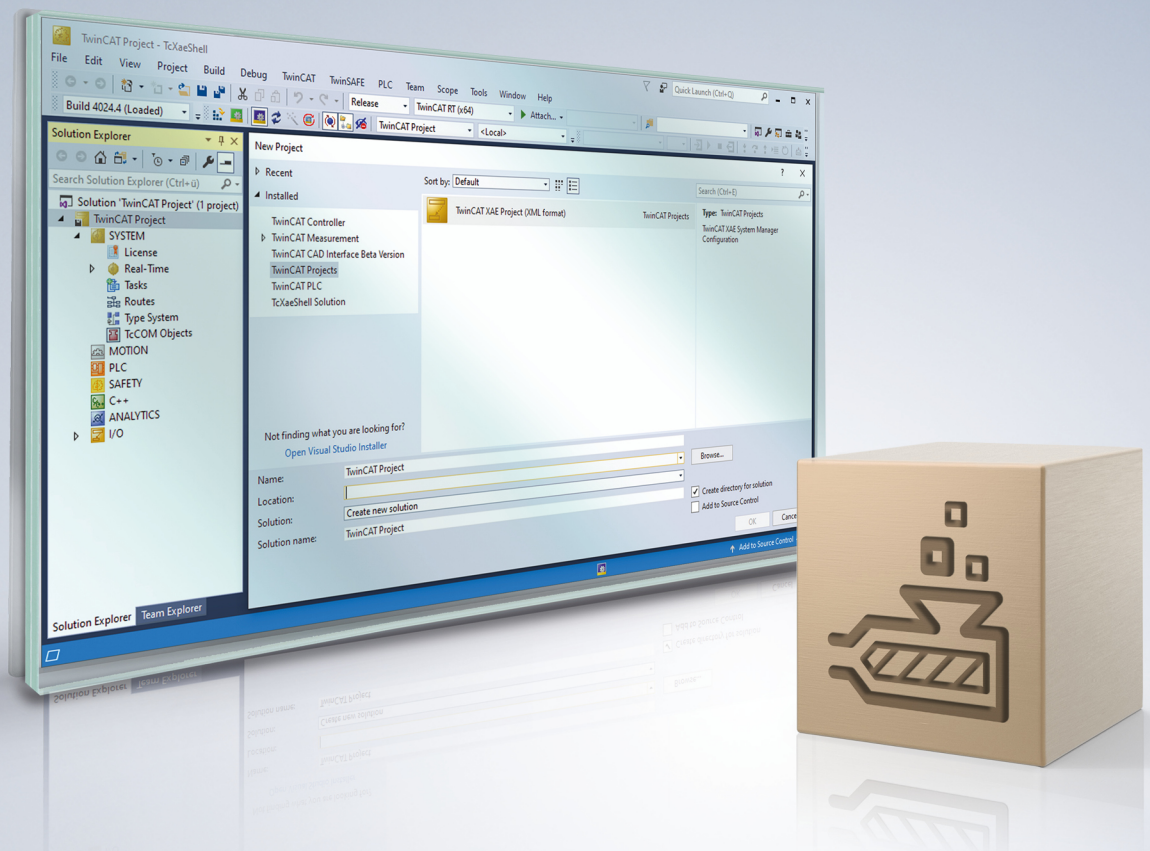
BECKHOFF New Automation Technology

Manual | EN

TF85xx

TwinCAT 3 | Plastic Application



2023-09-18 | Version: 1.0.0

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.
The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all 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 notice.
No claims to modify products that have already been supplied may be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

Patents

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

EtherCAT®

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

Copyright

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are 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 technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER

Hazard with high risk of death or serious injury.

⚠ WARNING

Hazard with medium risk of death or serious injury.

⚠ CAUTION

There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE

The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

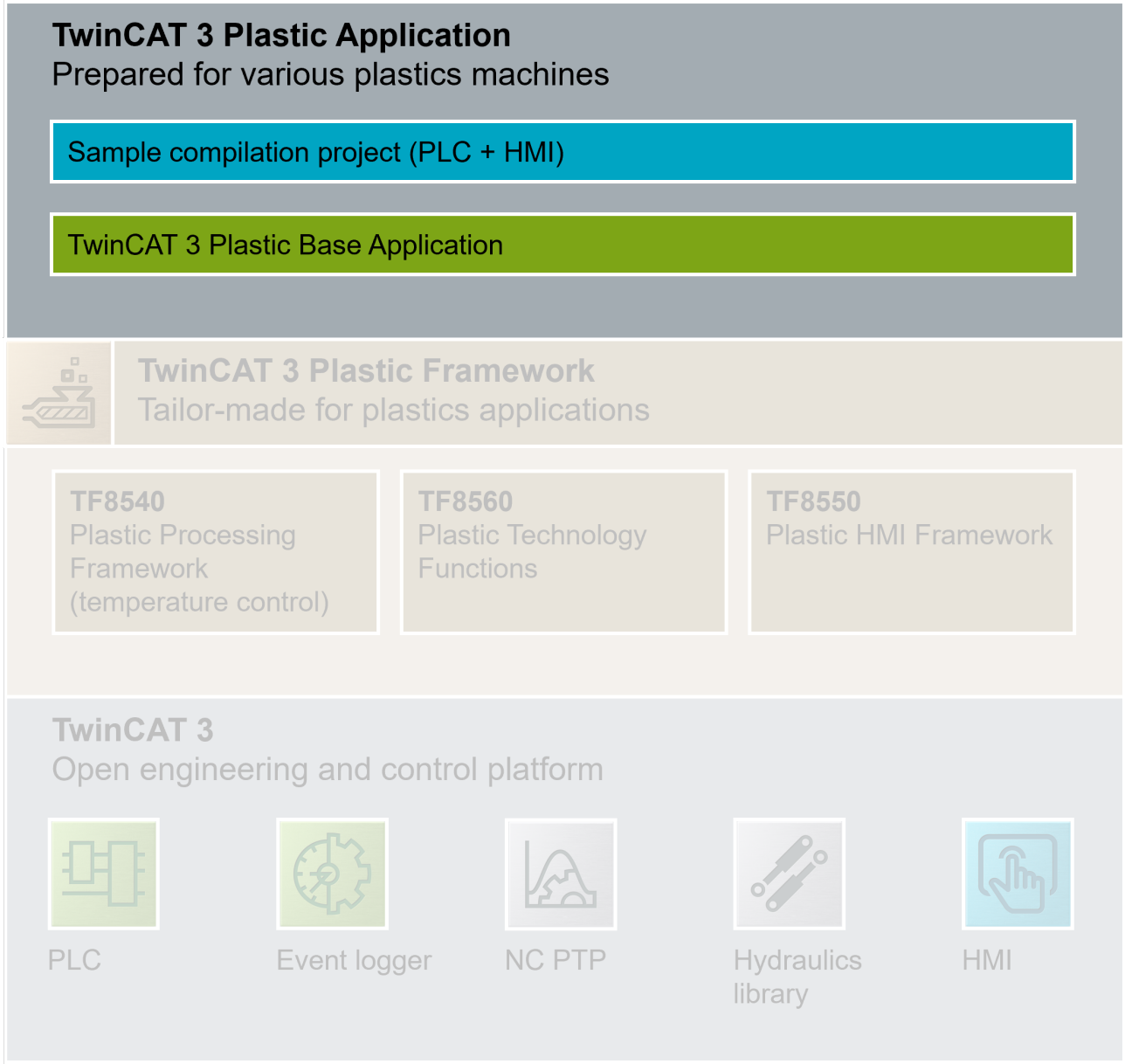
In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Introduction

The TwinCAT 3 Plastic Application represents the third implementation level of the industry solution for plastics machines from Beckhoff Automation. In this level, the technology-specific functions of the TwinCAT 3 Plastic Framework (second level) are implemented in production-specific execution.



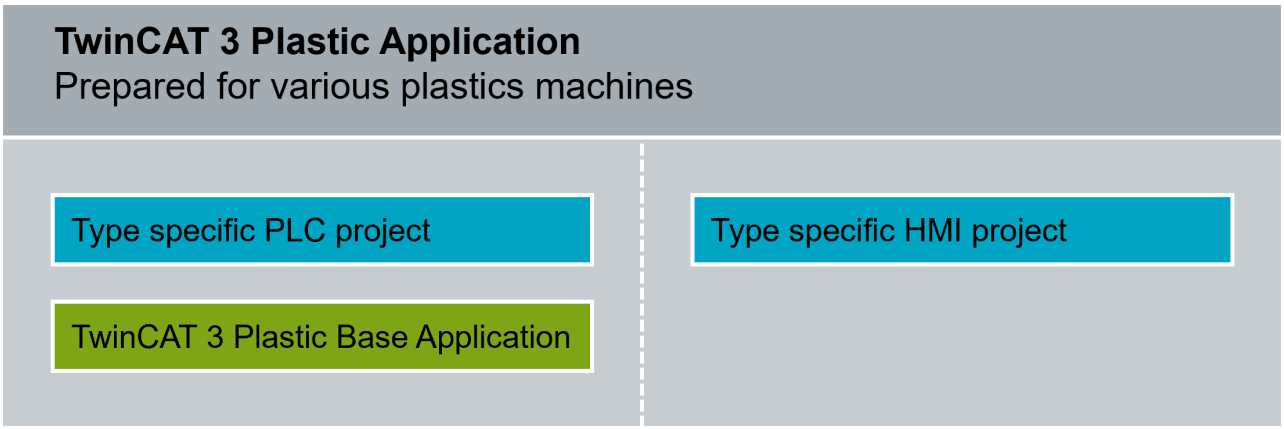
Various production part processes can be listed as examples of production-specific functions:

- Blow molding application: wall thickness control (WTC)
- Injection molding application: injection process (Injection Unit)
- Extrusion application: constant extrusion (extruder)

In the TwinCAT 3 Plastic Application, precisely these production part processes are implemented and preconfigured as examples. The source code for the Plastic Application is provided free of charge, which makes it possible to edit and extend the solution in any way. Therefore, the Base Application is ideal as a basis for starting a new project.

