control mode

Once the tuning has been completed successfully, the `eCtrlState` is set to `eCTRL_STATE_TUNED`. The controller enters standby mode. Closed-loop operation with the estimated parameters can only be activated by setting the control mode to `eCTRL_MODE_ACTIVE`.

9. Linking the internal control parameters with the external connections.

The controller parameters determined in the tuning process can be supplied again to the controller as external parameters. This may be necessary if the tuning is only to be carried out once (e.g. only during the initial commissioning).

- To do this, trace the structure `sParaControllerInternal` to the input of the controller `sParaControllerExternal` and set the flag `bSelCtrlParameterSet` to TRUE.

10. Perform fine-tuning manually.

The control parameters determined in the tuning process are designed to produce fast settling, with about 10% overshoot. If only very little overshoot is permitted, or even none at all, then the following parameters from the ST_ControllerParameter structure can be used to perform fine tuning. These values are guide values.

Behavior	fTuneKp	fTuneTn	fTuneTv	fTuneTd
Fast settling with overshoots of 10%-20%	1.2	2.0	0.42	0.25
Slower settling with low overshoot	1.0	2.5	0.42	0.25
Almost asymptotic settling with extremely small overshoot	0.5	3.0	1.0	0.25

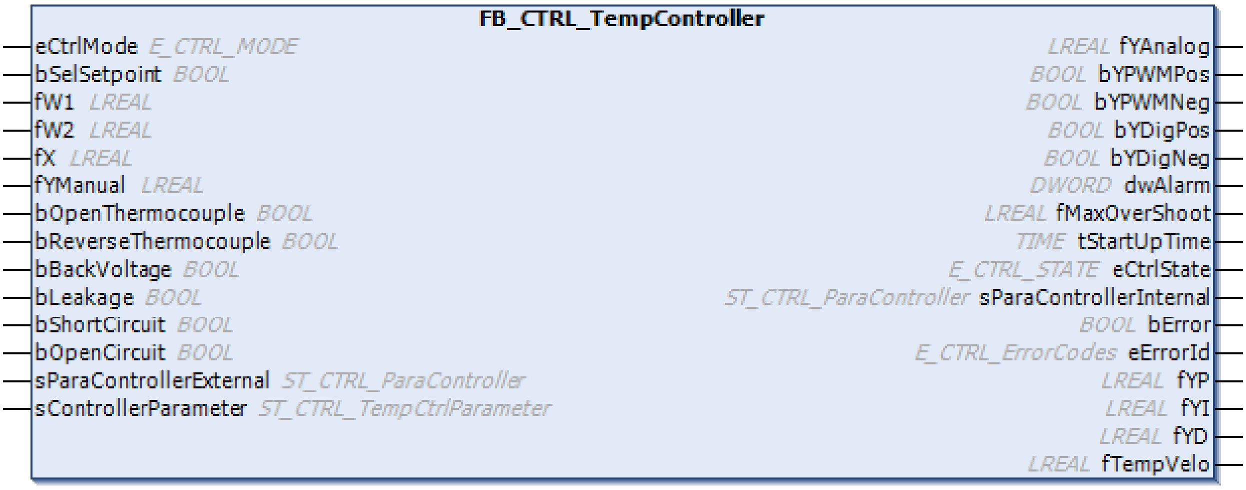
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5 PLC libraries

5.1 Function Block

5.1.1 FB_CTRL_TempController



FB_CTRL_TempController

eCtrlMode *E_CTRL_MODE*	*LREAL* fYAnalog
bSelSetpoint *BOOL*	*BOOL* bYPWMPos
fW1 *LREAL*	*BOOL* bYPWMNeg
fW2 *LREAL*	*BOOL* bYDigPos
fX *LREAL*	*BOOL* bYDigNeg
fYManual *LREAL*	*DWORD* dwAlarm
bOpenThermocouple *BOOL*	*LREAL* fMaxOverShoot
bReverseThermocouple *BOOL*	*TIME* tStartUpTime
bBackVoltage *BOOL*	*E_CTRL_STATE* eCtrlState
bLeakage *BOOL*	*ST_CTRL_ParaController* sParaControllerInternal
bShortCircuit *BOOL*	*BOOL* bError
bOpenCircuit *BOOL*	*E_CTRL_ErrorCodes* eErrorId
sParaControllerExternal *ST_CTRL_ParaController*	*LREAL* fYP
sControllerParameter *ST_CTRL_TempCtrlParameter*	*LREAL* fYI
	LREAL fYD
	LREAL fTempVelo

The temperature controller function block **FB_CTRL_TempController** has various inputs and outputs, which are described below. All the controller's parameters are passed to it via structures.

Interface

VAR_INPUT

```
VAR_INPUT
  eCtrlMode               : E_CTRL_MODE;
  bSelSetpoint            : BOOL;
  fW1                     : LREAL;
  fW2                     : LREAL;
  fX                      : LREAL;
  fYManual                : LREAL;
  bOpenThermocouple       : BOOL; (* thermocouple *)
  bReverseThermocouple    : BOOL;
  bBackVoltage            : BOOL;
  bLeakage                : BOOL; (* heating system *)
  bShortCircuit           : BOOL;
  bOpenCircuit            : BOOL;
  sParaControllerExternal : ST_CTRL_ParaController
END_VAR
```

Name	Unit	Value range	Description
eControlMode	1	E_CTRL_MODE	Mode switching
bSelSetpoint	1	[TRUE,FALSE]	Selects one of two possible setpoints. FALSE selects the normal setpoint, while TRUE selects the standby setpoint.
fW1	°C	LREAL	Setpoint
fW2	°C	LREAL	Standby setpoint is usually smaller than fW1. fSelSetpoint can be used to switch between fW1 and fW2.
fX	°C	LREAL	Actual value
fYManual	-100% - +100%	LREAL	Control value in manual operation
bOpenThermocouple	1	[TRUE,FALSE]	The thermocouple is open if TRUE. Must be indicated by the hardware.
bReverseThermocouple	1	[TRUE,FALSE]	TRUE indicates that the thermocouple has been connected with the wrong polarity. Must be indicated by the hardware.
bBackVoltage	1	[TRUE,FALSE]	TRUE indicates that the input voltage at the thermocouple is too high. Must be indicated by the hardware.
bLeakage	1	[TRUE,FALSE]	TRUE indicates that leakage current has been detected at the heating element. Must be indicated by the hardware.
bShortCircuit	1	[TRUE,FALSE]	TRUE indicates that a short circuit has been detected at the heating element. Must be indicated by the hardware.
bOpenCircuit	1	[TRUE,FALSE]	TRUE indicates that an open circuit has been detected at the heating element. Must be indicated by the hardware.
sControllerParameter	none	Structure	General parameters (sampling time etc.) are passed to the function block in this structure.
sParaControllerExternal	none	Structure	An external controller parameter set is passed to the function block in this structure.

