Manual

TC3 Temperature Controller

TwinCAT 3

Version: 1.1
Date: 2015-08-18
Order No.: TF4110

BECKHOFF

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 the applicable national standards.
It is essential that the following notes and explanations are followed when installing and commissioning these components.

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.
For that reason the documentation is not in every case checked for consistency with performance data, standards or other characteristics.
In the event that it contains technical or editorial errors, we retain the right to make alterations at any time and without warning.
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®, Safety over EtherCAT®, TwinSAFE®, XFC®and XTS® 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, DE102004044764, DE102007017835
with corresponding applications or registrations in various other countries.

The TwinCAT Technology is covered, including but not limited to the following patent applications and patents:
EP0851348, US6167425 with corresponding applications or registrations in various other countries.

EtherCAT® is 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.
 Attention	**Damage to the environment or devices** Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.
 Note	**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 following modes:

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

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

The set value is also limited to permitted minimum and maximum values, and can also have steep slopes or ramps. A bit is available in the function block interface that provides easy switching from the set value to a stand-by set value. A soft start can be parameterised to support "heater baking". This involves the set value (optionally ramped) being initially set to a low value, held constant for a certain time, then changed to the true set value (again optionally ramped up).

The actual value can be digitally filtered.

The control algorithm is PID-based. An additional pre-regulator can be inserted in order to minimise the overshoot.

The controller has a variety of parameterisable monitoring functions. There is:

- tolerance band monitoring (two different tolerance bands),
- absolute value monitoring,
- sensor monitoring (open, back voltage, reverse) and
- monitoring of the heating current (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 step response, and uses a method of inflectional tangents to determine the maximum speed and delay time of the loop. 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.

Commissioning the controller in stages [▶ 17]

Documentation of the Function Block [▶ 21] and the structures.

3 Installation

3.1 System requirements

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

Engineering enviroment

An engineering environment, which usually describes the computer used to develop the PLC application, requires following:

- TwinCAT3 XAE build 4012 or higher
- Please note that for engineering purposes, a 7-Day trial license can be (repeatedly) used, as described in our licensing [▶ 10] section

Runtime environment

A runtime environment, which describes a computer that runs PLC application, requires:

- TwinCAT3 XAR build 4012 or higher
- Licenses for TC1200 PLC and for TF 4110 Temperature controller
- Please note: For testing purposes, a 7-Day trial license may be used, as described in our licensing [▶ 10] article

Engineering and runtime environment on the same computer

Engineering and runtime environments on the same computer (e.g. develope the application and download it on the PLC), require following :

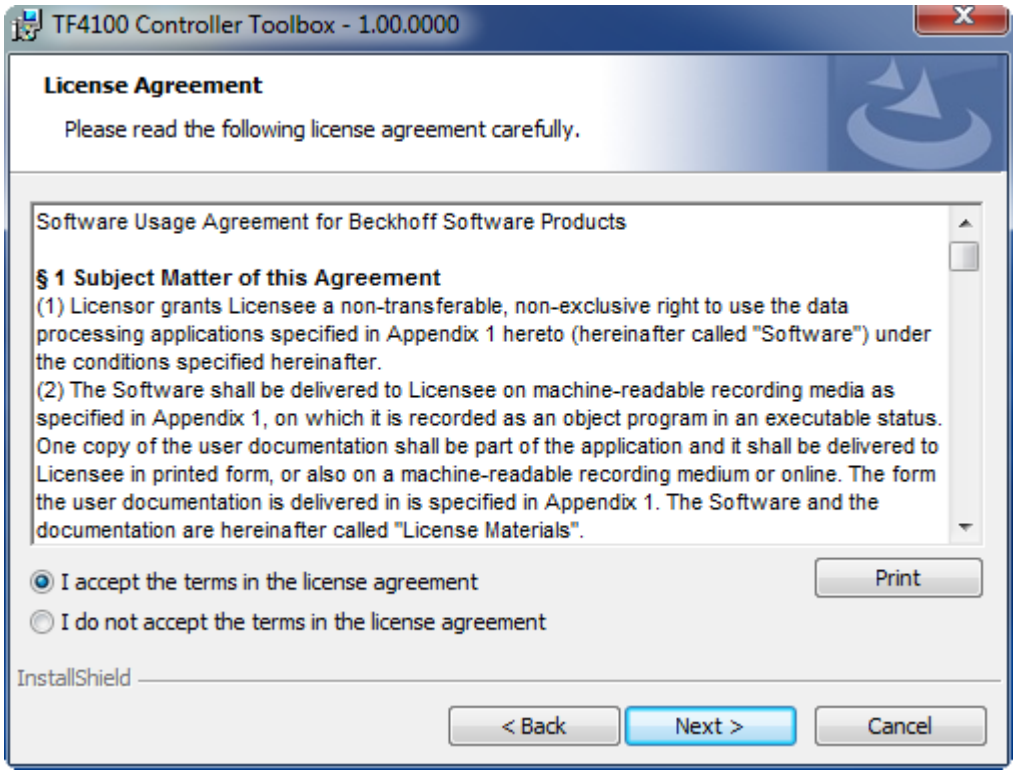
- TwinCAT3 XAE (engineering installation) build 4012 or higher
- Licenses for TC1200 PLC and for TF4110 Temperature controller
- Please note: For testing purposes, a 7-Day trial license may be used, as described in our licensing [▶ 10] article

3.2 Installation

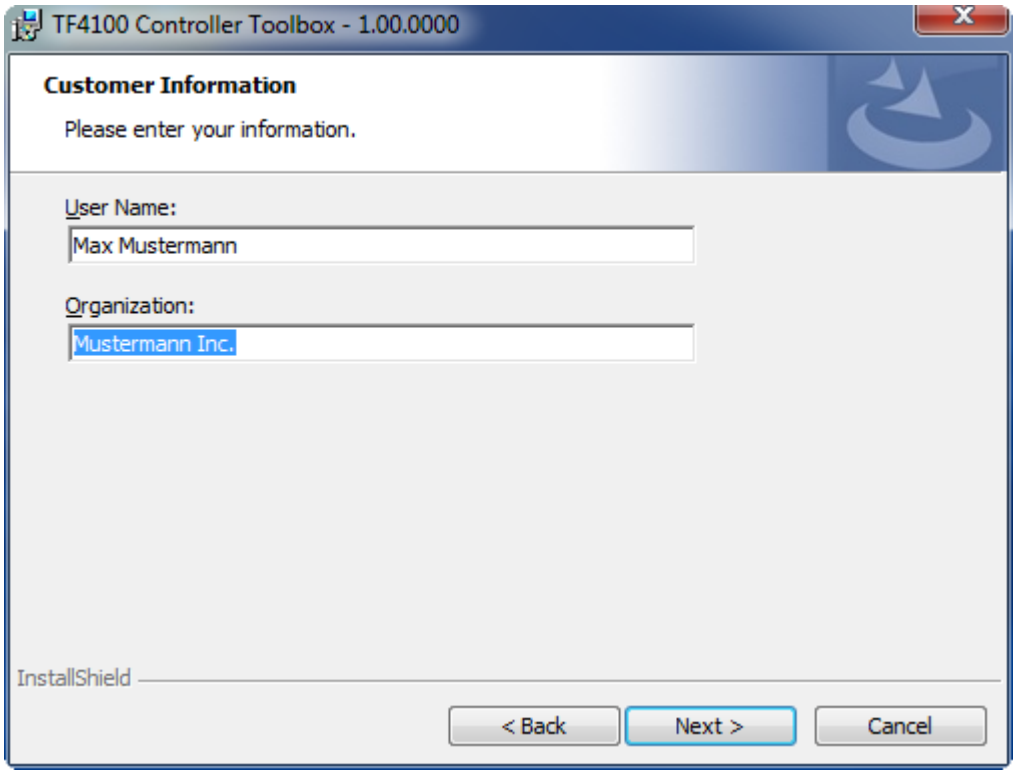
Description of the installation procedure of a TwinCAT 3 Function for Windows-based operating Systems.

1. Double-click the downloaded setup file "TFxxxx".
   Please note: Under Windows 32-bit/64-bit, please start the installation with "Run as Administrator" by right-clicking the setup file and selecting the corresponding option in the context menu.

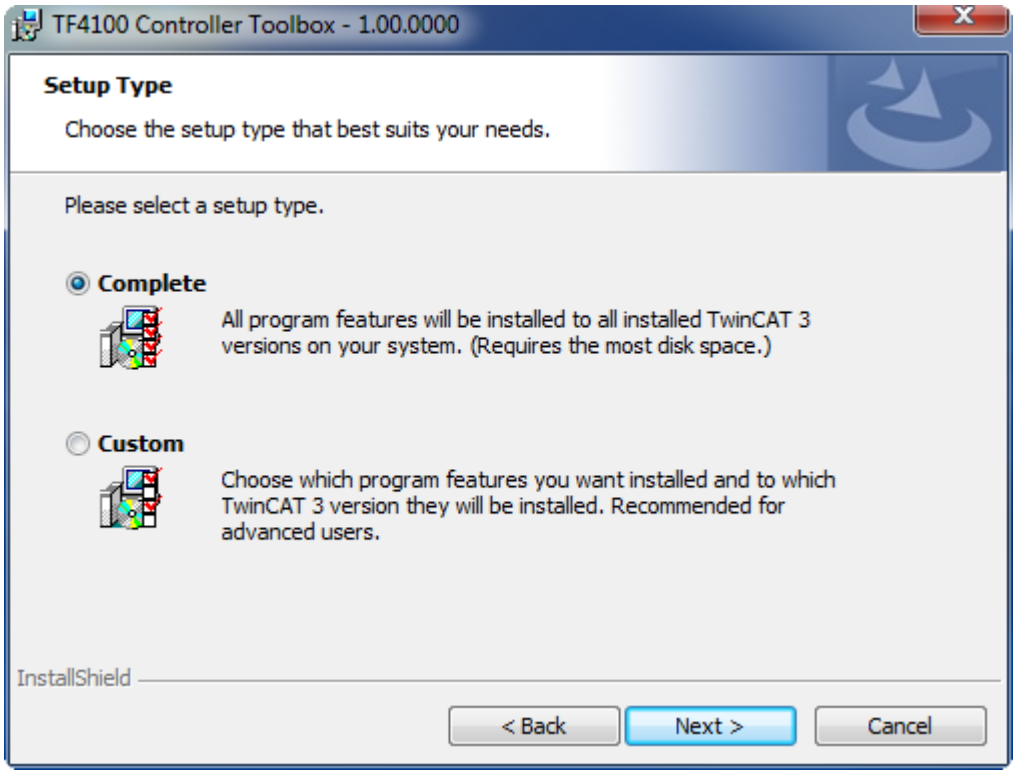
2. Click on "Next" and accept the license Agreement.