Concrete structure

In addition to the vertical division into Sample Code and Plastic Base Application, the project can also be differentiated horizontally: **PLC** and **HMI.**



For the horizontal division, there are two projects on the PLC side, which implement the functions of the Plastic Application in a coordinated manner.

In the vertical distinction, both projects are divided into a general part (Plastic Base Application) and a machine-specific part (Type specific PLC project). This division ensures that even after the start of a project, the further development of the Plastic Application can be benefited from.

Editing the Plastic Base Application not recommended

Editing the Plastic Base Application is quite possible. However, the option of support from Beckhoff Automation in terms of adaptations and extensions is thereby forfeited. Normally, you only edit the machine-specific part (Sample Code) and use the Plastic Base Application as a library.

Currently, the Plastic Application supports the following machine types:

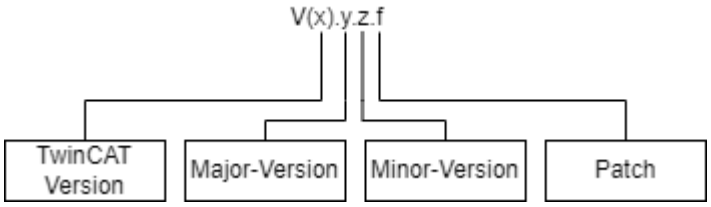
- Blow molding machines
- Extrusion machines

Further machine types are planned for future development.

3 Version of the documented software

Project:

TwinCAT 3 Plastic Application
V12.6.0



Dependencies:

Name	License number	Version
TwinCAT 3	/	3.1.4024.42 (or higher)
TwinCAT 3 HMI	TF2000	12.760.44
TC3 Plastic TC HMI Controls	TF8550	12.8.1
TC3 Plastic Technology Functions	TF8560	3.12.4.32

TwinCAT HMI Version

Only use the minor version of the TwinCAT 3 HMI specified here. Experience shows that minor updates may cause incompatibilities with the TwinCAT Plastic Application HMI project.

4 PLC concept and strategy

The PLC of the Plastic Application implements the control functions of the Plastic Framework (TF8540 & TF8560) in a manufacturing-oriented approach. The framework (in particular TF8560) itself is detached from this approach and implements core functions (CoreFunctions) for plastics technology. For example, a general point to point movement or cam plates are provided. In addition, the general axis interface `I_AxisBase` offers the possibility to implement the production-oriented application algorithms without specifying the drive technology. This results in the classes (FBs) of the Plastic Base Application library.

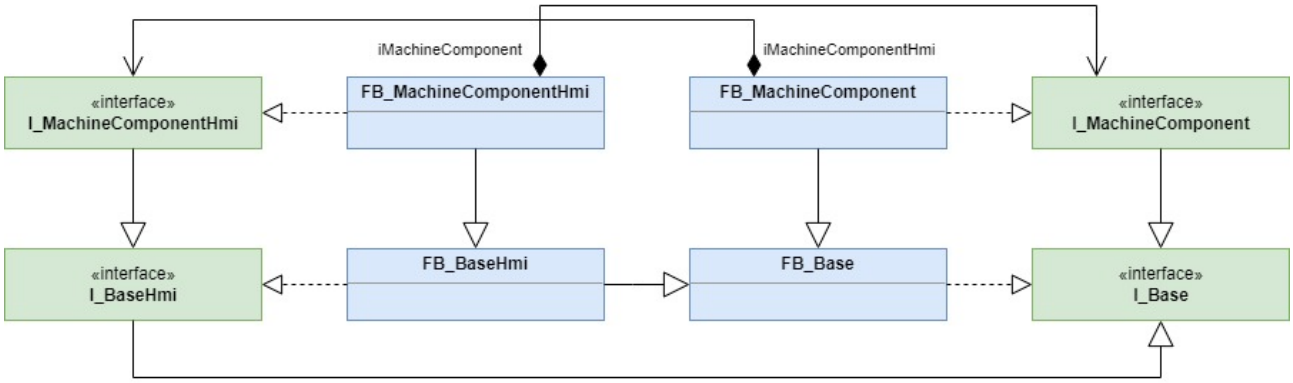
As an example of the concept, the wall thickness control (WTC) can be considered here:
This component of a blow molding machine is fundamentally based on the cam core function provided by TF8560 (`Camming`). For the user, this is often of secondary importance, since he wants to integrate the functions based on it, such as **Start wall thickness profile** or **Move to test position**, into his procedure. This step is pre-implemented by the Plastic Base Application so that the scope of tasks to be completed to the finished machine is minimized as much as possible.

Other concept points are discussed in more detail in the following subsections:

Topic	Description
PLC and HMI [▶ 13]	Communication path between PLC and HMI
Inheritance [▶ 14]	Use of inheritance for more efficient use of redundant code and easier function extension
Interfaces [▶ 14]	Structure of interface in Plastic Base Application and programming with interfaces to extend control flexibility
Initialization & Cycle Methods [▶ 15]	Initialization of references, parameters and data
Data handling [▶ 16]	Data handling at machine and product level
Project structure [▶ 18]	Explanation of the TwinCAT project

4.1 HMI interface

The communication between PLC and HMI is conceptually considered in the Plastic Application. This ensures that controlled access of the HMI to the PLC is possible. In addition, it is also quickly apparent in PLC programming that the respective sections are input and/or output values of the HMI.

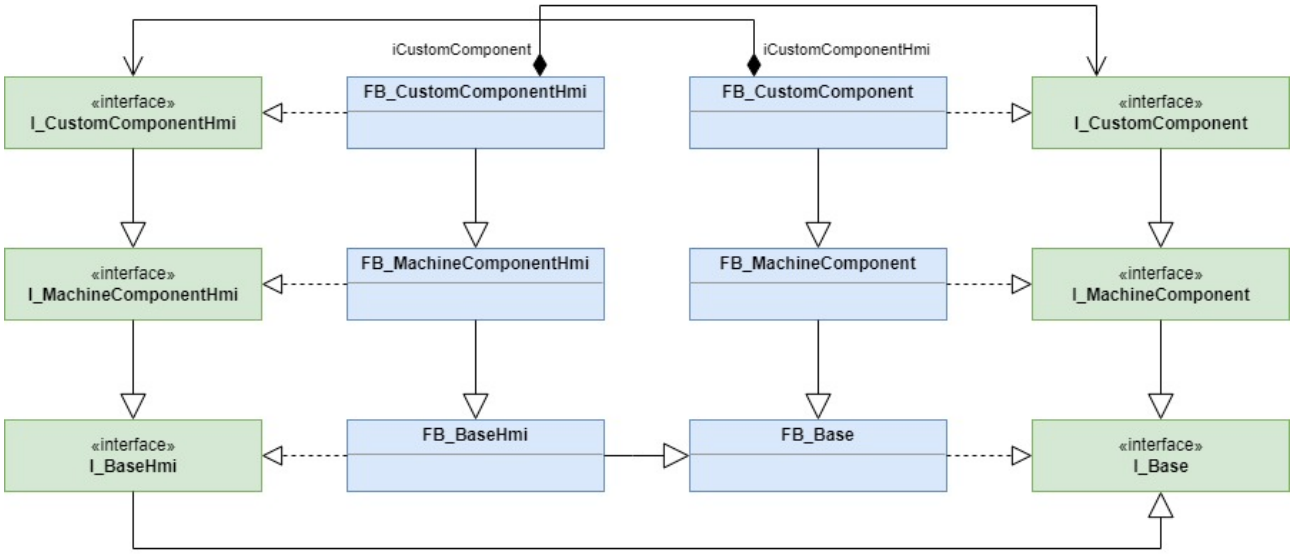


The concept is for each class with information exchange to be given a second class, which assumes precisely this function. This results in the parallel-class model. In addition to the main class (`FB_Xyz`), there is a parallel class of type `FB_XyzHmi` for each added class. The parallel class is linked to the main class via an interface. In the instantiation, the instance of the parallel class must be passed to the instance of the main class for this purpose. Depending on the use case, the main class of the parallel class can also be passed in this step.

This arrangement results in a division of control data and user input/display values. In addition, controls from the T8550 Plastic TC HMI Controls are implicitly supported, greatly improving the integrity of the project. In the same way, recipe management can be optimally linked to and used with the PLC.

4.2 Modification and extension by inheritance

In the vertical division of the project structure into Plastic Base Application and Sample Code, the extension is implemented using IEC 61131-3 inheritance. This means that the machine-specific part (Sample Code) is freely adaptable, but can build on the implemented algorithms of the Plastic Base Application.