VAR_OUTPUT

```
VAR_OUTPUT
  fYAnalog                 : LREAL;
  bYPWMPos                 : BOOL;
  bYPWMNeg                 : BOOL;
  bYDigPos                 : BOOL;
  bYDigNeg                 : BOOL;
  dwAlarm                  : DWORD;
  fMaxOverShoot            : LREAL;
  tStartUpTime             : TIME;
  eCtrlState               : E_CTRL_STATE := eCTRL_STATE_IDLE;
  sParaControllerInternal : ST_CTRL_ParaController;
  bError                   : BOOL;
  eErrorId                 : E_CTRL_ErrorCodes;
END_VAR
```

Name	Unit	Value range	Description
fYAnalog	none	LREAL	Analog control value
bYPWMPos	none	[TRUE,FALSE]	Boolean output, pulse width modulated. Positive/heating mode
bYPWMNeg	none	[TRUE,FALSE]	Boolean output, pulse width modulated. Negative/cooling mode
bYDigPos	none	[TRUE,FALSE]	Boolean output of a three-step controller (TRUE control value 100%, FALSE control value off)
bYDigNeg	none	[TRUE,FALSE]	Boolean output of a three-step controller (TRUE control value -100%, FALSE control value off)
dwAlarm	none	DWORD	Alarm messages (see ENUM ...)
fMaxOverShoot	°C	LREAL	Max. overshoot in °C above/below setpoint.
tStartUpTime	TIME	-	Startup time until the setpoint is reached for the first time
eCtrlState	none	E_CTRL_STATE	current controller status (see ENUM ...)
sParaControllerInternal	none	Structure	In this structure the internal controller parameter set (determined by the tuning) is made available.
bError	none	[TRUE,FALSE]	If an error is present, then `bError` is TRUE.
iErrorId	none	INT	If `bError` is TRUE, then `iErrorId` provides an error code (see ENUM ...)

VAR_IN_OUT

```
VAR_IN_OUT
  sControllerParameter    : ST_CTRL_TempCtrlParameter; (* controller parameter set *)
END_VAR
```

Name	Type	Description
sController parameter	ST_CTRL_TempCtrl Parameter	Parameter structure of the function block

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5.1.2 Structure definitions

ST_ControllerParameter

```
TYPE ST_CTRL_TempCtrlParameter:
STRUCT

  (* general parameters *)
  iMode                      : E_CTRL_ControlMode;
  iReactionOnFailure         : E_CTRL_ReactionOnFailure;
  bSelCtrlParameterSet       : BOOL;
  dwAlarmSupp                : DWORD;
  tCtrlCycleTime             : TIME;
  tTaskCycleTime             : TIME;

  (* tuning parameteer *)
  iTuningMode                : E_CTRL_TuneMode;
  tTuneStabilisation         : TIME       := T#20S;
  fEndTunePercentHeating     : LREAL      := 80.0;
  fYTuneHeating              : LREAL;
  fYStableHeating            : LREAL;
  fEndTunePercentCooling     : LREAL      := 20.0;
  fYTuneCooling              : LREAL;
  fYStableCooling            : LREAL;
  fScalingFactor             : LREAL      := 1.0;

  (* setpoint parameters *)
  fWMin                      : LREAL;
  fWMax                     : LREAL;

  (* start up *)
  bEnableSoftStart          : BOOL;
  bEnableRamping            : BOOL;
  fWStartUp                 : LREAL;
  tStartUp                  : TIME;
  bStartUpRamping           : BOOL;
  fWStartUpVeloPos          : LREAL;
  fWStartUpVeloNeg          : LREAL;
  fWVeloPos                 : LREAL;
  fWVeloNeg                 : LREAL;

  (* actual value parameters *)
  bFilter                   : BOOL;
  tFilter                   : TIME;

  (* deadband parameters *)
  bDeadband                 : BOOL;
  fEDeadband                : LREAL;

  (* control value parameters *)
  fYMin                     : LREAL;
  fYMax                     : LREAL;
  fYManual                  : LREAL;
  fYOnFailure               : LREAL;
  tPWMCycleTime             : TIME;
  tPWMMinOffTime            : TIME;
  tPWMMinOnTime             : TIME;
  tPWMWaitingTime           : TIME;
  fYThresholdOff            : LREAL;
  fYThresholdOn             : LREAL;
  nCyclesForSwitchOver      : INT        := 100;

  (* controller settings *)
  bEnablePreController      : BOOL;
  bEnableZones              : BOOL;
  bEnableCVFilter           : BOOL;
  iFilterType               : E_CTRL_FilterType;
  iControllerType           : E_CTRL_ControllerType;

  (* min max temperatures *)
  TempLow                   : LREAL;
  TempLowLow                : LREAL;
  TempHigh                  : LREAL;
  TempHighHigh              : LREAL;
  TempAbsoluteHigh          : LREAL;
  TempAbsoluteLow           : LREAL;

  (* internal tuning parameters *)
  fTuneKp                   : LREAL     := 1.2;
  fTuneTn                   : LREAL     := 2.0;
  fTuneTv                   : LREAL     := 0.42;
  fTuneTd                   : LREAL     := 0.25;
END_STRUCT
END_TYPE
```

Name	Unit	Value range	Description
General parameters			
iMode	none	INT	Controller operation mode (1 = heating, 2 = cooling, 3 = heating & cooling) (see below)
iReactionOnFailure	none	INT	Parameterizable reaction to errors (see below)
bSelCtrlParameterSet	none	BOOL	TRUE = external parameter set, FALSE = internal parameter set (determined by tuning)
dwAlarmSupp	none	DWORD	Masks out the alarms (see below)
tCtrlCycleTime	s	TIME	Controller's sampling time. In the course of the sampling time the controller re-calculates the control value.
tTaskCycleTime	s	TIME	Task cycle time. The FB is called with this time interval.
Tuning parameters			
iTuningMode	K	E_CTRL_TuneMode	Determination of the tuning sequence (see below.)
tTune stabilization	s	TIME	Waiting time until the section is stable for tuning.
fEndTunePercentHeating	%	LREAL	Percentage value of setpoint, from which the system switches to Closed Loop Control.
fYTuneCooling	K	LREAL	Step change in control value during tuning.
fYStableCooling	K	LREAL	Control value when switching to tuning during cooling.
fScalingFactor	none	LREAL	Scaling factor for parameter switching if no tuning is performed for cooling.
Setpoint parameters			
fWMin	K	LREAL	Minimum setpoint
fWMax	K	LREAL	Maximum setpoint
bEnableSoftStart	none	BOOL	FALSE = no soft start, TRUE = soft start
bEnableRamping	none	BOOL	FALSE = no ramping, TRUE = ramping
fWStartUp	K	LREAL	Setpoint at start-up
tStartUp	s	TIME	Time with the fWStartUp setpoint
bStartUpRamping	none	[TRUE,FALSE]	Switches on ramping during the start-up phase.
fWStartUpVeloPos	K/s	LREAL	Rate of rise (of ramp) during the start-up phase
fWStartUpVeloNeg	K/s	LREAL	Rate of fall (of ramp) during the start-up phase
fWVeloPos	K/s	LREAL	Rate of rise (of ramp)
fWVeloNeg	K/s	LREAL	Rate of fall (of ramp).
Actual value parameters			
tFilter	s	TIME	Time constant of the actual value filter (first order P-T1 filter)
bFilter	none	[TRUE,FALSE]	The actual value filter is actuated if TRUE.
Deadband parameters			
bDeadband	none	[TRUE,FALSE]	TRUE = deadband on, FALSE = deadband off
fEDeadband	K	LREAL	Deadband in degrees
Control value parameters			