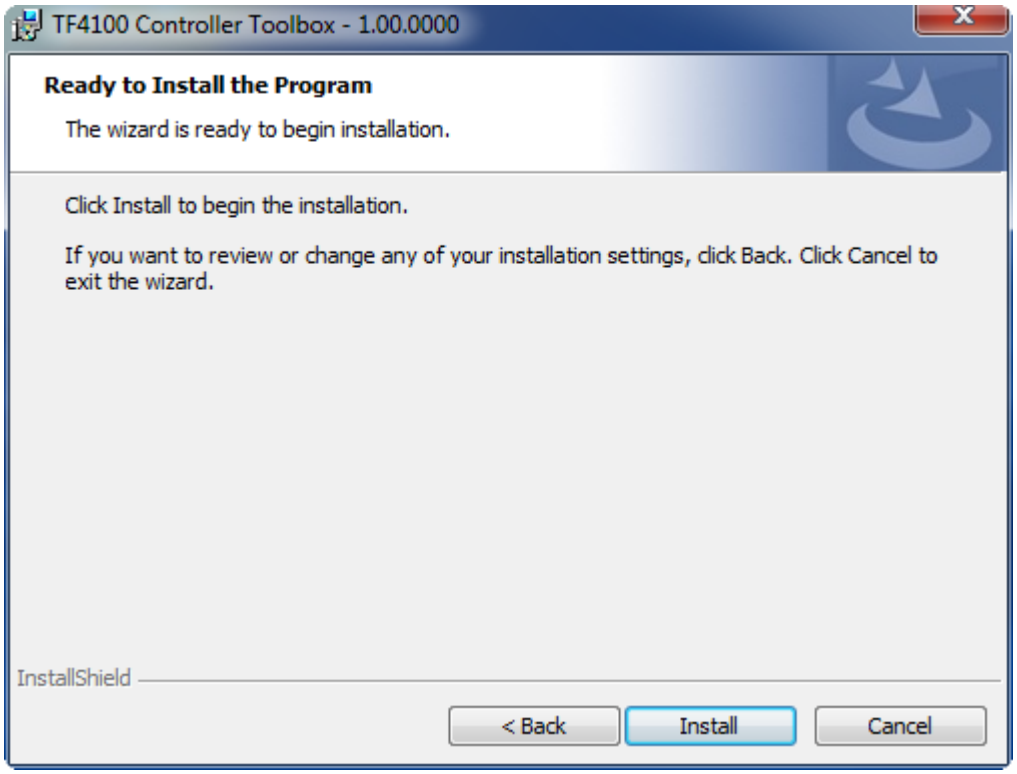
3. Enter your user information in the specified area.



4. To install the full product, including all sub-components, please choose **"Complete"** as the Setup Type.Alternatively you can also install each component seperately by choosing **"Custom"**.

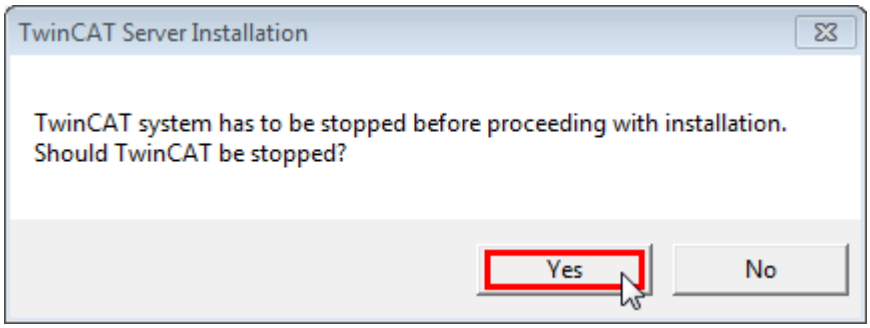


5. Click on **"Install"**after pressing the **"Next"** to start the Installation.

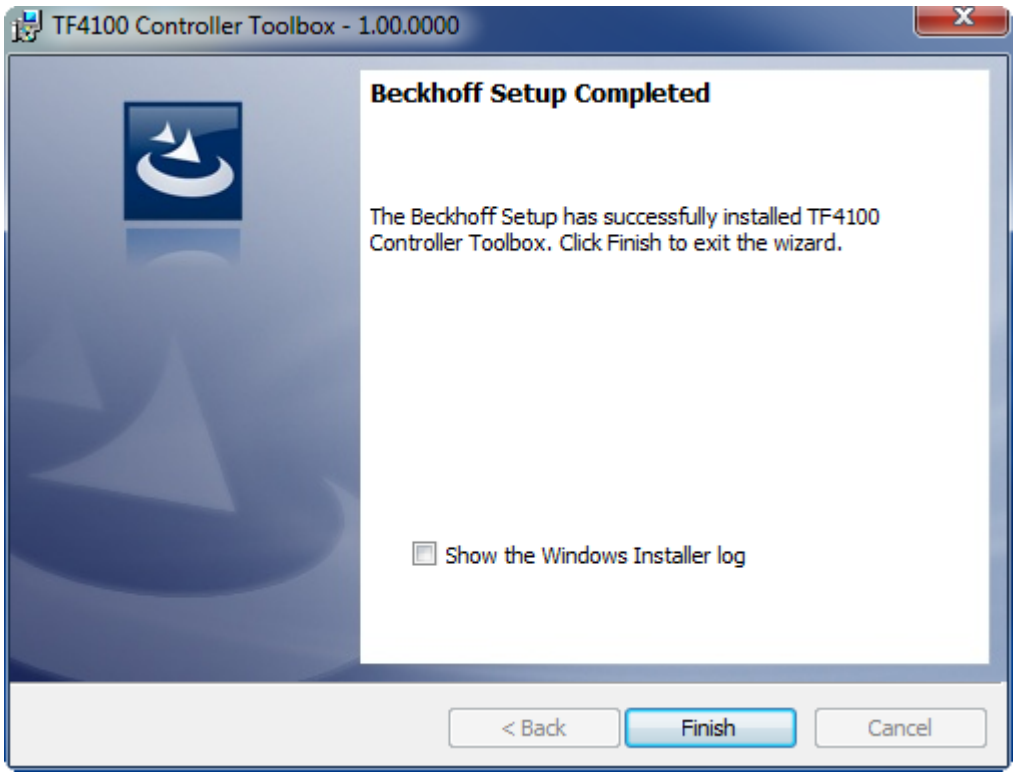


The TwinCAT system has to be stopped before proceeding with installation

6. Confirm the Dialog with **"Yes"**.



7. Select **"Finish"** to end the installation process.



⇨ The installation is complete now.

After a successful installation the TC 3Function needs to be licensed [▶ 10].

3.3 Licensing

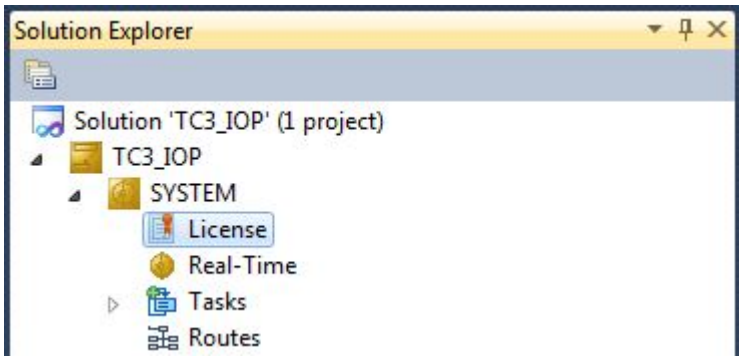
The TwinCAT 3 functions are available both as a full and as a 7-Day trial version. Both license types can be activated via TwinCAT XAE.For more information about TwinCAT 3 licensing, please consult the TwinCAT 3 Help System.The following document describes both licensing scenarios for a TwinCAT 3 function on TwinCAT 3 and is divided into the following sections:

- Licensing a 7-Day trial version [▶ 10]
- Licensing a full version [▶ 12]

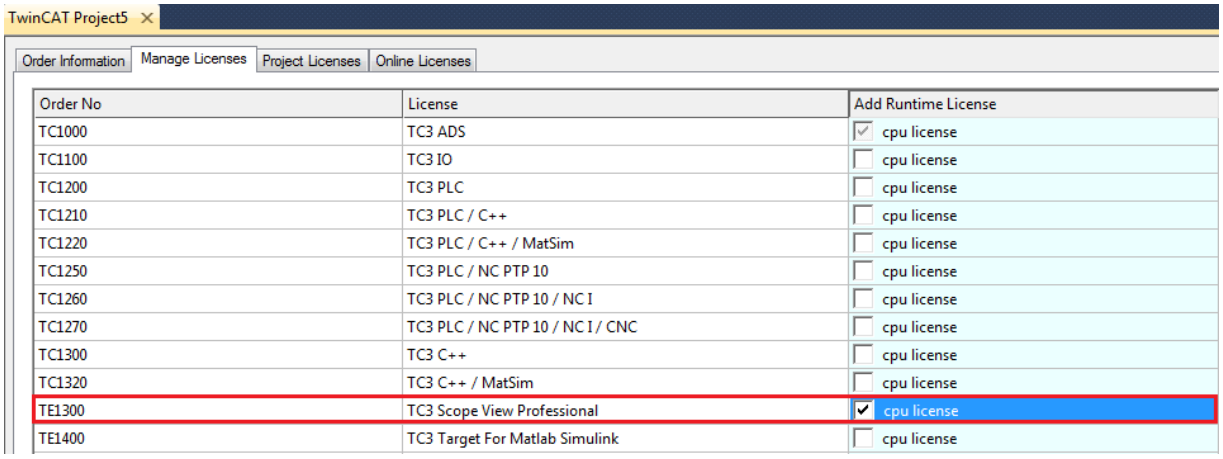
Licensing a 7-Day trial version

1. Start TwinCAT XAE
2. Open an existing TwinCAT 3 project or create a new project

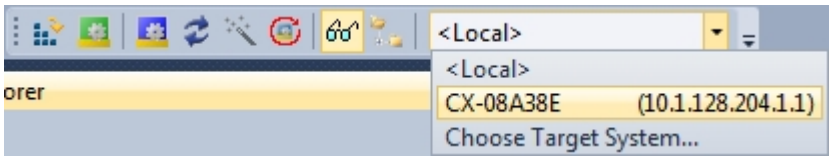
3. In “Solution Explorer”, please navigate to the entry **“System\License”**