The consistent use of inheritance for changes to the machine controller also enables updates to the Plastic Base Application to be applied directly by Beckhoff Automation.

Editing the Plastic Base Application not recommended

Editing the Plastic Base Application is quite possible. However, the option of support from Beckhoff Automation in terms of adaptations and extensions is thereby forfeited. Normally, you only edit the machine-specific part (Sample Code) and use the Plastic Base Application as a library.

Concept of object-oriented programming

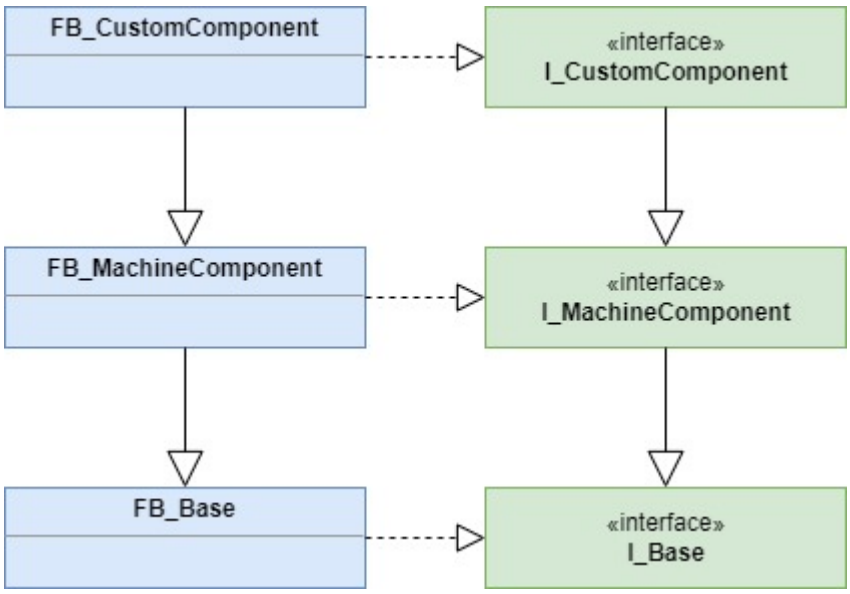
It is recommended to deal in detail with the concept of object-oriented programming according to IEC 61131-3 in order to be able to use the advantages effectively. Further information on object-oriented programming can be found in the documentation for TwinCAT 3 PLC programming

In Plastic Application, there are a total of four cases where an inheriting class (`EXTENDS`) can extend, modify or remove the inherited class:

- Runtime memory in the class variable declaration
  - Extension possible with `EXTENDS`
- Add, overwrite and extend properties or methods
  - Extension possible with `SUPER^.` call
- Extend, modify and disable inner procedures
  - Extension possible with FB_AdaptableSequence [▶ 97]
- Extend and modify HMI parallel class
  - Extending the interface and the parallel class with `EXTENDS` possible

4.3 Use of interfaces

The Plastic Base application, like the TF8560 Plastic Technology Functions, is based on working with interfaces. For this purpose, a standard interface exists on the part of the Plastic Base Application for almost every class, which provides access to the relevant properties and methods.



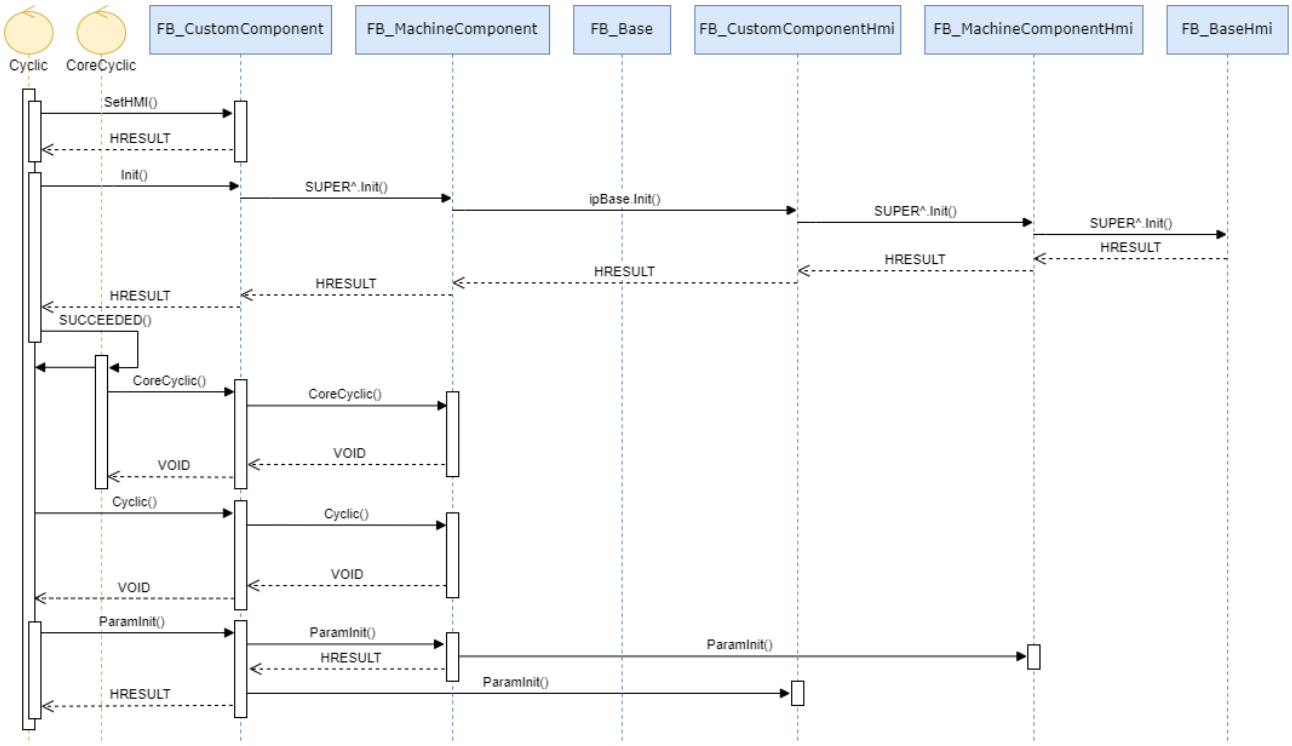
For many practical use cases in programming, access via an interface turns out to be much clearer. For example, methods for internal communication between two classes as well as other properties are not visible.

When extending a class using inheritance (see section Inheritance [▶ 14]), it is also possible to extend the corresponding standard interface. This can be accomplished in a comparable way to inheriting a class with the keyword `EXTENDS`.

4.4 Initialization and cycle methods

Since the Plastic Base Application concept frequently works with reference values such as interfaces, it must be ensured that the reference values used have a valid assignment. In order not to additionally burden the runtime by the constant validation of these assignments, the Plastic Base Application implements a concept for the initialization of an object.

The access of invalid reference values mainly concerns the call of cycle methods of a Plastic Base Application object. For this reason, an object must have been successfully initialized before the cycle method may be executed.



Handling with the FB_BaseRuntime class recommended

All the initialization steps and cycle calls described below can be implemented using FB_BaseRuntime. This means that objects only have to be passed to the runtime. If the initialization steps fail, corresponding information is output via the TcEventLogger.

SetHMI() method

By means of the SetHMI() method an HMI parallel class can be passed to the main class. The method must always be executed before the Init() method, unless the class also works without HMI parallel class.

Error code	Abbreviation	Description
0x701	SrvNotSupported	The class does not support HMI parallel class.
0x71B	InvalidInterface	The passed interface is invalid.

Init() method

The Init() method is used to check the initialization state in addition to setting and configuring reference value assignments. Accordingly, when initializing an object, the HRESULT return value of the method must be checked. If the execution of the Init() method is unsuccessful, an error exists in the structure of the program codes. This error must be solved on code level.

Use of the BaseState

Redundantly to the return value of the Init() method, the BaseState of the base class FB_Base can also be checked for a value greater than E_BaseState.eInit.

Cyclic() method

The Cyclic() cycle method acts as an object-oriented version of a class instance call. I.e., the method is usually called once per cycle to process procedures and/or updates. The routines implemented in the Cyclic() method are mostly medium priority mechanisms. Therefore, this method is suitable for the implementation of a predefined command sequence or the processing of control commands from the HMI (e.g. the implementation of manual functions).

ParamInit() method

The ParamInit() method is used to initialize parameters. I.e., in this method, hardcoded default values are defined with which commissioning is started without preconfigured machine data. Since asynchronous communication may also be required in this method, execution may continue over several cycles until the HRESULT return value signals the success of the operation. It is recommended to add a timeout to detect a failure of the execution.