Name	Unit	Value range	Description
fYMin	none	LREAL	Minimum value of the control value
fYMax	none	LREAL	Maximum value of the control value
fYManual	none	LREAL	Control value in manual operation
fYOnFailure	none	LREAL	Control value in case of error (parameterizable)
tPWMCycleTime	s	TIME	Cycle time of the PWM signal
tPWMMinOffTime	s	TIME	PWM: minimum switch-off time
tPWMMinOnTime	s	TIME	PWM: minimum switch-on time
tPWMWaitingTime	s	TIME	PWM: Waiting time when switching from heating to cooling
fYThresholdOff	%	LREAL	3-point: Switch-off threshold
fYThresholdOn	%	LREAL	3-point: Switch-on threshold
nCyclesForSwitchOver	none	INT	Number of cycles for transition from one parameter set to another
Controller parameters			
bEnablePreController	none	[TRUE,FALSE]	Switches pre-controller on.
bEnableZones	none	[TRUE,FALSE]	Switches open loop characteristic on until close to setpoint.
bEnableCVFilter	none	[TRUE,FALSE]	Switches on control value filter following the main controller.
iFilterType	none	ENUM	Selection of a filter type for the control value filter following the main controller (see below).
iControllerType	none	ENUM	Selection of a control algorithm (see below).
Alarming parameters			
TempLow	K	LREAL	Relative lower temperature limit in the first band
TempLowLow	K	LREAL	Relative lower temperature limit in the second band
TempHigh	K	LREAL	Relative upper temperature limit in the first band
TempHighHigh	K	LREAL	Relative upper temperature limit in the second band
TempAbsoluteHigh	K	LREAL	Absolute upper temperature limit
TempAbsoluteLow	K	LREAL	Absolute lower temperature limit
Expert parameters			
fTuneKp	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)
fTuneTn	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)
fTuneTv	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)
fTuneTd	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)

ST_CTRL_ParaController

```
TYPE ST_CTRL_ParaController :
STRUCT
  (* Controller parameter set - heating *)
  KpHeat     : FLOAT;
  TnHeat     : TIME;
  TvHeat    : TIME;
  TdHeat    : TIME;
  (* Controller parameter set - cooling *)
  KpCool    : FLOAT;
  TnCool    : TIME;
  TvCool    : TIME;
  TdCool    : TIME;
END_STRUCT
END_TYPE
```

Name	Unit	Value range	Description
KpHeat	none	LREAL	Gain factor for the main controller
TnHeat	s	TIME	Integral action time for main controller (I component)
TvHeat	s	TIME	Derivative action time for main controller (D component)
TdHeat	s	TIME	Damping time for the main controller
KpCool	none	LREAL	Gain factor for the main controller
TnCool	s	TIME	Integral action time for main controller (I component)
TvCool	s	TIME	Derivative action time for main controller (D component)
TdCool	s	TIME	Damping time for the main controller

ENUM: E_CTRL_ERRORCODES

See documentation of the TwinCAT controller toolbox.

ENUM: E_CTRL_ReactionOnFailure

Name	Description
eCTRL_ReactionOnFailure_NoFailure	No error
eCTRL_ReactionOnFailure_StopController	If there is an error (an alarm) the controller will stop.
eCTRL_ReactionOnFailure_SetManMode	If there is an error (an alarm) the controller will switch to manual operation.
eCTRL_ReactionOnFailure_SetYMax	If there is an error (an alarm) set the control value to its maximum.
eCTRL_ReactionOnFailure_SetYMin	If there is an error (an alarm) set the control value to its minimum.
eCTRL_ReactionOnFailure_SetYMean	If error (alarm), set control value to average value (in preparation).

ENUM: E_CTRL_ControllerStateInternal

Name	Description
E_CTRL_ControllerStateInternalHeating	internal
E_CTRL_ControllerStateInternalCooling	internal

ENUM: E_CTRL_ControlMode

Name	Description
eCTRL_ControlMode_HEATING	Heating only
eCTRL_ControlMode_COOLING	Cooling only
eCTRL_ControlMode_HEATING_COOLING	Heating and cooling

ENUM: E_CTRL_STATE

See documentation of the TwinCAT controller toolbox.

ENUM: E_CTRL_STATE_TUNIG

Name	Description
eCTRL_STATE_TUNING_INIT	Tuning: Initialization
eCTRL_STATE_TUNING_IDLE	Tuning: Waiting for stable input signal (control variable)
eCTRL_STATE_TUNING_PULSE	Tuning: Excitation through short pulse (in preparation)
eCTRL_STATE_TUNING_STEP	Tuning: Excitation through step change (inflectional tangent method)
eCTRL_STATE_TUNING_READY	Tuning: Determining the parameters, finalization
eCTRL_STATE_TUNING_ERROR	Tuning: Error during tuning

ENUM: E_CTRL_TuneMode

Name	Description
eCTRL_TuneMode_HEATING	Tuning: heating only
eCTRL_TuneMode_COOLING	Tuning: cooling only
eCTRL_TuneMode_HEATING_COOLING	Tuning: first heating, then cooling
eCTRL_TuneMode_COOLING_HEATING	Tuning: first cooling, then heating
eCTRL_TuneMode_OSCILLATION	Tuning: on-the-fly parameter estimation through vibration excitation (in preparation)

ENUM: E_CTRL_FilterType

Name	Description
eCTRL_FilterType_FIRSTORDER	First order filter
eCTRL_FilterType_AVERAGE	Mean value filter

ENUM: E_CTRL_ControllerType

Name	Description
eCTRL_ControllerType_PID	Standard PID control algorithm
eCTRL_ControllerType_PI	Standard PI control algorithm
eCTRL_ControllerType_PID_Pre	Standard PID control algorithm with pre-controller (in preparation)
eCTRL_ControllerType_PIDD2	Serial PID control algorithm (in preparation)