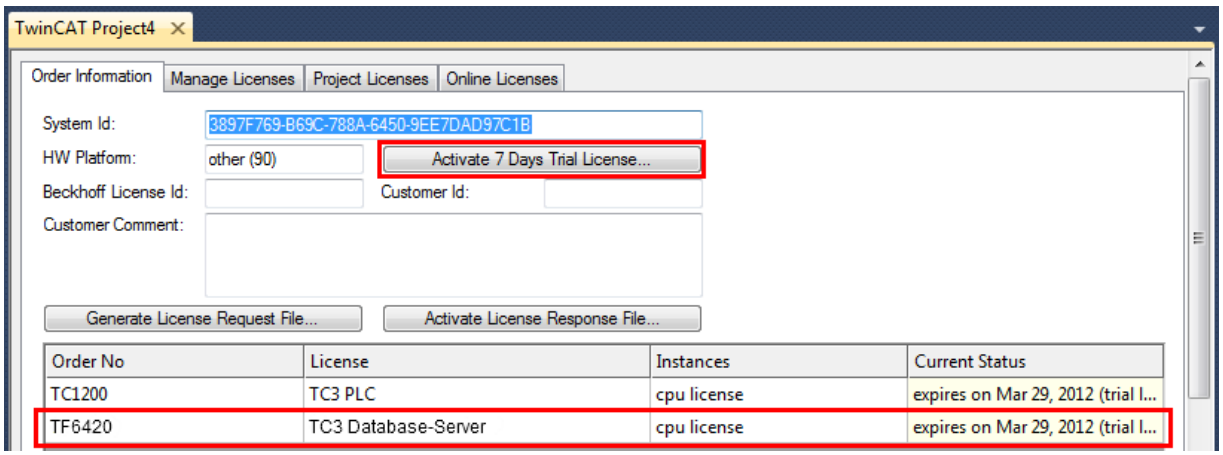
4. Open the tab **"Manage Licenses"** and add a **"Runtime License"** for your product (in this screenshot “TE1300: TC3 Scope View Professional”)



5. **Optional**: If you would like to add a license for a remote device, you first need to connect to the remote device via TwinCAT XAE toolbar



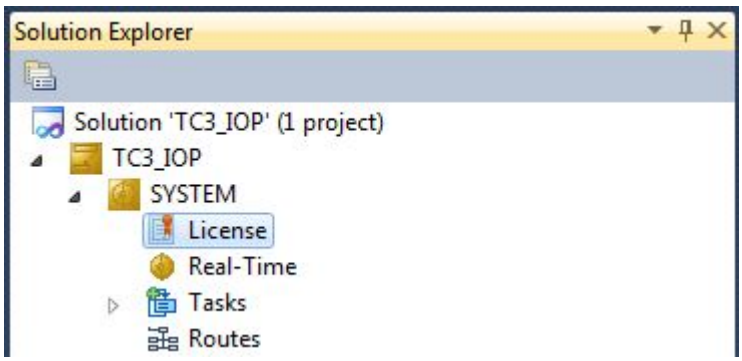
6. Switch to the tab **"Order Information"** and click the button **"Activate 7 Days Trial License..."**to activate a test version



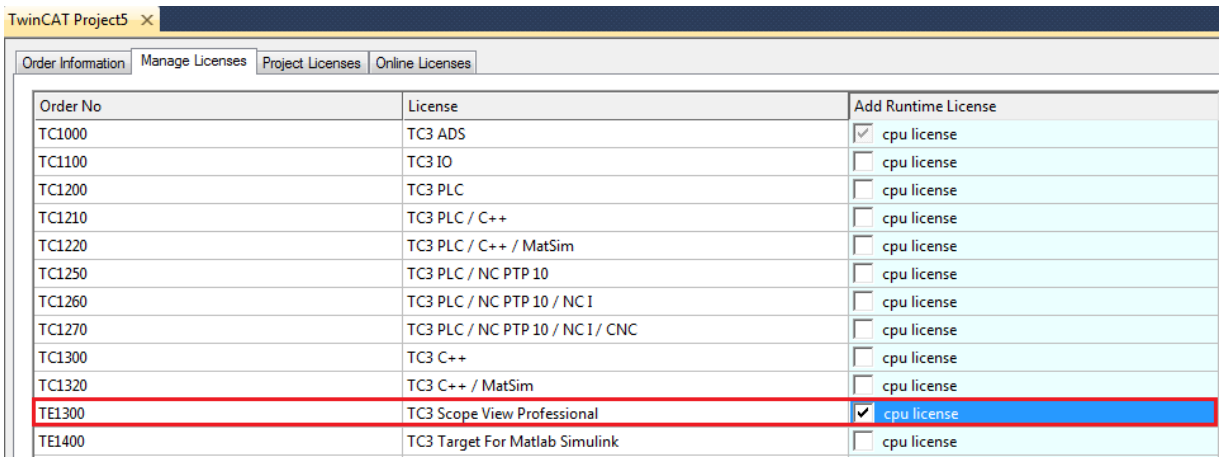
7. Please restart TwinCAT 3 afterwards.

Licensing a full version

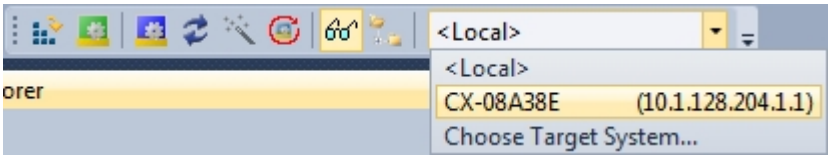
8. Start TwinCAT XAE
9. Open an existing TwinCAT 3 project or create a new project
10. In "Solution Explorer", please navigate to the entry **"SYSTEM\License"**



11. Open the tab **"Manage Licenses"** and add a **"Runtime License"** for your product (in this screenshot " TE1300: TC3 Scope View Professional").

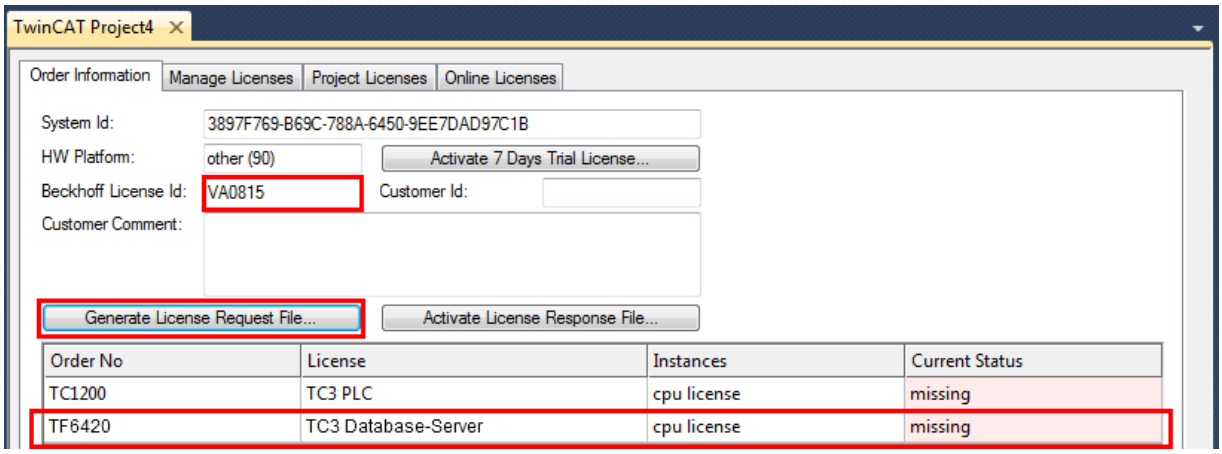


12. **Optional:** If you would like to add a license for a remote device, you first need to connect to the remote device via TwinCAT XAE toolbar

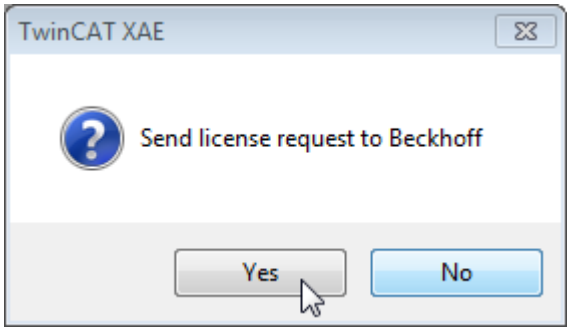


13. Navigate to the **"Order Information"** tab
The fields "System-ID" and "HW Platform" cannot be changed and just describe the platform for the licensing process in general a TwinCAT 3 license is always bound to these two identifiers:
the "System-ID" uniquely identifies your system.
The "HW Platform" is an indicator for the performance of the device.

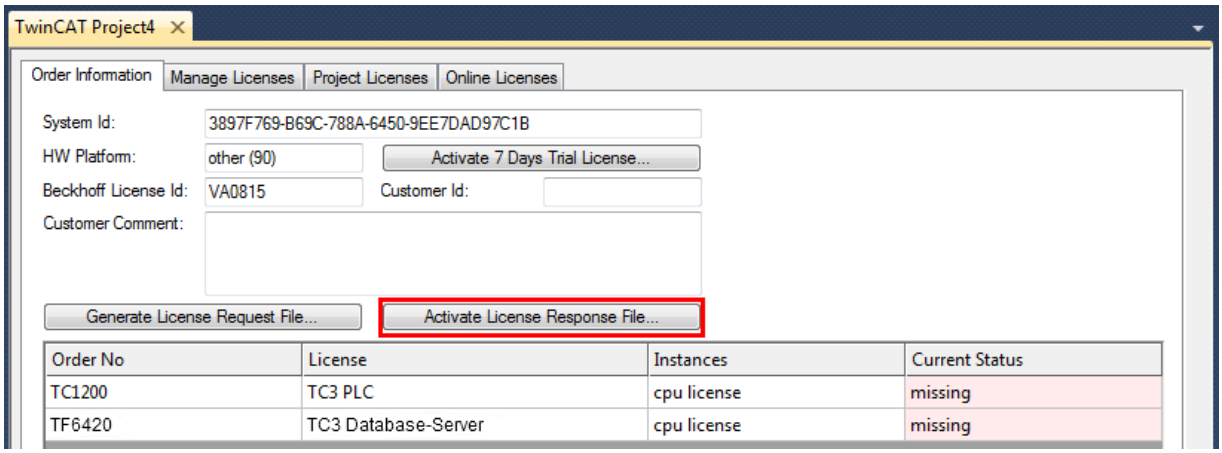
14. Optionally, you may also enter an own order number and description for your convenience