Loading the machine data

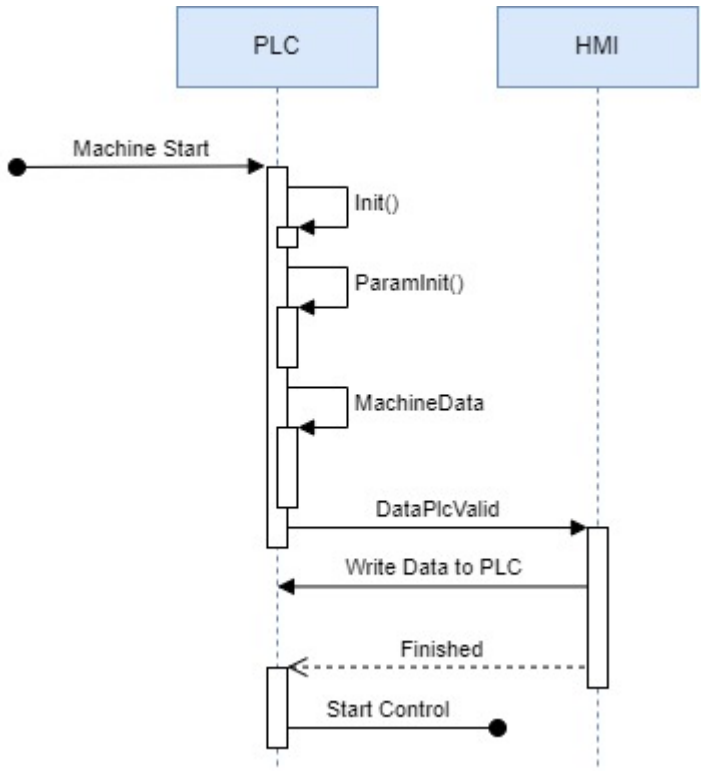
If loading machine data from a file is desired when starting the machine, this is not possible in the ParamInit() method. Therefore, implement a separate procedure that starts following the successful execution of ParamInit(). This is implemented by the FB_BaseRuntime class in the Plastic Application project.

CoreCyclic() method

The special modification of a Cyclic() method as CoreCyclic() can be found again with individual classes such as FB_Axis. This method handles particularly real-time critical tasks such as calling the cycle method of a TF8560 axis. Since such an axis also contains control algorithms, it is recommended to choose the cycle time as small as possible (standard 2 ms).

4.5 Data handling

The concept for the Plastic Application data handling is split into several levels.



Essentially, the Plastic Base Application implements three levels for loading parameters and other data. These levels are executed one after the other during the start-up phase and can partly be executed again after the start.

Standard parameterization (in code)- ParamInit()

After successful execution of the Init() method, the default parameters defined in the code are loaded. Therefore, this level is considered to be the foundation, but can be overwritten by any subsequent instance.

Design-\commissioning-dependent data in a binary file - MachineData (machine data)

Machine data includes data that is set only once during the construction, commissioning or modification of the machine. Since this data can influence the machine behavior, the file is stored in binary format and roughly secured against external manipulation by a CRC checksum. The management of the machine data is completely handled by the PLC, but can be set and/or loaded/saved through the HMI.

The implemented machine data handling is based on the TF8560 utilities. The PLC code part is split into two classes: containers and components.

Containers are integrated into many objects of the Plastic Base Application and serve as a collection object for the data of a file. A class that contains machine data either has its own container or has a component that can be added to a container. The classes with their own container inherit from the class FB_BaseMd and provide an interface to it. In the inheritance structure, multiple components (per inheritance level) are added to the container. This also allows the scope of the machine data to be further expanded in the end application.

Since the indices of the parameters must not be chosen arbitrarily for this purpose, the parameters are stored in the Plastic Base Application according to a defined scheme. This also ensures that the parameters can be prepared outside the controller in a program that will be provided in the future.

Product data / machine data - HMI-Recipe

The parameters to be regularly reset for the variation of the production process are summarized under the recipe data. This includes, for example, the set temperatures of the temperature zones, timings, production speeds, etc... The recipe data is managed collectively by the HMI and can be expanded as required in engineering. For more detailed information on recipe management, please refer to the chapter Recipe management [▶ 173].

4.6 Project structure

The individual components of the project tree are discussed in more detail below.

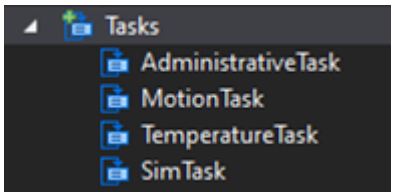
System > License

Order No	License	Instances	License TAN	Current Status
TC1200	TC3 PLC	cpu license		unknown
TF2000	TC3 HMI Server	cpu license		unknown
TF5000	TC3 NC PTP	cpu license		unknown
TF5050	TC3 NC Camming	cpu license		unknown
TF5810	TC3 Hydraulic Positioning	cpu license		unknown
TF8540	Plastic Processing TwinCAT Framework	cpu license		unknown
TF8550	TC3 Plastic TC HMI Framework	cpu license		unknown
TF8560	TC3 Plastic Technology Functions	cpu license		unknown

Licenses may be excluded from the project under the following conditions:

Number	Context	Exclusion option
TC1200	PLC project	No
TF2000	TwinCAT HMI	Removing manually from the license list
TF5000	Electric NC axes	Removing NC task under MOTION
TF5050	Electric NC transformer axes	Removing NC transformer axes from the PLC
TF5810	Hydraulic axes	Removing Tc3_PlasticHydraulics reference and axes from PLC
TF8540	Temperature control	No
TF8550	HMI Plastic Controls	Removing NuGet packages and controls in HMI
TF8560	Plastic technology functions	No (used consistently in the Plastic Application)

System > Tasks



The four created tasks follow the following strategies:

- **AdministrativeTask** - General control tasks of medium priority
  - Standard cycle time: 10 ms
  - Calling the HMI <> PLC Recipe communication
  - Storage and loading routines
  - Communication commissioning tool of hydraulic axes
  - Cyclic processing of application routines like: manual functions, process sequence, etc..
- **MotionTask** - Real-time critical routines
  - Standard cycle time: 2 ms (= NC task)
  - Cycle methods of the motion axes
  - Internal control mechanisms
- **TemperatureTask** - Inertial temperature control
  - Standard cycle time: 25 ms
  - Calling the temperature control
- **SimTask** -Simulation
  - Standard cycle time: 2 ms

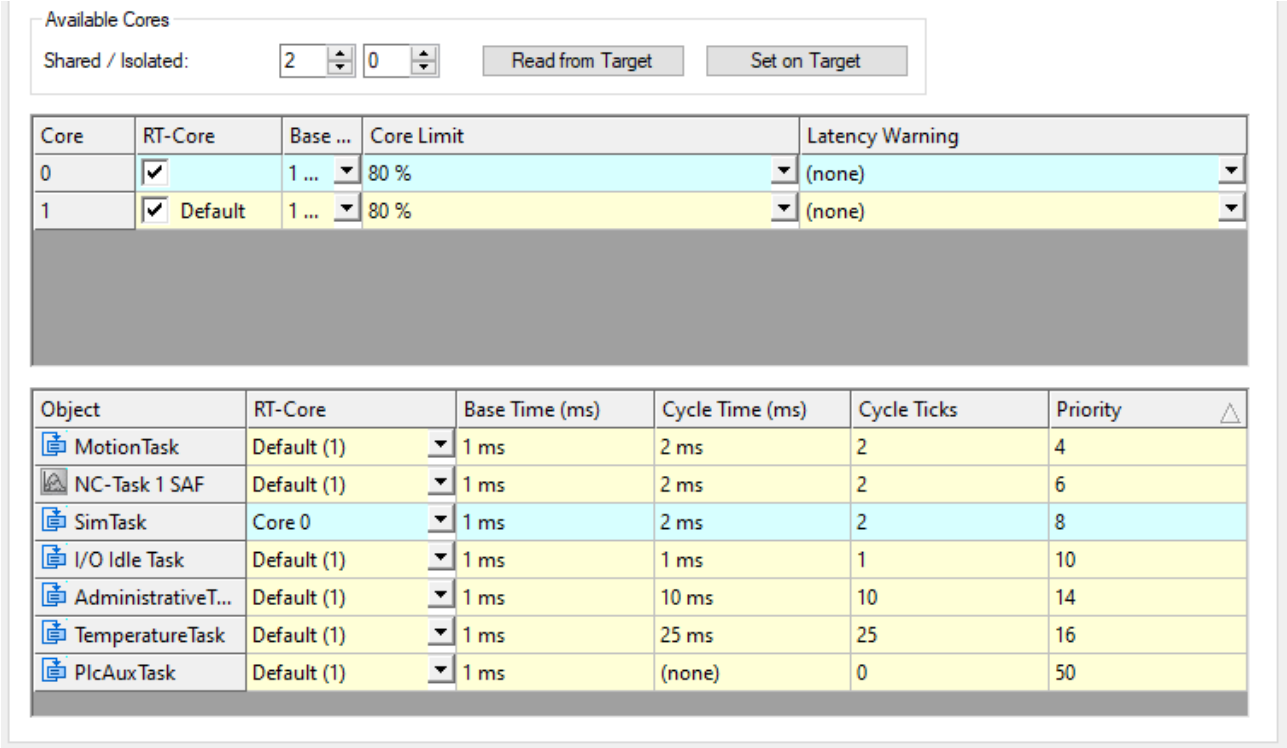
- Calling the cycle methods of the simulation

NOTICE

Correct cycle time for temperature control

The cycle time of the temperature control should be asynchronous to the frequency of the AC supply voltage. A synchronous cycle time promotes instability of the control!