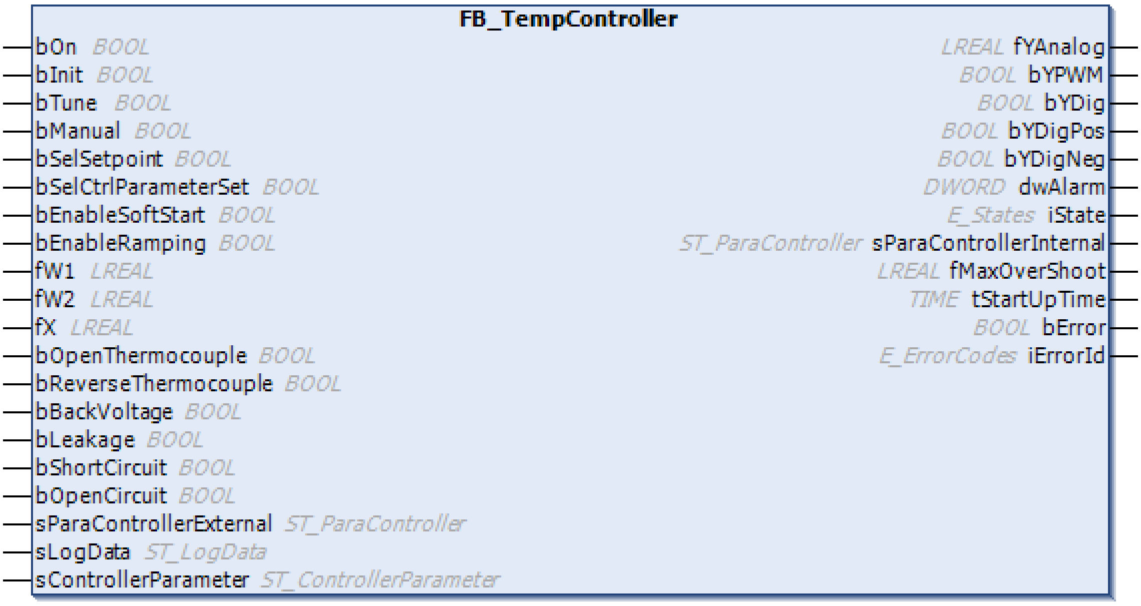
Bit-masks for alarms

Name	Mask	Description
nAlarmOpen Thermocouple	2#0000_0000_0000_0000_0000_0000_0000_0001	Hardware: open temperature sensor
nAlarmReverse Thermocouple	2#0000_0000_0000_0000_0000_0000_0000_0010	Hardware: reverse connected temperature sensor
nAlarmBackVoltage	2#0000_0000_0000_0000_0000_0000_0000_0100	Hardware: excessive voltage at temperature sensor
nAlarmLeakage Current	2#0000_0000_0000_0000_0000_0000_0000_1000	Hardware: leakage current measured
nAlarmShortCircuit	2#0000_0000_0000_0000_0000_0000_0001_0000	Hardware: Short circuit
nAlarmOpenCircuit	2#0000_0000_0000_0000_0000_0000_0010_0000	Hardware: no current
nAlarmLimitLow	2#0000_0000_0000_0000_0000_0001_0000_0000	Software: fallen below first lower relative temperature
nAlarmLimitLowLow	2#0000_0000_0000_0000_0000_0010_0000_0000	Software: fallen below second lower relative temperature
nAlarmLimitHigh	2#0000_0000_0000_0000_0000_0100_0000_0000	Software: first upper relative temperature exceeded
nAlarmLimitHighHigh	2#0000_0000_0000_0000_0000_1000_0000_0000	Software: second upper relative temperature exceeded
nAlarmAbsoluteHigh	2#0000_0000_0000_0000_0001_0000_0000_0000	Software: upper absolute temperature exceeded
nAlarmAbsoluteLow	2#0000_0000_0000_0000_0010_0000_0000_0000	Software: fallen below lower absolute temperature

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5.1.3 FB_TempController



The temperature controller function block **FB_TempController** has a variety of inputs and outputs that are described below. All the controller's parameters are passed to it via structures. The definition of the structures and enums can be found here [▶ 31].

NOTE

This version of the function block is obsolete.

You should no longer use this function block.

Interface

VAR_INPUT

```
VAR_INPUT
  bOn                        : BOOL;
  bInit                      : BOOL;
  bTune                      : BOOL;
  bManual                    : BOOL;
  bSelSetpoint               : BOOL;
  bSelCtrlParameterSe        : BOOL;
  bEnableSoftStart           : BOOL;
  bEnableRamping             : BOOL;
  fW1                        : LREAL;
  fW2                        : LREAL;
  fX                         : LREAL;
  bOpenThermocouple          : BOOL;
  bReverseThermocouple       : BOOL;
  bBackVoltage               : BOOL;
  bLeakage                   : BOOL;
  bShortCircuit              : BOOL;
  bOpenCircuit               : BOOL;
  sParaControllerExternal    : ST_ParaController;
  sLogData                   : ST_LogData := (bLog := FALSE, strLogFileName :='', strLogString :=
'' );
END_VAR
```

Name	Unit	Value range	Description
bOn	1	[TRUE,FALSE]	TRUE switches the controller on.
bInit	1	[TRUE,FALSE]	Initialization flag, which must be active (TRUE) for precisely the first cycle in which the controller is called.
bTune	1	[TRUE,FALSE]	A rising edge switches the self-tuning on. If it is switched to FALSE during the self-tuning process then the self-tuning is aborted and the controller continues operation using the old parameters (if they are still present).
bManual	1	[TRUE,FALSE]	TRUE switches manual operation on. If the signal goes FALSE again, the controller returns to automatic mode.
bSelSetpoint	1	[TRUE,FALSE]	Selects one of two possible setpoints. FALSE selects the normal setpoint, while TRUE selects the standby setpoint.
bSelCtrlParameterSet	1	[TRUE,FALSE]	Selects one of two parameter sets. FALSE causes the internal (determined) parameter set to be used, while TRUE switches to one provided externally.
bEnableSoftStart	1	[TRUE,FALSE]	The soft start-up process is used if TRUE.
bEnableRamping	1	[TRUE,FALSE]	TRUE causes each setpoint step-change to be converted to a ramp.
fW1	°C	LREAL	Setpoint
fW2	°C	LREAL	Standby setpoint, generally smaller than fW1. fSelSetpoint can be used to switch between fW1 and fW2.
fX	°C	LREAL	Actual value - this value must be converted to LREAL.
bOpenThermocouple	1	[TRUE,FALSE]	The thermocouple is open if TRUE. Must be indicated by the hardware (e.g. KLxxxx).
bReverseThermocouple	1	[TRUE,FALSE]	TRUE indicates that the thermocouple has been connected with the wrong polarity. Must be indicated by the hardware.
bBackVoltage	1	[TRUE,FALSE]	TRUE indicates that the input voltage at the thermocouple is too high. Must be indicated by the hardware.
bLeakage	1	[TRUE,FALSE]	TRUE indicates that leakage current has been detected at the heating element. Must be indicated by the hardware.
bShortCircuit	1	[TRUE,FALSE]	TRUE indicates that a short circuit has been detected at the heating element. Must be indicated by the hardware.
bOpenCircuit	1	[TRUE,FALSE]	TRUE indicates that an open circuit has been detected at the heating element. Must be indicated by the hardware.
sControllerParameter	none	Structure	General parameters (sampling time etc.) are passed to the function block in this structure.
sParaControllerExternal	none	Structure	An external controller parameter set is passed to the function block in this structure.
sLogData	none	Structure	This structure passes parameters for logging to the function block (filenames etc.).

VAR_OUTPUT

```
VAR_OUTPUT
  fYAnalog                    : LREAL;
  bYPWM                       : BOOL;
  bYDig                       : BOOL;
  bYDigPos                    : BOOL;
  bYDigNeg                    : BOOL;
  dwAlarm                     : DWORD;
  iState                      : States        := TC_STATE_IDLE;
  sParaControllerInternal     : ST_ParaController;
  bError                     : BOOL;
  iErrorId                   : ErrorCodes;
END_VAR
```

Name	Unit	Value range	Description
fYAnalog	none	LREAL	Analog control value
bYPWM	none	[TRUE,FALSE]	Boolean output, pulse width modulated
bYDig	none	[TRUE,FALSE]	Boolean output of an on-off controller (TRUE control value 100%, FALSE control value off)
bYDigPos	none	[TRUE,FALSE]	Boolean output of a three-step controller (TRUE control value 100%, FALSE control value off)
bYDigNeg	none	[TRUE,FALSE]	Boolean output of a three-step controller (TRUE control value -100%, FALSE control value off)
dwAlarm	none	DWORD	Alarm messages (see ENUM ...)
iState	none	INT	Current controller status (see ENUM ...)
sParaControllerInternal	none	Structure	In this structure the internal controller parameter set (determined by the tuning) is made available.
bError	none	[TRUE,FALSE]	If an error is present, then bError is TRUE.
iErrorId	none	INT	If bError is TRUE, then iErrorId provides an error code (see ENUM ...).