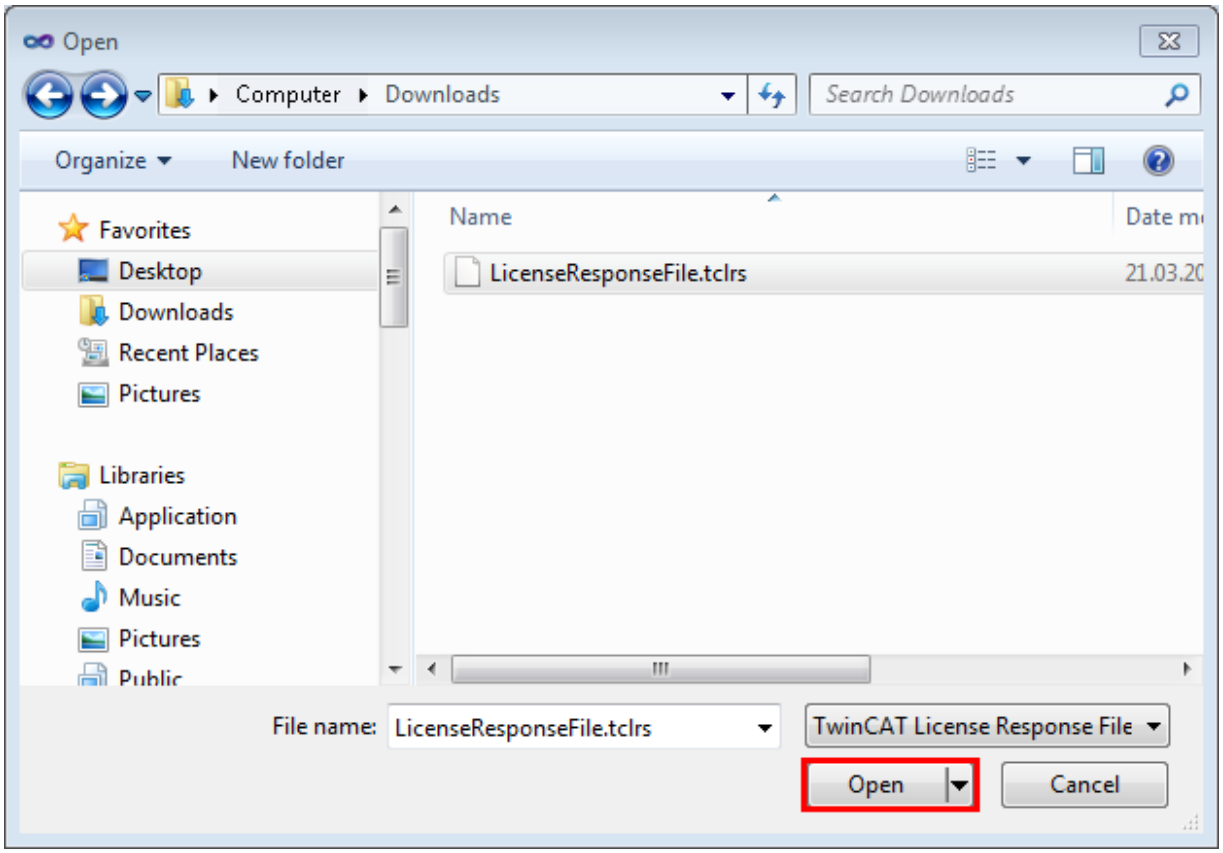
15. enter the "Beckhoff License ID" and click on **"Generate License Request File..."**. If you are not aware of your **"Beckhoff License ID"** please contact your local sales representative.
16. After the license request file has been saved, the system asks whether to send this file via E-Mail to the Beckhoff Activation Server



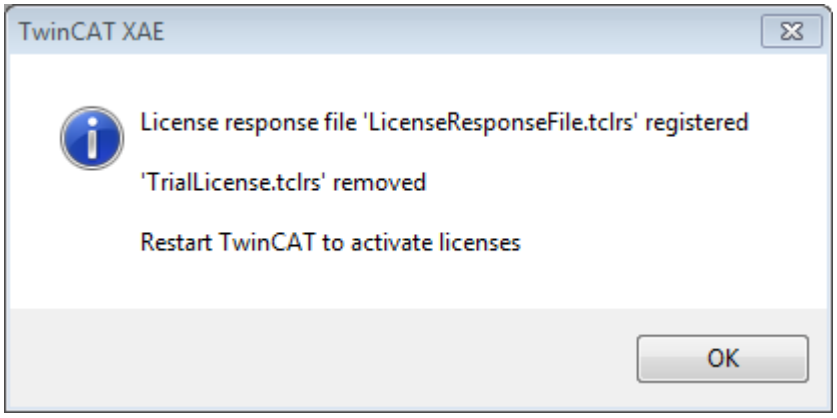
17. After clicking "Yes", the standard E-Mail client opens and creates a new E-Mail message to "tclicense@beckhoff.com" which contains the "License Request File"
18. Send this Activation Request to Beckhoff
    NOTE! The "License Response File" will be sent to the same E-Mail address used for sending out the "License Request File"
19. After receiving the activation file, please click on the button "Activate License Response File..."in the TwinCAT XAE license Interface.



20. Select the received "Licnse response file" and click on "Open"



21. The "License Response File" will be imported and all included licenses will be activated. If there have been any trial licenses, these will be removed accordingly.
22. Please restart TwinCAT to activate licenses..



NOTE! The license file will be automatically copied to "..\TwinCAT\3.1\Target\License" on the local device.

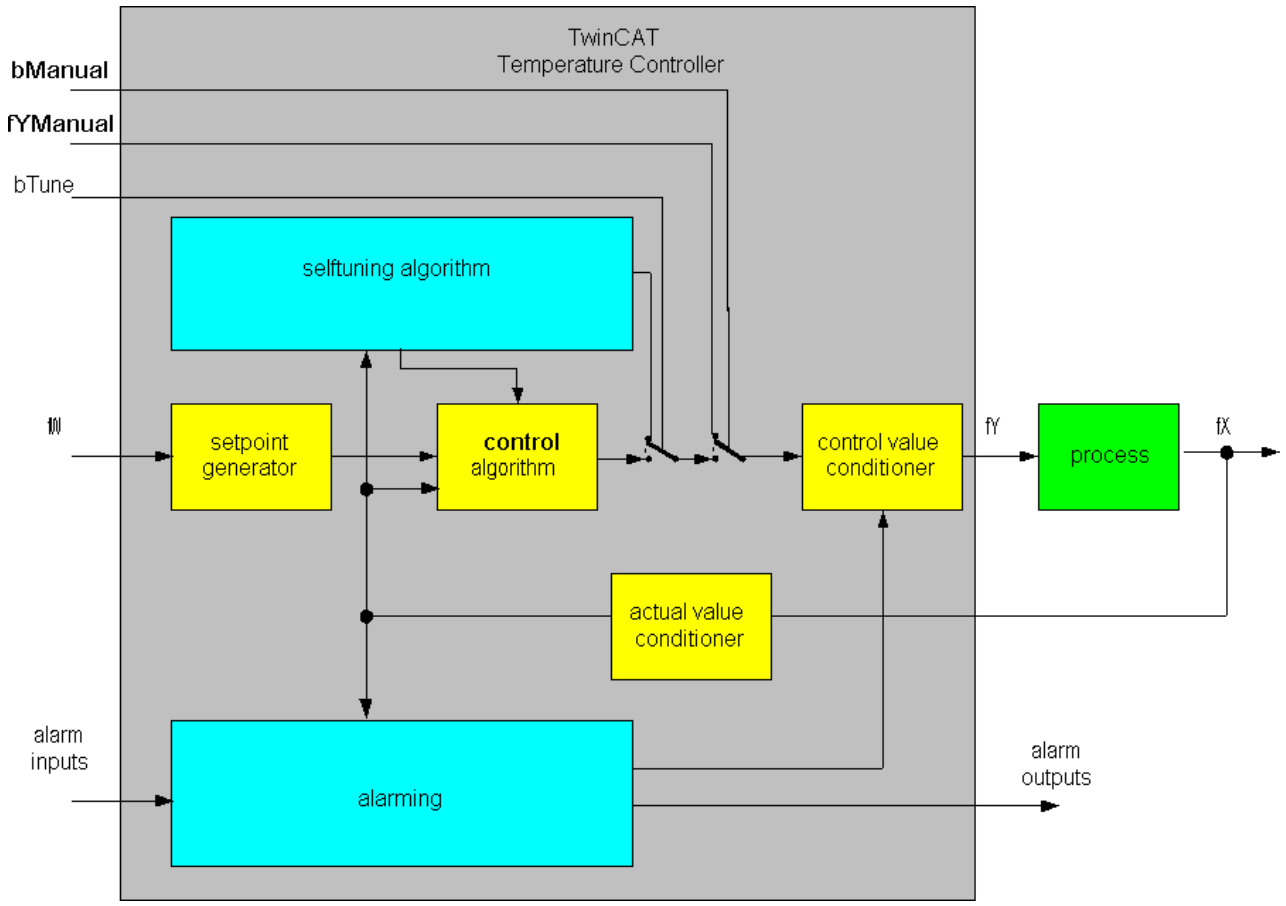
4 Configuration

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)
- Set value generator (FB_SetpointConditioner)
- Control value generator (FB_ControlValueConditioner)
- Alarming (FB_Alarming)

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

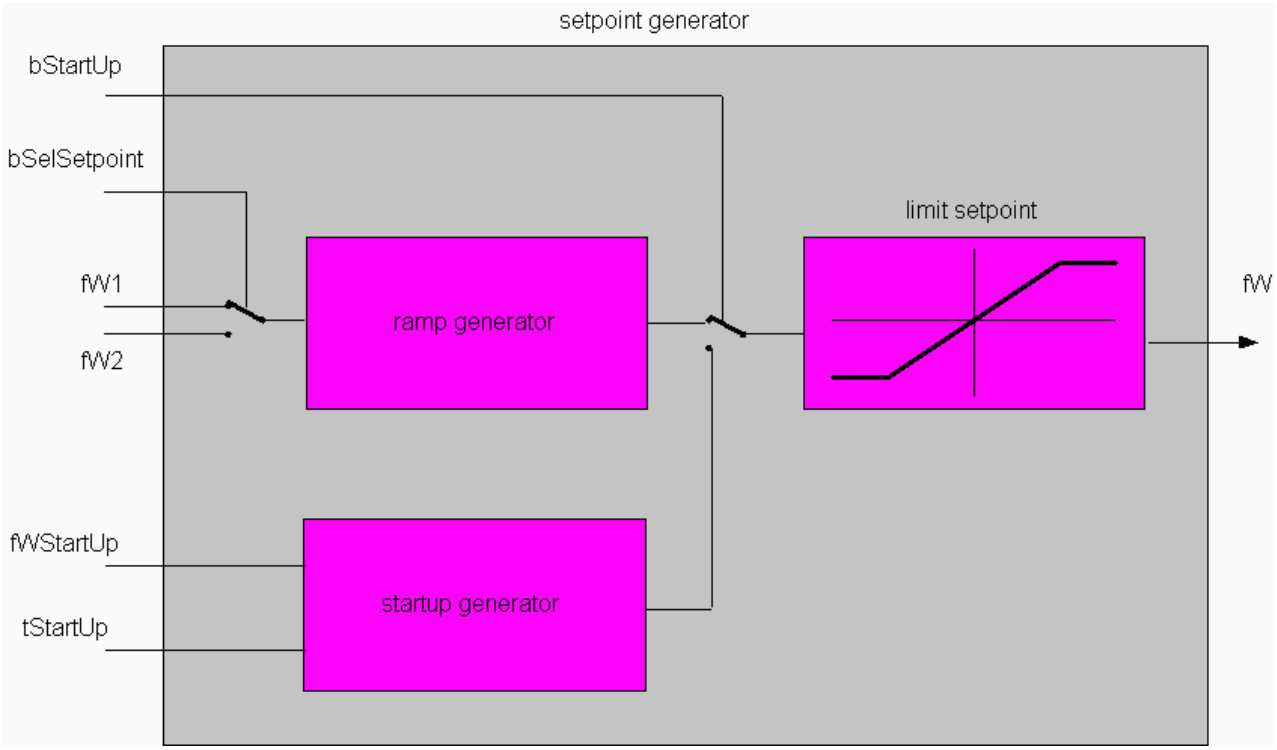


The diagram illustrates the individual function blocks.