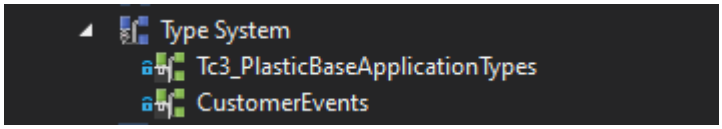
System > Real-Time



The PLC tasks can be split according to the number of cores and the single core performance of the CPU. For development systems, the project configuration can be adopted; for end devices, the following configurations have been tested in the field:

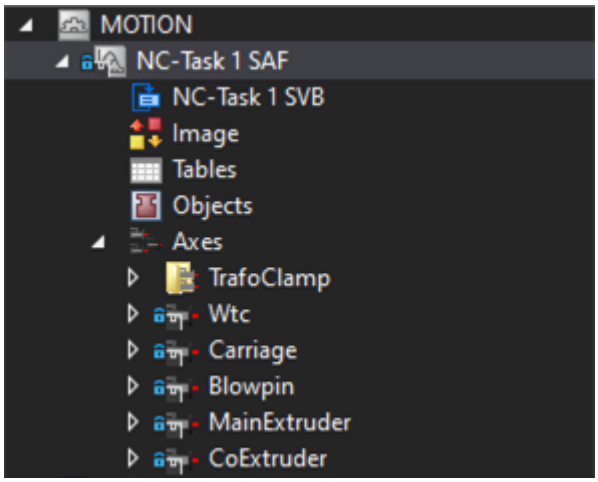
IPC / CX	Adjustments
CX2033	None
CX2043	None
C6030 (Basis)	None

System > Type System



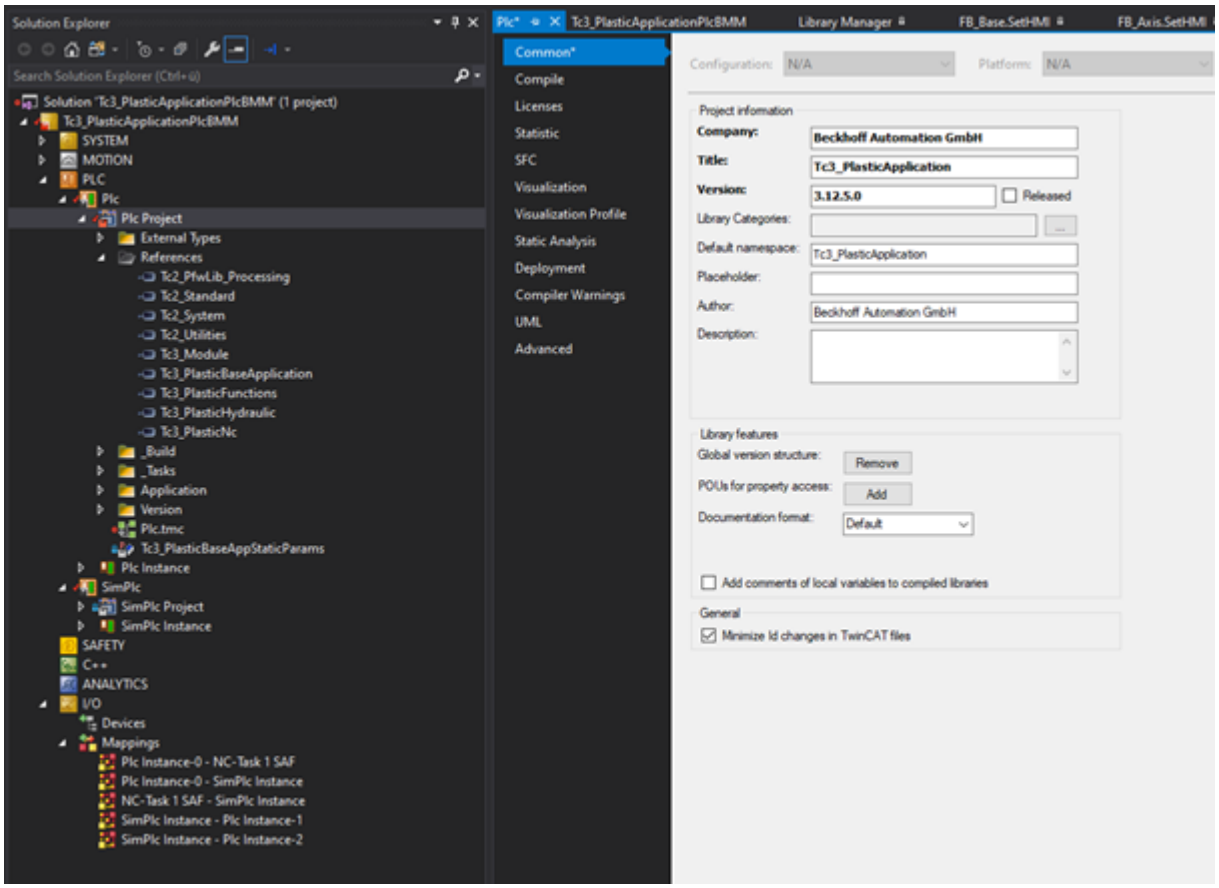
The Type System contains the used EventClasses. These EventClasses contain the respective events that are used in the Plastic Application.

Motion



All exemplarily implemented NC axes (electrical) are created in the NC task. The designations of the axes are based on the manufacturing significance.

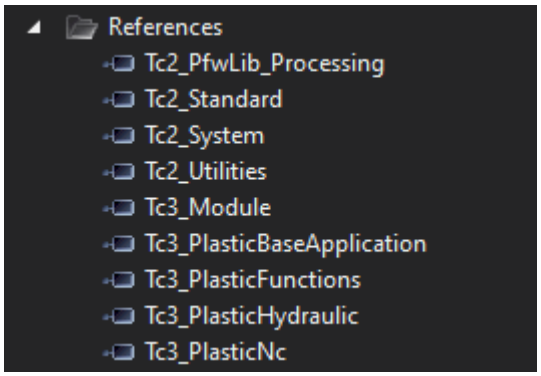
PLC > Version



The currently used version of the project can be determined in two ways:

- PLC > PLC > PLC Project > Properties (right click) > Common > Version:
- PLC > PLC > PLC Project > Version > Global_Version

PLC > Library references



The following libraries are installed in the project in addition to some general libraries from the general TwinCAT pool:

- Tc2_PfwLib_Processing
  - Temperature control algorithms
- Tc3_PlasticBaseApplication
  - Collection of application-oriented classes, explained in this documentation
- Tc3_PlasticFunctions
  - Virtual axis interface and technology functions
- Tc3_PlasticHydraulic
  - Hydraulic axes compatible with virtual axis interface
- Tc3_PlasticNc
  - Electric axes compatible with virtual axis interface

Error despite existing library

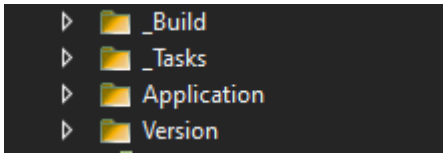
For compatibility and maintenance purposes, the versions of the included libraries are fixed. If you have already worked with the mentioned libraries in past versions, you have to install the updated versions. These can be found in the delivered project folder at **Dependencies**.

NOTICE

Adaptation to an older version not permitted

Do not change the library versions set in the project to an older version. Incompatibilities and unpredictable behavior of the software may occur!

PLC > Code



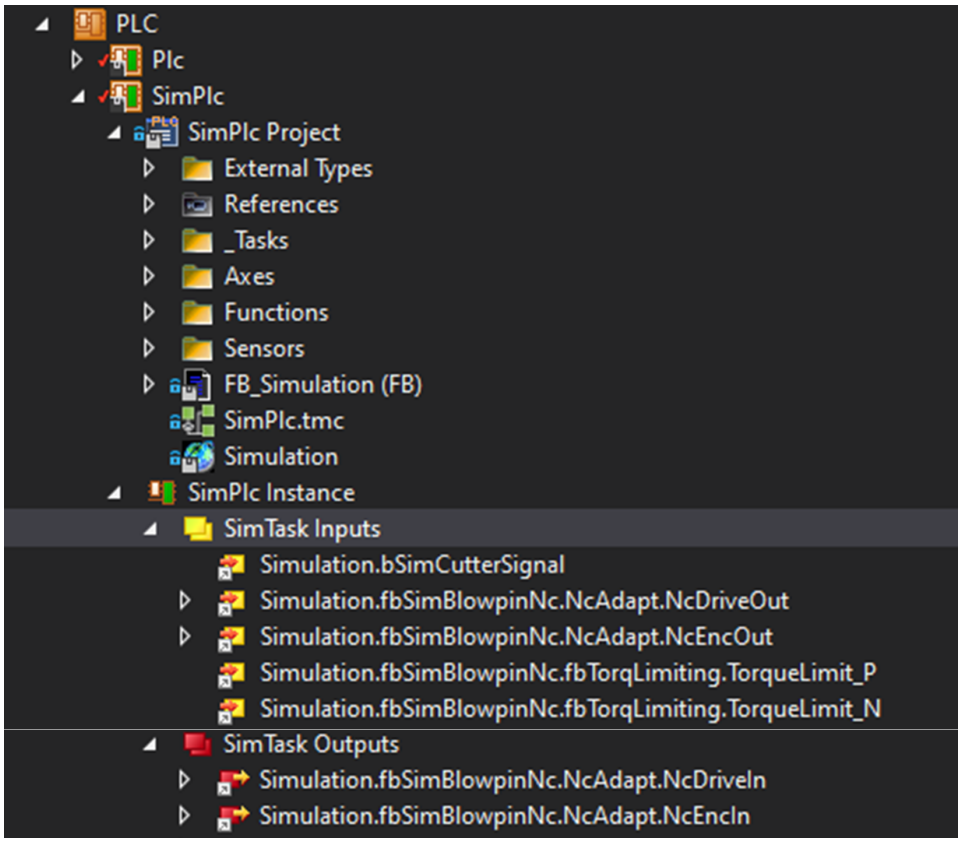
The control code is divided into four folders.