VAR_IN_OUT

```
VAR_IN_OUT
  sControllerParameter       : ST_ControllerParameter;
END_VAR
```

Name	Type	Description
sController parameter	ST_ControllerParameter	Parameter structure of the function block

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5.1.4 Structure Definitions

NOTE

This version of the function block is obsolete.

You should no longer use this function block.

```
TYPE ST_ParaControlValue :
STRUCT

  (* general parameters *)
  iMode                        : E_ControlMode;
  iReactionOnFailure         : E_ReactionOnFailure;
  fYTune                     : LREAL;
  fYStable                   : LREAL;
  dwAlarmSupp                : DWORD;
  tCtrlCycleTime             : TIME;
  tTaskCycleTime             : TIME;

  (* setpoint parameters *)
  fWMin                      : LREAL;
  fWMax                      : LREAL;

  (* start up *)
  fWStartUp                  : LREAL;
  tStartUp                   : TIME;
  bStartUpRamping            : BOOL;
  fWStartUpVeloPos           : LREAL;
  fWStartUpVeloNeg      : LREAL;
  fWVeloPos             : LREAL;
  fWVeloNeg             : LREAL;

  (* actual value parameters *)
  bFilter               : BOOL;
  tFilter               : TIME;

  (* control value parameters *)
  fYMin                 : LREAL;
  fYMax                 : LREAL;
  fYManual              : LREAL;
  fYOnFailure           : LREAL;
  tPWMCycleTime         : TIME;

  (* controller settings *)
  bEnablePreController  : BOOL;
  bEnableZones          : BOOL;
  bEnableCVFilter       : BOOL;
  iFilterType           : E_FilterType;
  iControllerType       : E_ControllerType;

  (* min max temperatures *)
  TempLow               : LREAL;
  TempLowLow            : LREAL;
  TempHigh              : LREAL;
  TempHighHigh          : LREAL;
  TempAbsoluteHigh      : LREAL;
  TempAbsoluteLow       : LREAL;

  (* internal tuning parameters *)
  fTuneKp               : LREAL := 1.2;
  fTuneTn               : LREAL := 2.0;
  fTuneTv               : LREAL := 0.42;
  fTuneTd               : LREAL := 0.25;
END_STRUCT
END_TYPE
```

ST_ControllerParameter

Name	Unit	Value range	Description
iMode	none	INT	Controller operation mode (1 = heating, 2 = cooling, 3 = heating & cooling) (see below)
iReactionOnFailure	none	INT	Parameterizable reaction to errors (see below)
fYTune	none	LREAL	Control value during the self-tuning (normally 100%)
fYStable	none	LREAL	Control value during the settling phase (normally 0%)
dwAlarmSupp	none	DWORD	Masks out the alarms (see below)
tCtrlCycleTime	s	TIME	Controller's sampling time. In the course of the sampling time the controller re-calculates the control value.
tTaskCycleTime	s	TIME	Task cycle time. The FB is called with this time interval.
fWMin	K	LREAL	Minimum setpoint
fWMax	K	LREAL	Maximum setpoint
fWVeloPos	K/s	LREAL	Rate of rise (of ramp)
fWVeloNeg	K/s	LREAL	Rate of fall (of ramp).
fWStartUp	K	LREAL	Setpoint at start-up
tStartUp	s	TIME	Time with the fWStartUp setpoint
bStartUpRamping	none	[TRUE,FALSE]	Switches on ramping during the start-up phase.
fWStartUpVeloPos	K/s	LREAL	Rate of rise (of ramp) during the start-up phase
fWStartUpVeloNeg	K/s	LREAL	Rate of fall (of ramp) during the start-up phase
fYMin	none	LREAL	Minimum value of the control value
fYMax	none	LREAL	Maximum value of the control value
fYManual	none	LREAL	Control value in manual operation
fYOnFailure	none	LREAL	Control value in case of error (parameterizable)
tPWMCycleTime	s	TIME	Cycle time of the PWM signal
tFilter	s	TIME	Time constant of the actual value filter (first order P-T1 filter)
bFilter	none	[TRUE,FALSE]	The actual value filter is actuated if TRUE.
bEnablePreController	none	[TRUE,FALSE]	Switches pre-controller on.
bEnableZones	none	[TRUE,FALSE]	Switches open loop characteristic on until close to setpoint.
bEnableCVFilter	none	[TRUE,FALSE]	Switches on control value filter following the main controller.
iFilterType	none	ENUM	Selection of a filter type for the control value filter following the main controller (see below).
iControllerType	none	ENUM	Selection of a control algorithm (see below).
TempLow	K	LREAL	Relative lower temperature limit in the first band
TempLowLow	K	LREAL	Relative lower temperature limit in the second band
TempHigh	K	LREAL	Relative upper temperature limit in the first band
TempHighHigh	K	LREAL	Relative upper temperature limit in the second band
TempAbsoluteHigh	K	LREAL	Absolute upper temperature limit
TempAbsoluteLow	K	LREAL	Absolute lower temperature limit
fTuneKp	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)
fTuneTn	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)

Name	Unit	Value range	Description
fTuneTv	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)
fTuneTd	none	LREAL	FineTuning parameters for the PID controller (only for advanced users)

ST_ParaController

```
TYPE ST_ParaController :
STRUCT
  (* Main Controller parameter set *)
  KpMain     : LREAL;
  TnMain     : LREAL;
  TvMain     : LREAL;
  TdMain     : LREAL;
  (* Pre Controller parameter set *)
  KpPre      : LREAL;
  TvPre      : LREAL;
  TdPre      : LREAL;
END_STRUCT
END_TYPE
```

Name	Unit	Value range	Description
KpMain	none	LREAL	Gain factor for the main controller
TnMain	s	TIME	Integral action time for main controller (I component)
TvMain	s	TIME	Derivative action time for main controller (D component)
TdMain	s	TIME	Damping time for the main controller
KpPre	none	LREAL	Amplification factor for the pre-controller
TvPre	s	TIME	Derivative action time for pre-controller (D component)
TdPre	s	TIME	Damping time for the pre-controller

Table 1: ENUM: Error codes

Name	Description
TC_ERR_NOERROR	No error
TC_ERR_INVALIDPARAM	Invalid parameter
TC_ERR_NO_INIT	Missing function block initialization.
TC_ERR_NO_INFLECTION_POINT	No inflection was found during self-tuning. No parameters could be determined.
TC_ERR_INVALID_PARAM	Invalid parameter
TC_ERR_INVALID_CYCLETIME	Invalid combination of cycle times (sampling times and PWM cycle times)
TC_ERR_WRONG_TU	A valid value for the Tu parameter could not be found due to faulty or aborted self-tuning.