4.2 Generating the Set Value

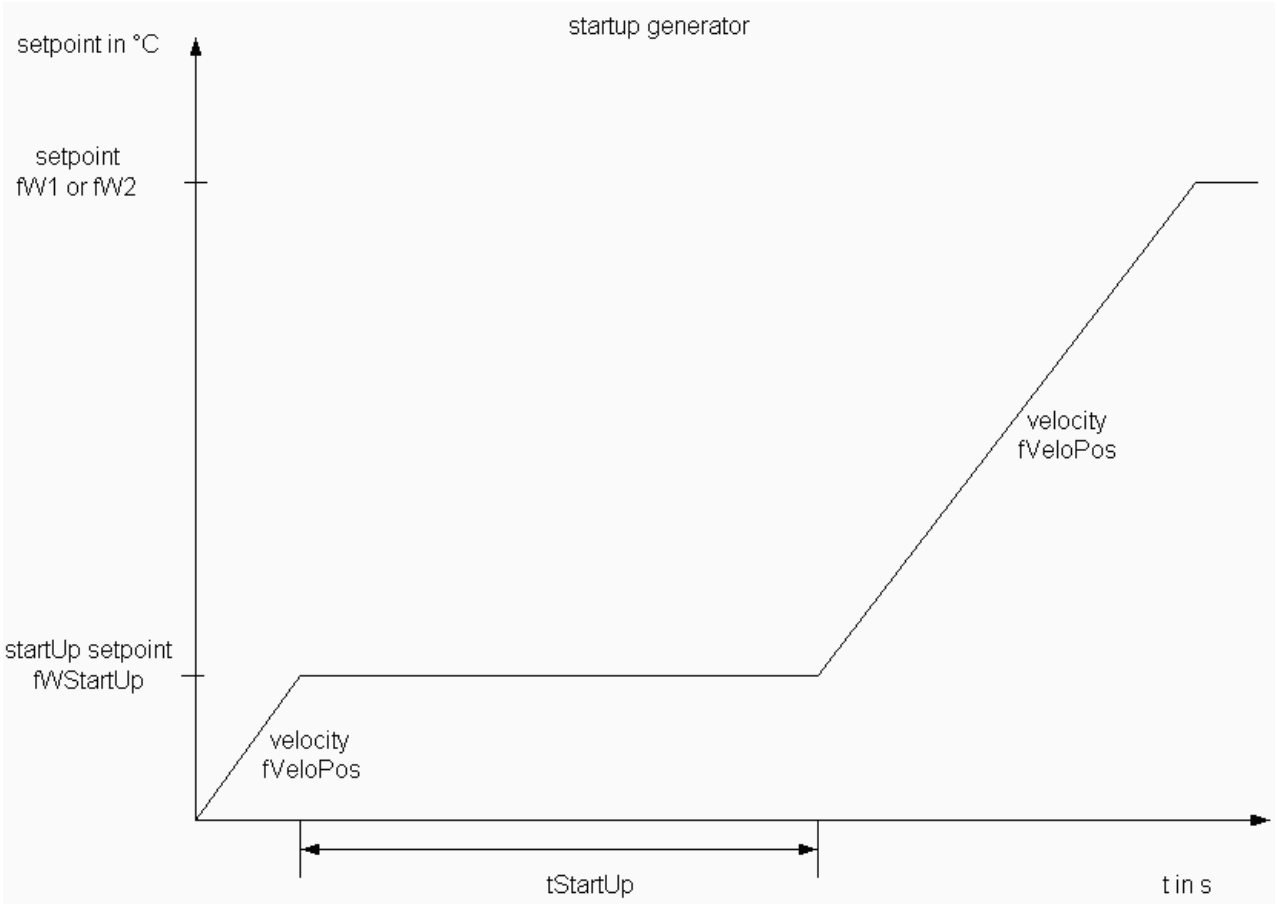
One bit switches between the set values. In addition to the actual set value, there is also a standby set value. The standby set value can be used to reduce the temperature during operating pauses to a lower value in order to save power. If necessary the steps in the set value can be ramped. The parameter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

The set values are restricted to their limits.



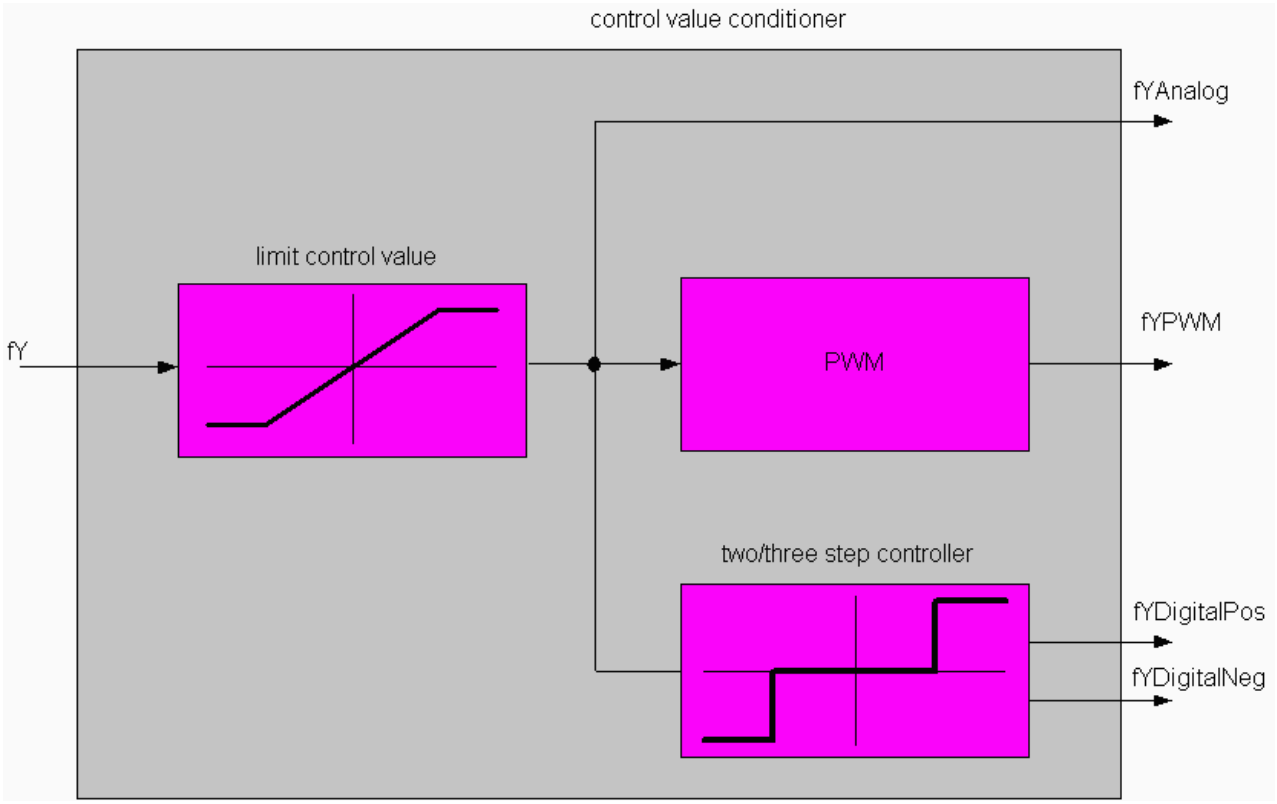
In order to permit "heater baking", a soft start can be parameterised. In this case, the temperature is first ramped up from ambient to a low set value (fWStartUp). This temperature is them maintained for a period of time (tStartUp), and only after that has elapsed does the ramp up to the actual set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

Start-up



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 block in the control value structure. The control value is made available in three different ways. The control value can be picked up in analogue form. However it is more usual for the digital output to take the form of 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:

- **The controller library must be added to the project using the library manager.**

TcTempCtrl.lib is to be added in the library manager.

- **At least one instance of the controller must be programmed.**

An instance of the FB_TempController block is to be created. It is also necessary for an instance of tge ST_Controllerparameter structure to be created.

- **Perform the required external connection.**

Name		Beschreibung
eCtrlMode	Connection necessary	Switches controller to different operating modes (active, passive, tuning)
bSelSetpoint	Connection optional	Selects one of two possible set values. FALSE selects the normal set value, while TRUE selects the standby set value.
fW1	Connection necessary	Set value.
fW2	Connection optional	Standby set value, generally smaller than fW1. fSelSetpoint can be used to switch between fW1 and fW2.
fX	Connection necessary	Actual value. This value must be converted to LREAL.
fYManual	Connection optional	Control value in manual mode.
bOpenThermocouple	Connection optional	The thermocouple is open if TRUE. Must be indicated by the hardware (e.g. KLxxxx).
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 rate etc.) are passed to the block in this structure.
sParaControllerExternal	Connection optional	An external controller parameter set is passed to the block in this structure.

- **Perform the necessary parameterisation of the controller via the structure.**

The parameters can be specified through initial values, or by assignment. If initial values are used, then the instance of the structure with initial values looks like this:

```
(* 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
);
```

The marked parameters are optional, and only need to be initialised if they are needed.

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

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

- **Specification of the controller sampling time, that task cycle time and the PWM cycle time**

The controller's sampling time must be adapted to the particular process. 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 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.

- **Parameterisation of TwinCAT Scope**

To check the results, a scope examination should always be made of the tuning process and of the closed loop regulation behaviour. To do this, TwinCAT Scope View should be started and parameterised. The following channels should be recorded: set value (fW1 or fW2), actual value (fX) and the analog control value (fYAnalog).

- **Switching off the Alarms during the Commissioning Phase**

The alarms can be temporarily switched off during the commissioning phase. An appropriate bit mask must be written into the dwAlarmSupp data word. If a bit is set in this data word, the corresponding alarm is disabled. The assignment of the individual alarms is described here [▶ 33].

Warning: All the required alarms should be switched on again after the initial commissioning!

- **Starting the Controller with Tuning**

If the controller parameters are to be determined with the aid of the Tuning system, the bOn and bTune inputs must both be TRUE. A fixed waiting period of 20s first elapses. During this waiting period 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 magnitude of fYTune. The process then reacts with the step response. As long as 80% of the set value is not reached, the process parameters are determined using the inflectional tangent method. For safety reasons, after 80% of the set value 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.

Attention	**[Step size definition!]** [A step of at least 40°C must be used for the purposes of tuning. Smaller steps can result in incorrect parameter determination!]

Attention	**[Set Control Mode!]** [When tuning is finished successful eCtrlState is set to eCTRL_STATE_TUNED. Controller waits for new command. If control mode is set eCTRL_MODE_ACTIVE the controller starts in Closed-Loop-mode with the new parameter set from tuning.]

- **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, the sParaControllerInternal structure is fed back to the controller's sParaControllerExternal input, and the bSelCtrlParameterSet flag set to TRUE.