- _Build: Machine type configuration
- _Tasks: Instantiation of the runtime and definition of the task calls
- Application: Application program
- Version: Automatically generated GVL with the used project version

Further information about the PLC code

- Class overview [▶ 23]
- Extension by inheritance [▶ 14]

PLC > SimPlc



In the second PLC (SimPlc) a machine simulation is implemented. This can be connected to the main control via mapping (comparable to the I/O of a machine). This allows the PLC to be implemented without including simulation elements. In addition, simulation can also be extended with the elements of PLC programming. This offers the advantage that the control PLC from project preparation does not have to be adapted to operation with a real machine. Only the mapping to I/O components is required.

5 PLC-API (class overview)

5.1 Base - Base classes

5.1.1 FB_Base

FB_Base is the base class for most of the available classes of the Plastic Base Application.

Internal functions:

- Standard Error and Reset
- Creation of an instance default name
- Provision of the event interface

Syntax:

```
FUNCTION_BLOCK FB_Base
```

Properties

Name	Type	Access	Initial value	Description
BaseState	E_BaseState [▶ 24]	Get	eNotHandled	Initialization state of the class instance.
Error	BOOL	Get	FALSE	Class is in an error state.
Name	STRING	Get, Set	-	Specifies the name of the class instance.
ResultMessage	I_TcMessage	Get	-	Access to the class internal EventLogger.
_Name	STRING	Get	<SourceName>	[PROTECTED] .Name without leading 'fb'

Methods

Name	Description
Reset()	Resets the error state of the class.
SetHMI()	General assignment method for an FB_BaseHmi extending class.

Interfaces

Type	Description
I_Base	Standard interface on FB_Base.
I_BaseEmpty	For extension without standard interface.
I_BaseDev	Covers all methods and properties

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.1.1.1 E_BaseState

Base State for the state of a Plastic Base Application object.

Syntax:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_BaseState :
(
    eFailed := -9999,     // Init routine failed
    eReinit := -100,      // Object has to be reinitialized

    eNotHandled := 0,     // It starts Here

    eInit,                // FB_Init is succeeded
    eReady,               // Init is succeeded
    eIdle,                // ParamInit is succeeded
    eBusy,                // implementation usable busy flag

    eError    := 1000     // implementation usable error flag
);
END_TYPE
```

Values

Name	Description
eFailed	Init procedure failed.
eReinit	Object must be re-initialized.
eNotHandled	Object was not handled.
eInit	FB_init was executed successfully.
eReady	Init was executed successfully.
eIdle	ParamInit was executed successfully.
eBusy	Implementation-available busy flag.
eError	Implementation-available error flag.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.1.2 FB_BaseHmi

FB_BaseHmi is the base class of all HMI parallel classes. The provided Init(ipBase) method predefines that the parallel object is passed via the I_Base interface.

Syntax:

```
FUNCTION_BLOCK FB_BaseHmi EXTENDS FB_Base
```

Methods

Name	Description
Init()	Default initialization method to pass the base class.

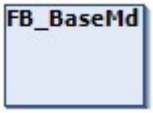
Interfaces

Type	Description
I_BaseHmi	Standard interface on FB_BaseHmi.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.1.3 FB_BaseMd



Extends the base class FB_Base by the functions of the machine data handling. The structure of the machine data is explained in more detail in the chapter Machine data [▶ 73].

Syntax:

```
FUNCTION_BLOCK FB_BaseMd EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
MachineData	I_MdBaseContainer	Get	-	Interface to the machine data handling.

Methods

Name	Description
AddMdComp()	[PROTECTED] Adds a component of type I_MdComponent to the machine data container.
Cyclic()	Cycle method - call once per PLC cycle.

Interfaces

Type	Description
I_BaseMd	Standard interface on FB_BaseMd.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.1.4 FB_BaseMdHmi

HMI class in parallel with the base machine data class FB_BaseMd. Is necessary for loading and saving the machine data via the HMI.

Syntax:

```
FUNCTION_BLOCK FB_BaseMdHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
ParamHandle	REFERENCE TO FB_MdHandleHmi	Get	-	Interface for handling the machine data via the HMI.

Methods

Name	Description
Init()	Default initialization method to pass the base class.

Interfaces

Type	Description
I_BaseMdHmi	Standard interface on FB_BaseMdHmi.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.2 Runtime - Handling of initialization and cycle calls

Advantage / Benefit

The FB_BaseRuntime [▶ 26] class of the Plastic Base Application is used to simplify cycle calls and automated initialization of control objects. This greatly simplifies several steps such as creating, initializing, checking return values, generating error messages, cyclic calls, etc.

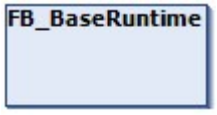
Requirements

For an object to be attached to the runtime, one of the following interfaces must be implemented.

Interfaces

Name	Description
I_RuntimeInterface	General interface for an attachable control object.
I_OneTaskInterface	Interface with an executable method.
I_TwoTaskInterface	Interface with two executable methods.
I_TempTaskInterface	Interface with a slow executable method.

5.2.1 FB_BaseRuntime



The class automates the initialization and cyclic calls of instantiated control objects such as axes, sequence algorithms, temperature controllers, etc.

Internal functions:

- Initialization (Init(), ParamInit() and MdInit())
- Cycle calls (3-tasks)
- PlcMcManager support function (when using hydraulic axes)
- Software version check

Syntax:

```
FUNCTION_BLOCK FB_BaseRuntime EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
DisableMdFillCollections	BOOL	Get, Set	FALSE	Blocks the filling of attached FB_MdCollection [▶ 77] instances.
DisableMdInit	BOOL	Get, Set	FALSE	Blocks the automatic loading of the machine data by the runtime.
DisableRecipe	BOOL	Get, Set	FALSE	Locks the recipe handling of objects that support the interface I_Recipe [▶ 72].
MdInitExecuted	BOOL	Get	FALSE	Loading of the machine data was performed for all objects.
NumRuntimeObjects	INT	Get	0	Number of appended runtime objects.

Methods

Name	Description
Append() [▶ 28]	Appends a control object to the runtime.
Clear()	Deletes the list of appended control objects.
MdSaveAll()	Starts the saving process of all appended control objects with machine data container.

Cycle methods

Name	Description	Cycle time (recommended)
CoreCyclic()	Fast cycle method for control objects with I_TwoTaskInterface.	2 ms
Cyclic()	Normal cycle method for control objects with I_OneTaskInterface or I_TwoTaskInterface.	10 ms
TemperatureCyclic()	Slow cycle method for control objects with I_TempTaskInterface.	25 ms

The cycle methods must be called with programs (PRG) assigned to different tasks, so that the runtime can assign the appended objects to the individual tasks.

Possible events

ID	Description	Alarm/Message
1xx	Recipe	Message
10xx	Initialization	Alarm + Message
11xx	Parameterization	Alarm + Message
12xx	Version incompatibility	Alarm
20xx	Machine data	Alarm + Message

Interfaces

Type	Description
I_BaseRuntime	Standard interface on FB_BaseRuntime.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.2.1.1 Append()

Appends a control object to the runtime.

Syntax:

```
METHOD Append : VOID
VAR_INPUT
    iObj:           I_RuntimeInterface;
    iObjHmi:        I_BaseHmi;
END_VAR
```

Inputs

Name	Type	Description
iObj	I_RuntimeInterface	Object to be attached to the runtime.
iObjHmi	I_BaseHmi	Associated HMI object which is to be linked to the object.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.2.2 I_RuntimeInterface

Represents the general interface for a runtime-compatible control object.

Syntax:

```
INTERFACE I_RuntimeInterface
```

Methods

Name	Description
Init()	Method for checking the initialization of an object
ParamInit()	Method for standard parameterization of an object
SetHMI(ipBaseHmi)	Method for passing a parallel HMI object

All methods are checked by the FB_BaseRuntime class for the HRESULT return value.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3 Axis - General axis types

5.3.1 FB_Axis

Represents the standard class for all axis objects based on TF8560 axes.