Table 2: ENUM: ReactionOnFailure

Name	Description
TC_OnFailureNoFailure	No error
TC_OnFailureStopController	If there is an error (an alarm) the controller will stop.
TC_OnFailureSetManMode	If there is an error (an alarm) the controller will switch to manual operation.
TC_OnFailureSetYMax	If there is an error (an alarm) set the control value to its maximum.
TC_OnFailureSetYMin	If there is an error (an alarm) set the control value to its minimum.

Table 3: ENUM: ST_ControlMode

Name	Description
CTRLMODE_HEATING	Heating only
CTRLMODE_COOLING	Cooling only
CTRLMODE_HEATING_COOLING	Heating and cooling

Table 4: ENUM: states

Name	Description
TC_STATE_IDLE	Controller switched off.
TC_STATE_INIT	Controller is being initialized.
TC_STATE_OFF	Controller switched off, was previously switched on.
TC_STATE_TUNE	Controller in tuning / self adjustment state.
TC_STATE_MANUAL_OPERATION	Controller in manual operation.
TC_STATE_CLOSED_LOOP	Controller in automatic operation.
TC_STATE_TUNE_IDLE	Tuning started but not yet running. Waiting for idle.
TC_STATE_TUNE_PULSE	Pulse for determination of dead time.
TC_STATE_TUNE_STEP	Step for determination of dead time and maximum velocity.
TC_STATE_TUNE_READY	Self-tuning complete.
TC_STATE_ERROR	Error (logical error)

Table 5: ENUM: E_FilterType

Name	Description
E_FilterType_FIRSTORDER	First order filter
E_FilterType_AVERAGE	Mean value filter

Table 6: ENUM: E_ControllerType

Name	Description
E_ControllerType_PID	Standard PID control algorithm
E_ControllerType_PIDD2	Planned serial PID control algorithm

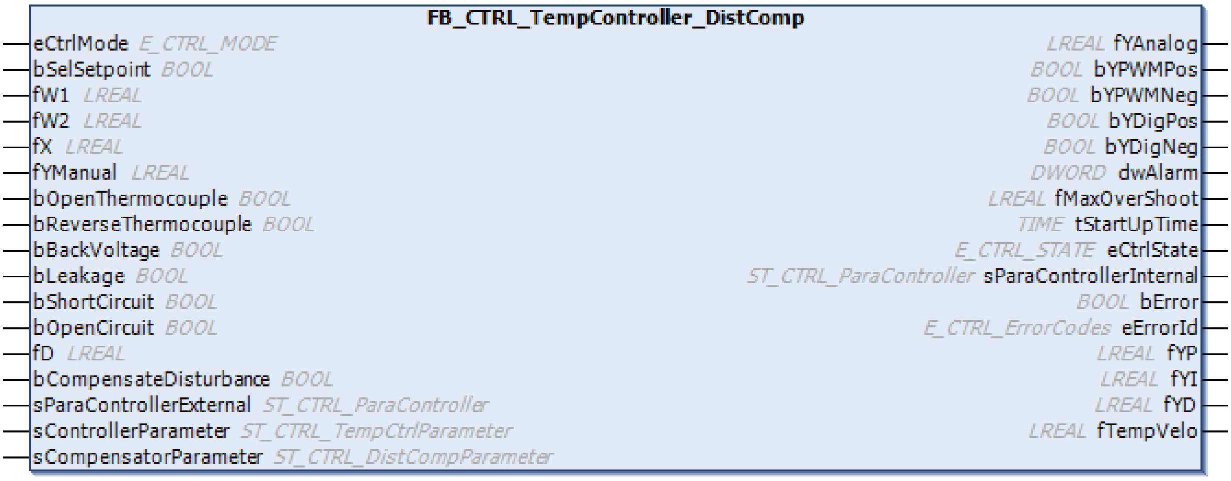
Bit-masks for alarms

Name	Mask	Description
nAlarmOpen Thermocouple	2#0000_0000_0000_0000_0000_0000_0000_0001	Hardware: open temperature sensor
nAlarmReverse Thermocouple	2#0000_0000_0000_0000_0000_0000_0000_0010	Hardware: reverse connected temperature sensor
nAlarmBack Voltage	2#0000_0000_0000_0000_0000_0000_0000_0100	Hardware: excessive voltage at temperature sensor
nAlarmLeakage Current	2#0000_0000_0000_0000_0000_0000_0000_1000	Hardware: Measured leakage current.
nAlarmShortCircuit	2#0000_0000_0000_0000_0000_0000_0001_0000	Hardware: Short circuit
nAlarmOpenCircuit	2#0000_0000_0000_0000_0000_0000_0010_0000	Hardware: no current
nAlarmLimitLow	2#0000_0000_0000_0000_0000_0001_0000_0000	Software: fallen below first lower relative temperature.
nAlarmLimitLow Low	2#0000_0000_0000_0000_0000_0010_0000_0000	Software: fallen below second lower relative temperature.
nAlarmLimitHigh	2#0000_0000_0000_0000_0000_0100_0000_0000	Software: first upper relative temperature exceeded.
nAlarmLimitHigh High	2#0000_0000_0000_0000_0000_1000_0000_0000	Software: second upper relative temperature exceeded.
nAlarmAbsolute High	2#0000_0000_0000_0000_0001_0000_0000_0000	Software: upper absolute temperature exceeded.
nAlarmAbsolute Low	2#0000_0000_0000_0000_0010_0000_0000_0000	Software: fallen below lower absolute temperature.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5.1.5 FB_CTRL_TempController_DistComp



This temperature controller function block adds disturbance compensation to the **FB_CTRL_TempController** function blocks. The structure is described here.

VAR_INPUT

```
VAR_INPUT
  eCtrlMode                       : E_CTRL_MODE;
  bSelSetpoint                    : BOOL;
  fW1                             : LREAL;
  fW2                             : LREAL;
  fX                              : LREAL;
  fYManual                        : LREAL;
  bOpenThermocouple               : BOOL;
  bReverseThermocouple            : BOOL;
  bBackVoltage                    : BOOL;
  bLeakage                        : BOOL;
  bShortCircuit                   : BOOL;
  bOpenCircuit                    : BOOL;
  fD                              : LREAL;
  bCompensateDisturbance          : BOOL;
  stParaControllerExternal        : ST_CTRL_ParaController;
END_VAR
```

Name	Unit	Area	Description
eControlMode	Obsolete	E_CTRL_MODE	Switches mode.
bSelSetpoint	Obsolete	[True, False]	Selects one of the two possible setpoints; TRUE selects the standby setpoint.
fW1	°C	LREAL	Setpoint
fW2	°C	LREAL	Standby setpoint (normally less than fW1, bSelSetpoint is used to switch between fW1 and fW2).
fX	°C	LREAL	Actual value
fYManual	%	[-100%, +100%]	Control value in manual mode
bOpenThermocouple	Obsolete	[True, False]	The thermocouple is open when TRUE; must be specified by the hardware.
bReverseThermocouple	Obsolete	[True, False]	The thermocouple is connected with incorrect polarity if TRUE; must be specified by the hardware.
bBackVoltage	Obsolete	[True, False]	The input voltage at the thermocouple is too high if TRUE; must be specified by the hardware
bLeakage	Obsolete	[True, False]	Leakage current was detected if TRUE; must be specified by the hardware.
bShortCircuit	Obsolete	[True; False]	Short circuit was detected if TRUE; must be specified by the hardware.
bOpenCircuit	Obsolete	[True, False]	Open circuit was detected if TRUE; must be specified by the hardware.
fD	Obsolete	LREAL	Actual value of the measured disturbance variable
bCompensateDisturbance	Obsolete	[True, False]	Disturbance compensation is enabled if TRUE.
sParaControllerExternal	Obsolete	Structure	An external controller parameter set was transferred to the controller.