- **Fine tuning**

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 should be considered only as a guide.

Behaviour	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 3.1.4006	PC or CX (x86)	Tc2_TempController

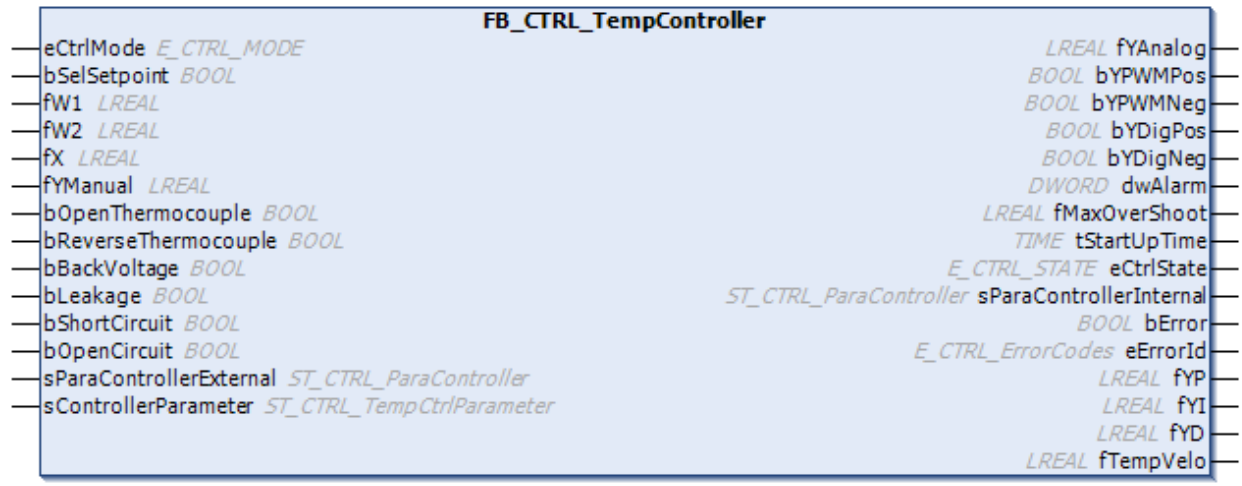
5 PLC libraries

5.1 Function Block

5.1.1 FB_CTRL_TempController

The temperature controller block FB_CTRL_TempController has a variety of inputs and outputs that are described below. All the controller's parameters are passed to it via structures. The structures and enums are defined here [▶ 33].

Function Block




```
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

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

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
```

Interface

Table 1: Inputs

Name	Unit	Range	Description
eControlMode	1	E_CTRL_MODE	Switches modes.
bSelSetpoint	1	[TRUE,FALSE]	Selects one of two possible set values. FALSE selects the normal set value, while TRUE selects the standby set value.
fW1	°C	LREAL	Setpoint
fW2	°C	LREAL	Standby set value, 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.
fYManual	-100% - +100%	LREAL	Control value in manual mode.
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 rate etc.) are passed to the block in this structure.
sParaControllerExternal	none	Structure	An external controller parameter set is passed to the block in this structure.

Table 2: Outputs

Name	Unit	Range	Description
fYAnalog	none	LREAL	Analogue control value.
bYPWMPos	none	[TRUE,FALSE]	Boolean output, pulse width modulated (positive, heating)
bYPWMNeg	none	[TRUE,FALSE]	Boolean output, pulse width modulated (negative, cooling)
bYDigPos	none	[TRUE,FALSE]	Boolean output of a three-point controller (TRUE control value 100%, FALSE control value off)
bYDigNeg	none	[TRUE,FALSE]	Boolean output of a three-point controller (TRUE control value -100%, FALSE control value off)
dwAlarm	none	DWORD	Alarm signals (see ENUM ...)
fMaxOverShoot	°C	LREAL	max. overshoot in °C under/over setpoint
tStartUpTime	TIME	-	rising time until the first time reaching the setpoint.
eCtrlState	none	E_CTRL_STATE	actuel state of the controller (s. ENUM ...)
sParaControllerInternal	none	Structure	PID controller parameter from tuning.
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 ...)

Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	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
```

Table 3: Description

Name	Unit	Range	Description
General Parameters			
iMode	none	INT	Controller operating mode (1 = heating, 2 = cooling, 3 = heating & cooling) (see below)
iReactionOnFailure	none	INT	Parameterisable reaction to errors (see below)
bSelCtrlParameterSet	none	BOOL	TRUE = external parameter set, FALSE = internal parameter set (from 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	°C	E_CTRL_TuneMode	Selects how tuning should be done (see below)
tTuneStabilisation	s	TIME	Waiting time (stabilisation of process) until step for tuning is switched on.
fEndTunePercentHeating	%	(L)REAL	percent of setpoint when controller stops tuning and waits for switching to closed loop control
fYTuneCooling	°C	(L)REAL	Control value for tuning step in cooling mode.
fYStableCooling	°C	(L)REAL	Control value for stabilizise process before cooling tuning starts.
fScalingFactor	none	(L)REAL	Scaling factor: if only heating tuning was done cooling parameters are heating parameter*scaling factor
setpoint parameters			
fWMin	°C	(L)REAL	Minimum value of the control variable.
fWMax	°C	(L)REAL	Maximum value of the control variable.
bEnableSoftStart	none	BOOL	FALSE = no soft start, TRUE = soft start
bEnableRamping	none	BOOL	FALSE = no ramping, TRUE = ramping
fWStartUp	°C	(L)REAL	Setpoint while soft start phase
tStartUp	s	TIME	Waiting time in soft start phase
bStartUpRamping	none	[TRUE,FALSE]	Enables ramping in soft start phase
fWStartUpVeloPos	°C/s	(L)REAL	Rate of rise (of ramp) during the start-up phase.
fWStartUpVeloNeg	°C/s	(L)REAL	Rate of fall (of ramp) during the start-up phase.
fWVeloPos	°C/s	(L)REAL	Rate of rise (of ramp).
fWVeloNeg	°C/s	(L)REAL	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	°C	(L)REAL	Size of deadband in °C
control value parameters			
fYMin	none	(L)REAL	Minimum value of the control variable.
fYMax	none	(L)REAL	Maximum value of the control variable.
fYManual	none	(L)REAL	Control value in manual operation.
fYOnFailure	none	(L)REAL	Control value in case of error (parameterisable).

Name	Unit	Range	Description
tPWMCycleTime	s	TIME	Cycle time of the PWM signal.
tPWMMinOffTime	s	TIME	PWM: minimal off time
tPWMMinOnTime	s	TIME	PWM: minimal on time
tPWMWaitingTime	s	TIME	PWM: waiting time before switching vorm heating to cooling and vice versa
fYThresholdOff	%	(L)REAL	3-Step: Off threshold
fYThresholdOn	%	(L)REAL	3-Step: On threshold
nCyclesForSwitchOver	none	INT	Number of cycles in which a change of parameter set is done (smooth switching)
controller Parameters			
bEnablePreController	none	[TRUE,FALSE]	Switches pre-controller on
bEnableZones	none	[TRUE,FALSE]	Switches open loop characteristic on until close to set value.
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	°C	(L)REAL	Relative lower temperature limit in the first band.
TempLowLow	°C	(L)REAL	Relative lower temperature limit in the second band.
TempHigh	°C	(L)REAL	Relative upper temperature limit in the first band.
TempHighHigh	°C	(L)REAL	Relative upper temperature limit in the second band.
TempAbsoluteHigh	°C	(L)REAL	Absolute upper temperature limit.
TempAbsoluteLow	°C	(L)REAL	Absolute lower temperature limit.
for experts only parameters			
fTuneKp	none	(L)REAL	FineTuning parameter for PID controller
fTuneTn	none	(L)REAL	FineTuning parameter for PID controller
fTuneTv	none	(L)REAL	FineTuning parameter for PID controller
fTuneTd	none	(L)REAL	FineTuning parameter for PID controller

Description

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
```

Table 4: Description

Name	Unit	Range	Description
KpHeat	none	(L)REAL	Heating: Amplification factor for the main controller.
TnHeat	s	TIME	Heating: Integral-action time for main controller (I component).
TvHeat	s	TIME	Heating: Derivative action time for main controller (D component).
TdHeat	s	TIME	Heating: Damping time for the main controller.
KpCool	none	(L)REAL	Cooling: Amplification factor for the main controller.
TnCool	s	TIME	Cooling: Integral-action time for main controller (I component).
TvCool	s	TIME	Cooling: Derivative action time for main controller (D component).
TdCool	s	TIME	Cooling: Damping time for the main controller.

ENUM: E_CTRL_ERRORCODES

s. Documentation of TcControllerToolbox

Table 5: 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 there is an error (an alarm) set the control value to the average control value (not yet implemented)

Table 6: ENUM: E_CTRL_ControllerStateInternal

Name	Description
E_CTRL_ControllerStateInternalHeating	internal
E_CTRL_ControllerStateInternalCooling	internal

Table 7: 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

s. Documentation of TcControllerToolbox.

Table 8: ENUM: E_CTRL_STATE_TUNING

Name	Description
eCTRL_STATE_TUNING_INIT	Tuning: Initialisation
eCTRL_STATE_TUNING_IDLE	Tuning: wainting for a stable actual value
eCTRL_STATE_TUNING_PULSE	Tuning: not yet realized
eCTRL_STATE_TUNING_STEP	Tuning: Tuning with step reponse
eCTRL_STATE_TUNING_READY	Tuning: Calculation of parameters
eCTRL_STATE_TUNING_ERROR	Tuning: Error while tuning.