Syntax:

```
FUNCTION_BLOCK FB_Axis EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
BaseAxisInterfaces	I_AxisBase	Get	NULL	Interface to the assigned TF8560 axis.
ErrorAlarmTL	TcEventSeverity	Get, Set	.Error (3)	Severity of the alarm that the axis triggers in the error state.
Homing [▶ 33]	I_Homing	Get	-	Homing functions
Specific [▶ 30]	I_AxisSpecific	Get	-	Axis technology-specific functions/values

Methods

Name	Description
CheckAxisState() [▶ 35]	Checks the current state of the axis.
FeedEnable()	Enable of a direction of movement.
Power()	Switching on the drive control.
Reset()	Reset axis errors.
SetAxisRef()	Assignment method for the TF8560 axis.
SetPosition()	Setting the axis position.

Cycle methods

Name	Description
CoreCyclic()	Cycle method with short cycle time (default: 2 ms)

Interfaces

Type	Description
I_Axis	Standard interface on FB_Axis
I_TwoTaskInterface	Runtime interface for two PLC tasks

Possible events

ID	Description	Alarm/Message
10	Axis Errors	Alarm
20	Axis command rejected	Alarm

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.5)

5.3.1.1 FB_AxisSpecific

The class implements technology-specific functions of a TF8560 axis.

Syntax:

```
FUNCTION_BLOCK FB_AxisSpecific EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Hydraulic [▶ 31]	I_HydraulicFunctions	Get	-	Hydraulic-specific functions
Nc [▶ 31]	I_NcFunctions	Get	-	NC-specific (servo-electric) functions
Trafo [▶ 32]	I_TrafoFunctions	Get	-	Transformation-specific functions

Interfaces

Type	Description
I_AxisSpecific	Standard interface on FB_AxisSpecific

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.1.1.1 FB_HydraulicFunctions

Implements hydraulic-specific functions of a TF8560 axis.

Syntax:

```
FUNCTION_BLOCK FB_HydraulicFunctions EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AutoIdentFinished	BOOL	Get	FALSE	The executed characteristic curve measurement was completed successfully.
IsHydraulic	BOOL	Get	FALSE	Hydraulic functions are supported by the assigned axis type.

Methods

Name	Description
AutoIdent()	Characteristic measurement of a hydraulic axis.

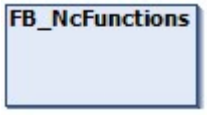
Interfaces

Type	Description
I_HydraulicFunctions	Standard interface on FB_HydraulicFunctions

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.1.1.2 FB_NcFunctions



Implements specific functions for an NC-based TF8560 axis.

Syntax:

```
FUNCTION_BLOCK FB_NcFunctions EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
IsNc	BOOL	Get	FALSE	NC functions are supported by the assigned axis type.

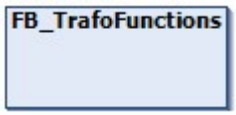
Interfaces

Type	Description
I_NcFunctions	Standard interface on FB_NcFunctions
I_AttachableMdInterface	Interface for containerless machine data components

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.1.1.3 FB_TrafoFunctions



Implements specific functions for a transforming TF8560 axis.

Syntax:

```
FUNCTION_BLOCK FB_TrafoFunctions EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
IsTrafo	BOOL	Get	FALSE	Transformer functions are supported by the assigned axis type.
LowerLimit	LREAL	Get, Set	0.0	Lower limit of the real transformation
ParamList	I_Parameter	Get	-	Internal list for storing the TableGenerator parameters in the machine data
Table	I_CammingLookUp	Get	-	Interface to the instance of the table
TableGenerator	I_TrafoTableGenerator	Get, Set	NULL	Interface to a TableGenerator class to be assigned
UpperLimit	LREAL	Get, Set	0.0	Upper limit of the real transformation

Methods

Name	Description
AssignTableToAxis()	Assigns the internal table to the axis instance (means activating the transformation curve)
ConvDriveEndsToLoadEnds()	Calculates the resulting software end positions of the load side using the transformation table
CopyTableDriveEnds()	Copies the start and end position of the TableGenerator class to the software end positions of the drive axis.
FillDebugTable()	Copies the contents of the internal table into a two-dimensional array
ParamListToTableGenerator()	Copies the internal parameter list to the parameters of the TableGenerator
TableGeneratorToParamList()	Copies the parameters of the TableGenerator into the internal parameter list

Possible events

ID	Description	Alarm/Message
201	Calling the TableGenerator property without previous assignment	Message

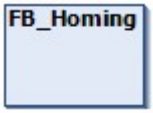
Interfaces

Type	Description
I_TrafoFunctions	Standard interface on FB_TrafoFunctions
I_AttachableMdInterface	Interface for containerless machine data components

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.5)

5.3.1.2 FB_Homing



The Homing class implements standard procedures for homing an axis.

Syntax:

```
FUNCTION_BLOCK FB_Homing EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AbsoluteSwitch	BOOL	Set	FALSE	Input value of an absolute position switch
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
ExternalStates	I_AdaptableSeqExt	Get, Set	NULL	Interface to an object with additional homing procedure steps

Methods

Name	Description
DoAbort()	Cancels the currently active homing.
DoAbsSwitch()	Starts axis homing against an absolute position switch.
DoAbsSwitchSequence()	Starts a sequence of DoAbsSwitch() and DoFinish().
DoBlock()	Starts homing against an end stop.
DoBlockDetect()	Starts the position detection at an end stop.
DoBlockSequence()	Starts a sequence of DoBlock(), DoBlockDetect() and DoFinish().
DoFinish()	Starts the termination of homing.
DoSetZero()	Sets the current position of the axis to 0

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
HomingStates() [▶ 34]	State machine of the homing procedures

Event-driven methods (callback methods)

Name	Description
ExtAdaptSeq()	Dummy method() for use without "ExternalStates"

Possible events

ID	Description	Alarm/Message
4000	Homing not allowed	Alarm
4001	Homing failed	Alarm
4002	Save homing was successful	Message
4003	Save homing was not successful	Message

Interfaces

Type	Description
I_Homing	Standard interface on FB_Homing
I_AttachableMdInterface	Interface for containerless machine data components
I_AdaptableSeqExt	Interface for providing external homing procedures

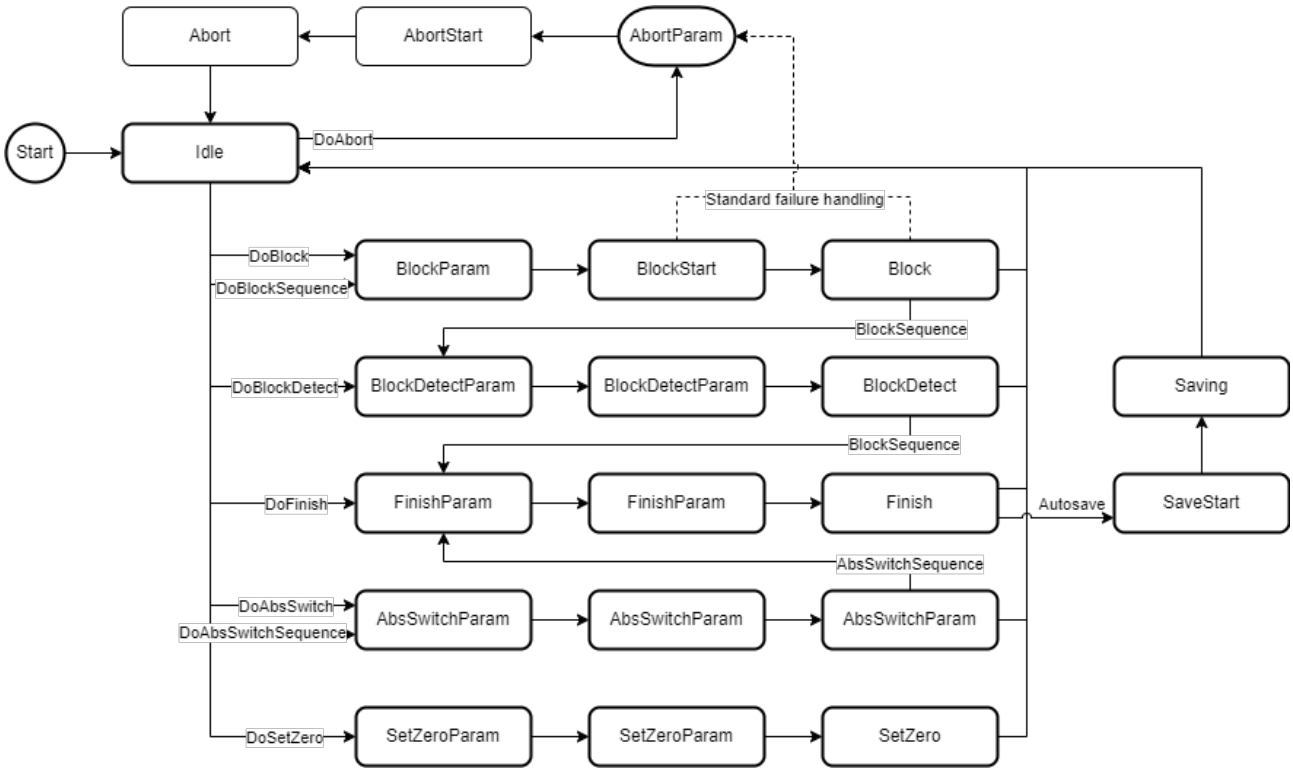
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.3.1.2.1 HomingStates()

Instance type	Instance Name
Master	`fbHomingStates`
Slaves	`aBaseSeqMembers[E_HomingState.eLength]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.3.1.3 CheckAxisState()

Checks the current state of the TF8560 axis.

Syntax:

```
METHOD CheckAxisState : BOOL
VAR_INPUT
    eState:         BOOL;
    bSendEvent:     BOOL;
END_VAR
```

Inputs

Name	Type	Description
eState	E_AxisState	Axis condition to be checked
bSendEvent	BOOL	Logging an event when the checked state does not match.