VAR_OUTPUT

```
VAR_OUTPUT
  fYAnalog                      : LREAL;
  bYPWMPos                      : BOOL;
  bYPWMPos                      : BOOL;
  bYPWMNeg                      : BOOL;
  bYDigPos                      : BOOL;
  bYDigNeg                      : BOOL;
  dwAlarm                       : DWORD;
  fMaxOverShoot                 : LREAL;
  tStartUpTime                  : TIME;
  eCtrlState                    : E_CTRL_STATE;
  sParaControllerInternal       : ST_CTRL_ParaController;
  bError                          : BOOL;
  eErrorId                        : E_CTRL_ErrorCodes;
END_VAR
```

VAR_IN_OUT

```
VAR_IN_OUT
  sControllerParameter            : ST_CTRL_TempCtrlParameter;
  sCompensatorParameter           : ST_CTRL_DistCompParameter;
END_VAR
```

Name	Type	Description
sController parameter	ST_CTRL_TempCtrl Parameter	Parameter structure of the function block
sCompensatorP arameter	ST_CTRL_DistComp Parameter	Parameter structure of the function block

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

5.1.6 Structure Definitions (ST_CTRL_DistCompParameter)

ST_CTRL_DistCompParameter

```
TYPE ST_CTRL_DistCompParameters
STRUCT
  fKd     : LREAL := 0;
  tT1     : TIME  := T#0MS;
  tT2     : TIME  := T#0MS;
END_STRUCT
END_TYPE
```

Name	Unit	Range	Description
fKd	NA	LREAL	Propportional gain of the Lead-Lag compensator
tT1	Time	TIME	First time constant of the Lead-Lag compensator
tT2	Time	TIME	Second time constant of the Lead-Lag compensator

5.2 Global Constants

5.2.1 Library version

All libraries have a characteristic version. This version can also be seen in the PLC library repository.
A global constant contains information on the library version:

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_TempController : ST_LibVersion;
END_VAR
```

The function **F_CmpLibVersion** (defined in the library Tc2_System) is offered to compare the existing version with a required version.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

6 Sample

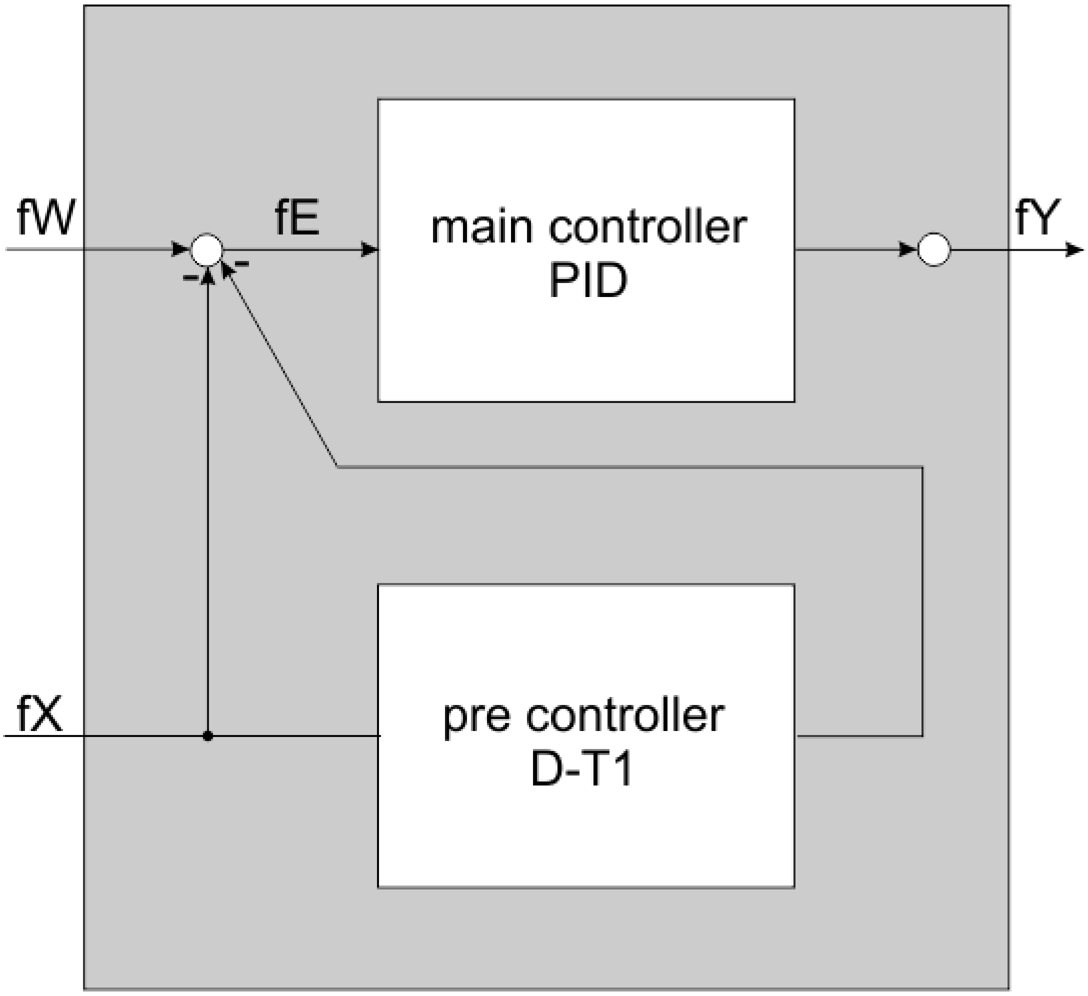
The sample program includes the integration into a MAIN program. The controlled system is simulated by a PT2 element. A recording of the values can be made with the TwinCAT Scope View.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

7 Appendix

7.1 Control Algorithm

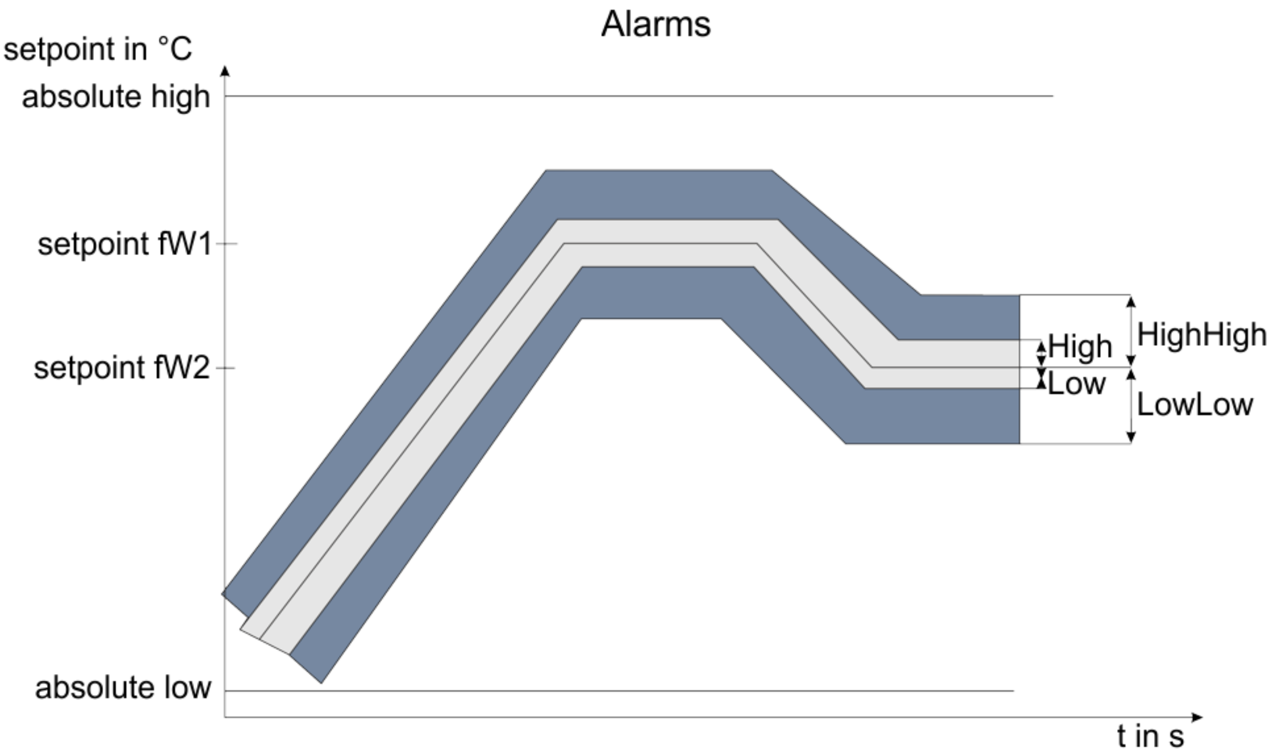


The heart of the TwinCAT Temperature Controller is a standard PID controller. This controller kernel also supports anti-reset windup measures to limit the I-component if the control value is subjected to limiting. Since the controller has been designed to minimize disturbances using the adjusting procedure according to Chien, Hrones and Reswick, overshoot is possible when the set point is changed. In order to reduce such overshoot, a pre-controller can be inserted to handle changes in the set point. The pre-controller has a D-T1 characteristic, and reduces ringing in the controller as a whole. Since the D component of the pre-controller has the effect of "roughening" the control value, the use of a pre-controller must be considered very carefully. The pre-controller is switched off when the actual value enters within a certain range of the setpoint and remains there for some length of time. The pre-controller is switched off by ramping it down over a considerable period of time. To minimize oscillation of the control value, it is optionally possible to follow the main controller with a filter. P-T1 and moving average filters are available for this purpose.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

7.2 Alarm



The following alarm conditions are continuously monitored by the temperature controller:

- Absolute temperatures (high and low)
- Relative temperatures (in two bands around the setpoint)

The following hardware conditions related to the sensor can also be linked to the temperature controller:

- Open thermocouple: broken wire to the temperature sensor.
- Back voltage: a voltage outside the permitted range is present at the temperature sensor.
- Reverse thermocouple: temperature sensor is connected with the wrong polarity.

If a current sensor is connected, then the following signals can be linked to the temperature controller:

- Short circuit
- open circuit
- Leakage current

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

7.3 Self-tuning

The self-tuning algorithm is based on the classic inflectional tangents method. This method was first developed by Ziegler and Nichols. It is assumed that a linear P-T1 loop with a dead time is being examined. The maximum rate of change is determined following an experimental step. This is achieved through examining the differences over a number of samples. A tangent is constructed to the point where the rate of change is a maximum, and its intersection with the time axis is found. The delay time, Tu, is the time from the start of the measurement up to the intersection of the inflection tangent and the time axis. Knowing Tu and Vmax, the Chien, Hrones and Reswick formula yields the controller parameters for suppression of disturbances with 20% overshoot. The parameters for the pre-controller can easily be derived from the parameters for the main controller with the aid of heuristic formulae. After completion of the self-tuning these parameters are used in an automatic switch to closed loop operation.

Requirements

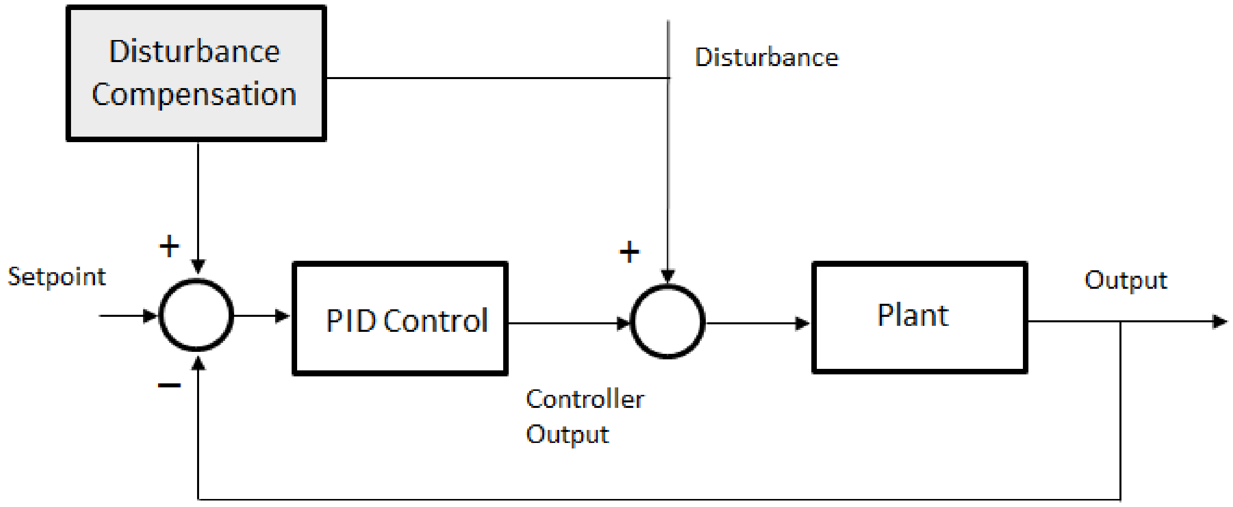
Development environment	Target platform	PLC libraries to include
TwinCAT 3.1.4016	PC or CX	Tc2_TempController

7.4 Disturbance Compensation

Disturbance compensation

A disturbance signal has a considerable influence on the quality of the controller and perhaps on the controlled process. A PID controller can passively counter the effect of an interference signal by increasing the controller output. However, this is an inefficient way of compensating.

The function block `FB_CTRL_TempController_DistComp` offers an additional lead/lag compensation to actively compensate an interference signal. It is assumed that the interference signal in question is measured and fed into the function block. The following block diagram explains the structure of the disturbance compensation:



The disturbance compensation is a lead/lag compensator. A lead/lag compensator is a versatile component that can be used to set up I, D, PI, PD, and PID compensations provided the gain and time constants are carefully selected. With the help of the compensator, permanent deviations and peaks can be reduced and the dynamic behavior in case of interference can be improved. Here you can find further information about the lead/lag compensator.

More Information:
www.beckhoff.com/tf4110

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com