Table 9: ENUM: E_CTRL_TuneMode

Name	Description
eCTRL_TuneMode_HEATING	Tuning: only heating
eCTRL_TuneMode_COOLING	Tuning: only cooling
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 tuning with a defined oscillation (in planning)

Table 10: ENUM: E_CTRL_FilterType

Name	Description
eCTRL_FilterType_FIRSTORDER	first order filter
eCTRL_FilterType_AVERAGE	moving average filter

Table 11: ENUM: E_CTRL_ControllerType

Name	Description
eCTRL_ControllerType_PID	Standard-PID controller
eCTRL_ControllerType_PI	Standard-PI controller
eCTRL_ControllerType_PID_Pre	Standard-PID controller with pre-controller(in preparation).
eCTRL_ControllerType_PIDD2	Special PID controller (in preparation)

Bit-Masken für Alarme

Name	Mask	Description
nAlarmOpenThermocouple	2#0000_0000_0000_0000_0000_0000_0000_0001	Hardware: open temperature sensor
nAlarmReverseThermocouple	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
nAlarmLeakageCurrent	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

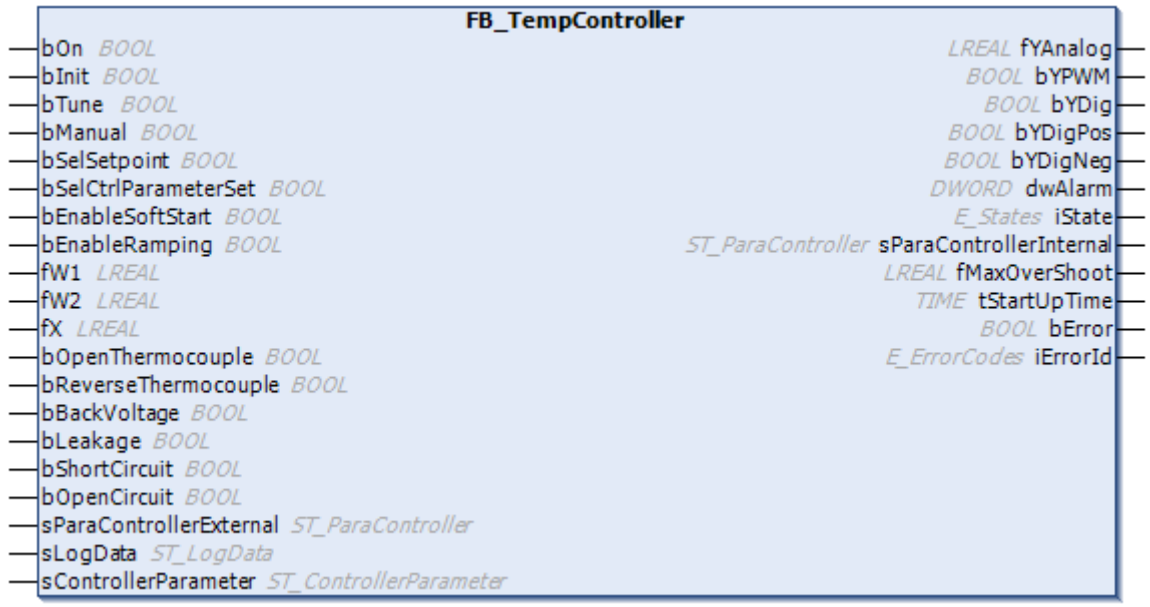
Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	Tc2_TempController

5.1.3 old:FB_TempController

The temperature controller 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 structures and enums are defined here [▶ 33].

Function Block



Interface

```
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

VAR_IN_OUT
  sControllerParameter      : ST_ControllerParameter;
END_VAR

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
```

Table 12: Inputs

Name	Unit	Value range	Description
bOn	1	[TRUE,FALSE]	TRUE switches the controller on.
bInit	1	[TRUE,FALSE]	Initialisation 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 set values. FALSE selects the normal set value, while TRUE selects the standby set value.
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 jump in the set value to be converted to a ramp.
fW1	°C	LREAL	Set value.
fW2	°C	LREAL	Standby set value, 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 rate etc.) are passed to the block in this structure.
sParaControllerExternal	none	Structure	An external controller parameter set is passed to the block in this structure.
sLogData	none	Structure	This structure passes parameters for logging to the block (filenames etc.).

Table 13: Outputs

Name	Unit	Value range	Description
fYAnalog	none	LREAL	Analogue control value.
bYPWM	none	[TRUE,FALSE]	Boolean output, pulse width modulated.
bYDig	none	[TRUE,FALSE]	Boolean output of a two-point controller (TRUE control value 100%, FALSE control value off)
bYDigPos	none	[TRUE,FALSE]	Boolean output of a three-point controller (TRUE control value 100%, FALSE control value off)
bYDigNeg	none	[TRUE,FALSE]	Boolean output of a three-point controller (TRUE control value -100%, FALSE control value off)
dwAlarm	none	DWORD	Alarm signals (see ENUM ...)
iState	none	INT	Present controller status (see ENUM ...)
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 ...)

Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	Tc2_TempController

5.1.4 old:Structure Definitions

```
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

Table 14: Description

Name	Unit	Value range	Description
iMode	none	INT	Controller operating mode (1 = heating, 2 = cooling, 3 = heating & cooling) (see below)
iReactionOnFailure	none	INT	Parameterisable 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	°C	LREAL	Minimum set value.
fWMax	°C	LREAL	Maximum set value.
fWVeloPos	°C/s	LREAL	Rate of rise (of ramp).
fWVeloNeg	°C/s	LREAL	Rate of fall (of ramp).
fWStartUp	°C	LREAL	Set value at start-up.
tStartUp	s	TIME	Time with the fWStartUp set value.
bStartUpRamping	none	[TRUE,FALSE]	Switches on ramping during the start-up phase.
fWStartUpVeloPos	°C/s	LREAL	Rate of rise (of ramp) during the start-up phase.
fWStartUpVeloNeg	°C/s	LREAL	Rate of fall (of ramp) during the start-up phase.
fYMin	none	LREAL	Minimum value of the control variable.
fYMax	none	LREAL	Maximum value of the control variable.
fYManual	none	LREAL	Control value in manual operation.
fYOnFailure	none	LREAL	Control value in case of error (parameterisable).
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 set value.
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	°C	LREAL	Relative lower temperature limit in the first band.
TempLowLow	°C	LREAL	Relative lower temperature limit in the second band.
TempHigh	°C	LREAL	Relative upper temperature limit in the first band.
TempHighHigh	°C	LREAL	Relative upper temperature limit in the second band.
TempAbsoluteHigh	°C	LREAL	Absolute upper temperature limit.
TempAbsoluteLow	°C	LREAL	Absolute lower temperature limit.

Description

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
```

Table 15: Description

Name	Unit	Value range	Description
KpMain	none	LREAL	Amplification 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 16: ENUM: Errorcodes

Name	Description
TC_ERR_NOERROR	No error.
TC_ERR_INVALIDPARAM	Invalid parameter.
TC_ERR_NO_INIT	Missing block initialisation.
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 17: 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 18: ENUM: ST_ControlMode

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

Table 19: ENUM: states

Name	Description
TC_STATE_IDLE	Controller switched off.
TC_STATE_INIT	Controller is being initialised.
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 delay time.
TC_STATE_TUNE_STEP	Step for determination of delay time and maximum speed.
TC_STATE_TUNE_READY	Self-tuning complete.
TC_STATE_ERROR	Error (logical error).

Table 20: ENUM: E_FilterType

Name	Description
E_FilterType_FIRSTORDER	First order filter.
E_FilterType_AVERAGE	Mean value filter.

Table 21: 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
nAlarmOpenThermocouple	2#0000_0000_0000_0000_0000_0000_0000_0001	Hardware: open temperature sensor
nAlarmReverseThermocouple	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
nAlarmLeakageCurrent	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_0000_0100_0000	Software: fallen below first lower relative temperature
nAlarmLimitLowLow	2#0000_0000_0000_0000_0000_0000_1000_0000	Software: fallen below second lower relative temperature
nAlarmLimitHigh	2#0000_0000_0000_0000_0000_0001_0000_0000	Software: first upper relative temperature exceeded
nAlarmLimitHighHigh	2#0000_0000_0000_0000_0000_0010_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

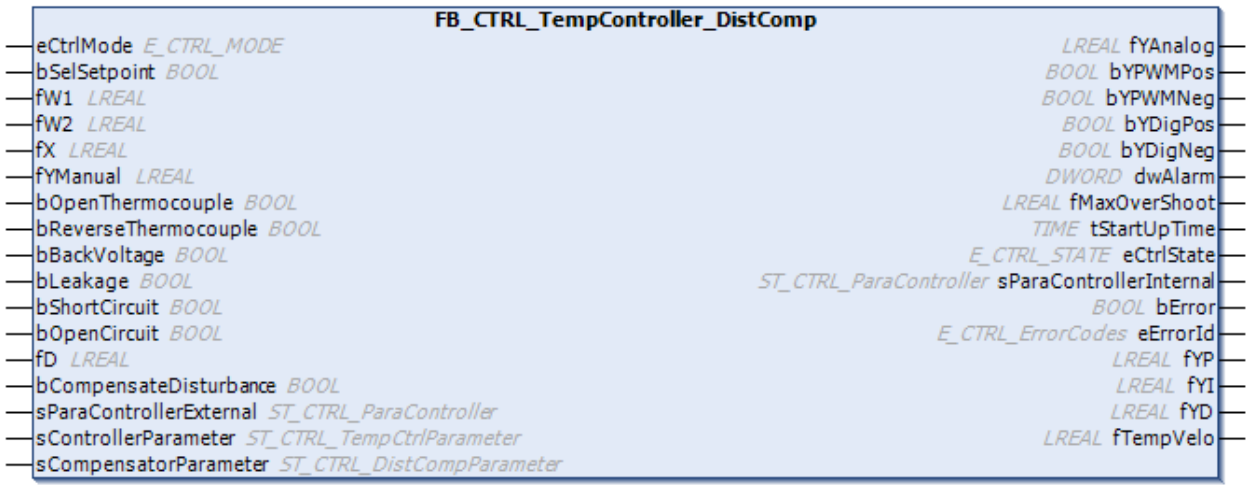
Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	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 illustrated here.

Function Block



Interface

```
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

VAR_IN_OUT
  sControllerParameter          : ST_CTRL_TempCtrlParameter;
  sCompensatorParameter         : ST_CTRL_DistCompParameter;
END_VAR

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
```

Inputs

Name	Unit	Range	Description
eControlMode	NA	E_CTRL_MODE	Switches mode
bSelSetpoint	NA	[True, False]	Selects one of the two possible setpoints; True selects the standby setpoint
fW1	°C	LREAL	Setpoint
fW2	°C	LREAL	Standby setpoint (generally smaller 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	NA	[True, False]	The Thermocouple is open if true; must be indicated by the hardware
bReverseThermocouple	NA	[True, False]	The Thermocouple is connected with wrong polarity if true; must be indicated by the hardware
bBackVoltage	NA	[True, False]	The input voltage at the Thermocouple is too high, if true; must be indicated by the hardware
bLeakage	NA	[True, False]	Leakage current has been detected, if true; must be indicated by the hardware
bShortCircuit	NA	[True; False]	Short circuit has been detected, if true; must be indicated by the hardware
bOpenCircuit	NA	[True, False]	Open circuit is detected, if true; must be indicated by the hardware
fD	NA	LREAL	Actual value of the measured disturbance
bCompensateDisturbance	NA	[True, False]	Disturbacen compensate is activated, if true
sParaControllerExternal	NA	Structrue	An external controller parameter set is passed to the controller

Outputs

Name	Unit	Range	Description
fYAnalog	NA	LREAL	Analog control value
bYPWMPos	NA	[True, False]	Digital output, Pulse width modulated
bYPWMNeg	NA	[True, False]	Digital output; Negation of bYPWMPos; Pulse width modulated
bYDigPos	NA	[True, False]	Digital output of 3-point controller; True equals +100% control value and False equals control value off
bYDigNeg	NA	[True, False]	Digital output of 3-point controller; True equals -100% control value and False equals control value off
dWAlarm	NA	DWORD	Alarm signals (see Enum)
fMaxOverShoot	°C	LREAL	Max. overshoot in degree C under/over setpoint
tStartUpTime	Time	TIME	Rising time until the actual value first hits the setpoint
eCtrlState	NA	E_CTRL_STATE	Acutal state of the controller
sParaControllerInternal	NA	Structure	PID Controller parameters after tuning
bError	NA	[True, False]	An error occursed, if true
eErrorId	NA	Structure	Error code of the current error

Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	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 specific version. This version is shown in the PLC library repository too.
A global constant contains the library version information:

Global_Version

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

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

Hint: All other possibilities known from TwinCAT2 libraries to query a library version are obsolete!

Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	Tc2_TempController

6 Sample

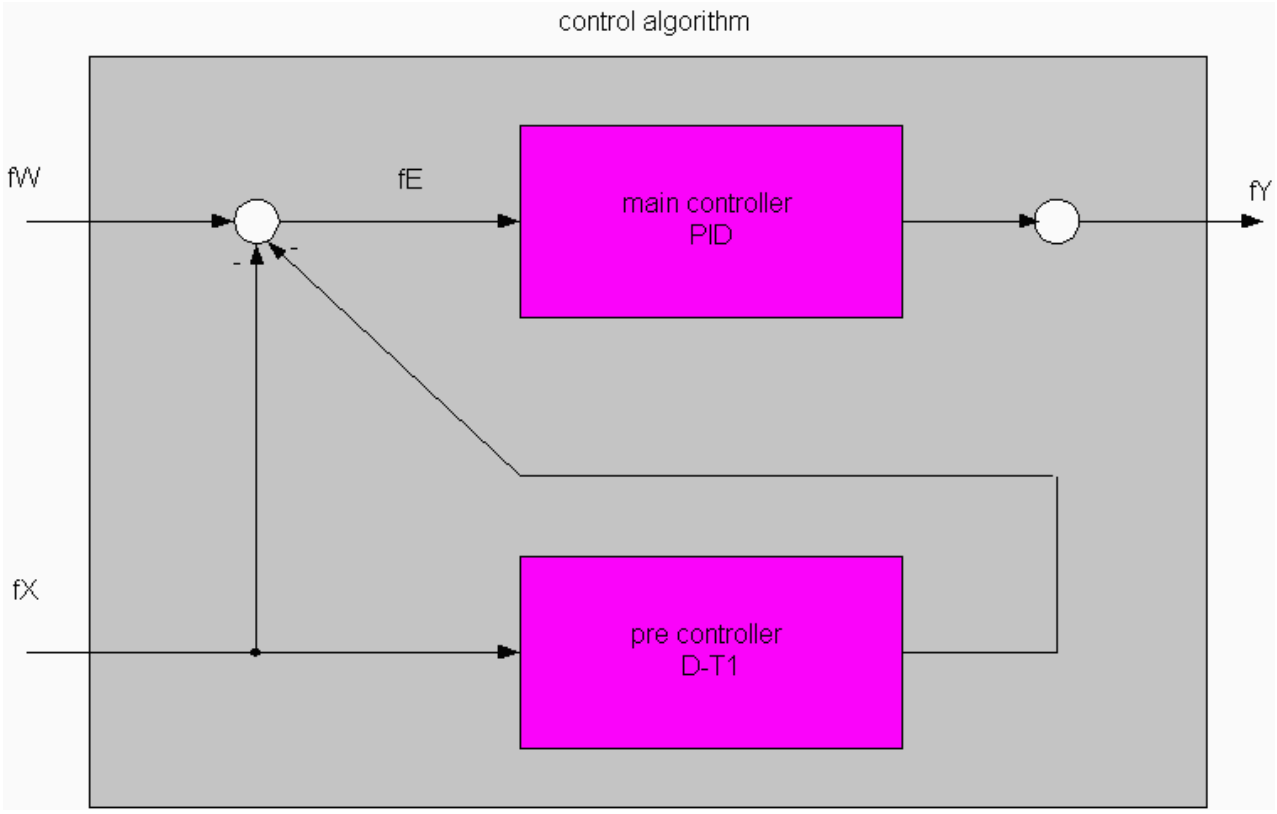
You can download TwinCAT sample code here. The sample code contains two separate programs using both old and new function blocks. For practical use, you may need one of two implementation. You can also use TwinCAT ScopeView to monitor variables such as temperature setpoint, acutal temperature or controller output.

Requirements

Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	Tc2_TempController

7 Appendix

7.1 Control Algorithm

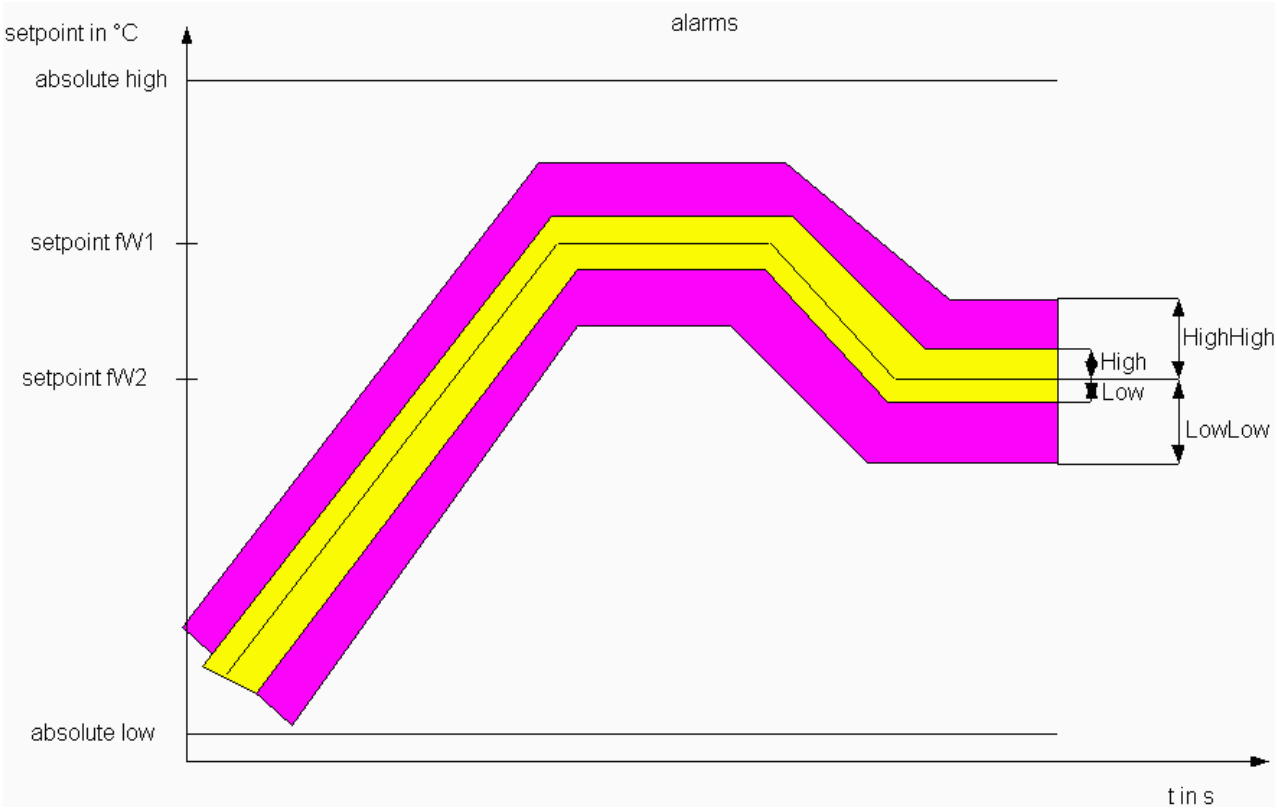


The heart of the TwinCAT Temperature Controller is a standard PID controller. This controller kernel also supports anti-reset windup measures to limit the I-component if the control value is subjected to limiting. Since the controller has been designed to minimise disturbances using the adjusting procedure 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-regulator 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 set value 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 minimise 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 3.1.4006	PC or CX (x86)	Tc2_TempController

7.2 Alarming



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

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

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 3.1.4006	PC or CX (x86)	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 delay 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

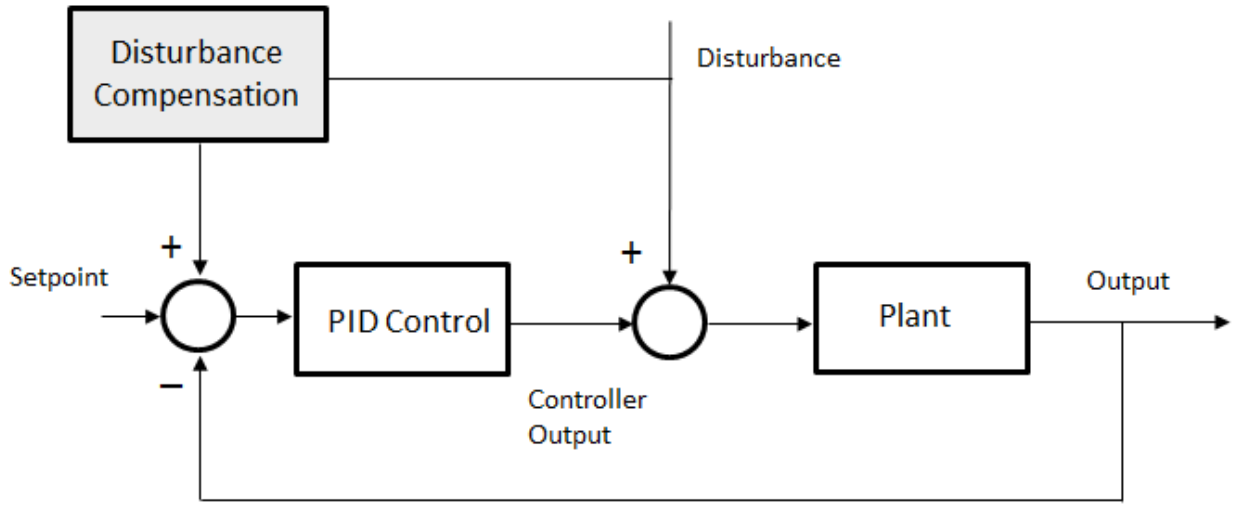
Development environment	target platform	PLC libraries to include
TwinCAT 3.1.4006	PC or CX (x86)	Tc2_TempController

7.4 Disturbance Compensation

Disturbance Compensation

A disturbance signal has significant effect on the quality of the controller and eventually on the controlled process. A PID controller can passively overcome the effect of a disturbance signal by increasing the controller output. But this is an inefficient way of handling the compensation.

The function block FB_CTRL_TempController_DistComp offers an addional Lead-Lag compensation to actively compensate the disturbance signal. It is assumed that the disturbing signal in question is measured and fed to the function block. Following block diagram explains the structure of the disturbance compensation:



The disturbance compensation is actually a Lead-Lag compensator. A Lead-Lag compensator is a versatile component, which can be used to achieve I, D, PI, PD and PID type compensators by carefully choosing its gain and time constants. The compensator is useful to reduce stead state error, peaks, and improving the dynamic response to the disturbance. More information about the Lead-Lag compensator can be found here.