Outputs

Name	Type	Description
CheckAxisState	BOOL	Is TRUE if the axis state matches.

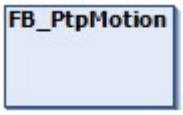
Possible events

ID	Description	Alarm/Message
1000	Axis is not in state '{0}'	Message

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.2 FB_PtpMotion



Inherits from the FB_Axis class and implements further functions for PTP-based movements. In addition, the TF8550 Control Arrow Motion graph is supported via the FB_PtpMotionHmi class.

Syntax:

```
FUNCTION_BLOCK FB_PtpMotion EXTENDS FB_Axis
```

Properties

Name	Type	Access	Initial value	Description
ClampOnlyLastSeg	BOOL	Get, Set	TRUE	Clamping is only possible in the last segment.
FixedMoveDirection[1]	BOOL	Get, Set	TRUE	Fixes the direction of movement for cluster 1 (negative) and 2 (positive).
PtpMoveFinished	BOOL	Get	FALSE	The PTP movement was executed successfully.
PtpMoveStopDone	BOOL	Get	FALSE	The PTP movement was successfully stopped.
StackSegmentCount	BOOL	Get, Set	TRUE	Aborted segments are taken into account in the subsequent command.
UseManualSpeed	BOOL	Get, Set	FALSE	Commanded PTP movements should be executed at manual speed.

[1] Is obsolete (Alternative: `FB_PtpMotionHmi.AxisMove.Moves[].Direction`)

Methods

Name	Description
CmpSegPos()	[PROTECTED] Forms the difference of two segment positions.
JogNegative()	Starts/stops a jog movement in negative direction.
JogPositive()	Starts/stops a jog movement in positive direction.
MovePtp()	Starts/stops a PTP movement with the parameterization from the HMI.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
PtpSeq() [▶ 37]	Procedure for loading a PTP movement

Possible events

ID	Description	Alarm/Message
2000	PTP command error	Message

Interfaces

Type	Description
I_PtpMotion	Standard interface on FB_PtpMotion

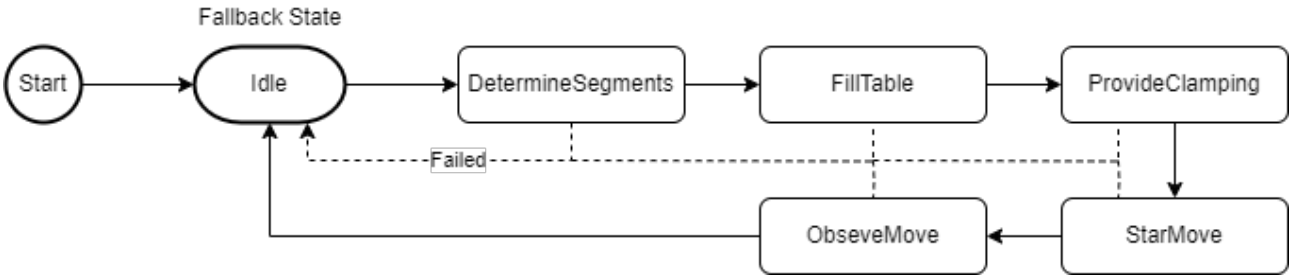
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.5)

5.3.2.1 PtpSeq()

Instance type	Instance Name
Master	`fbPtpSeq`
Slaves	`aBaseSeqMembers[E_PtpBaseSeq.eLength]`

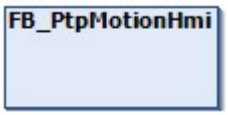
State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.3.3 FB_PtpMotionHmi



HMI parallel class of `FB_PtpMotion` class

Syntax:

```
FUNCTION_BLOCK FB_PtpMotionHmi EXTENDS FB_AxisHmi
```

Properties

Name	Type	Access	Initial value	Description
ActiveMove	INT	Get, Set	0	Index of the active MoveCluster
ActiveSegment	INT	Get, Set	0	Index of the active segment
AxisMove [▶ 38]	REFERENCE TO FB_AxisMoveHmi	Get	-	Motion configuration object
ParamPtpMotion	REFERENCE TO FB_MdPtpMotionHmi	Get	-	HMI access to the PTP-specific machine data

Interfaces

Type	Description
I_PtpMotionHmi	Standard interface on FB_PtpMotionHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.3.1 FB_AxisMoveHmi

Summarizes all Move clusters

Syntax:

```
FUNCTION_BLOCK FB_AxisMoveHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Moves [▶ 39]	REFERENCE TO ARRAY[] OF FB_MoveCfgHmi	Get	-	Move Cluster

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.3.1.1 FB_MoveCfgHmi

Summarizes the information of a Move Cluster

Syntax:

```
FUNCTION_BLOCK FB_MoveCfgHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Cams [▶ 39]	REFERENCE TO ARRAY[] OF FB_CamCfgHmi	Get	-	Cluster cams
ClampingDistance	LREAL	Get, Set	0.0	Position from which clamping starts
ClampingVelocity	LREAL	Get, Set	0.0	Pre-controlled velocity at which the clamping is performed
Direction	INT	Get, Set	0	Intended direction of the cluster • > 0 – Positive direction • = 0 – Undefined direction • < 0 – Negative direction
EndFunction	INT	Get, Set	0	ID of the Move Cluster start function
InUse	BOOL	Get, Set	FALSE	Cluster is used
ManualVelocity	LREAL	Get, Set	0.0	Manual velocity of the cluster
Segments [▶ 40]	REFERENCE TO ARRAY[] OF FB_SegCfgHmi	Get	-	Segments of the cluster
StartFunction	INT	Get, Set	0	ID of the Move Cluster end function

Methods

Name	Description
GetLastSegInUse()	[PROTECTED] Returns the last segment whose .InUse property is = TRUE

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.3.3.1.1.1 FB_CamCfgHmi

Summarizes the information of a cam

Syntax:

```
FUNCTION_BLOCK FB_CamCfgHmi EXTENDS FB_BaseHmi
```

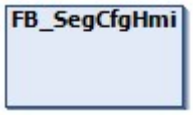
Properties

Name	Type	Access	Initial value	Description
CamActive	BOOL	Get, Set	FALSE	Cam is active
Hysteresis	LREAL	Get, Set	0.0	Threshold width
Threshold	LREAL	Get, Set	0.0	Threshold value

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.3.1.1.2 FB_SegCfgHmi



Summarizes the information of a segment

Syntax:

```
FUNCTION_BLOCK FB_SegCfgHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Acceleration	LREAL	Get, Set	0.0	Acceleration of the segment
Decelration	LREAL	Get, Set	0.0	Deceleration of the segment
IgnoreOnce	BOOL	Get, Set	FALSE	Ignores the segment once (at the next command)
InUse	BOOL	Get, Set	FALSE	Segment is used
Jerk	LREAL	Get, Set	0.0	Jerk of the segment
Limiting	LREAL	Get, Set	0.0	Pressure/torque limitation of the segment
Position	LREAL	Get, Set	0.0	Target position of the segment
SegFunction	INT	Get, Set	0	ID of the segment function
Velocity	LREAL	Get, Set	0.0	Speed of the segment

Methods

Name	Description
GetPtpLookup()	Returns the segment as a variable of type ST_LookUpPtpPoint

Interfaces

Name	Description
I_SegCfgHmi	Standard interface on FB_SegCfgHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.3.4 FB_BaseCammingHmi

Extends the FB_AxisHmi class for camming based axes with the necessary interface to the TF8550 CurveEditor.

Syntax:

```
FUNCTION_BLOCK FB_BaseCammingHmi EXTENDS FB_AxisHmi
```

Properties

Name	Type	Access	Initial value	Description
ActivateCurve	BOOL	Get, Set	FALSE	Command of the HMI to take over the cam plate on control level
CurrentIndex	UDINT	Get	0	Indicates the current index of the cam plate where the drive position is located.

Methods

Name	Description
GetActPoint()	Returns an actual value of the displayed curve.
GetSetPoint()	Returns a setpoint of the displayed curve.
SetActPoint()	Sets an actual value of the displayed curve.
SetSetPoint()	Sets a setpoint of the displayed curve.

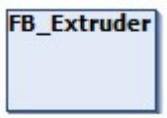
Interfaces

Type	Description
I_BaseCammingHmi	Standard interface on FB_BaseCammingHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.3.5 FB_Extruder



Inherits from the FB_Axis class and implements further functions for a continuous movement of an extruder.

Syntax:

```
FUNCTION_BLOCK FB_Extruder EXTENDS FB_Axis
```

Properties

Name	Type	Access	Initial value	Description
GuidingValue	LREAL	Get	0.0	Contains the current turnrate for a master/slave connection of two FB_Extruder objects.
MasterExtruder	I_Extruder	Get, Set	NULL	Used to assign a master extruder.

Methods

Name	Description
DoBasicRpm()	Starts a rotary motion with the base speed set in the HMI.
DoNominalRpm()	Starts a rotary motion at the production speed set in the HMI.
DoTurnrateDown()	Decreases the turnrate.
DoTurnrateUp()	Increases the turnrate.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
PowerStates() [▶ 42]	Procedure for executing the continuous rotary motion

Interfaces

Type	Description
I_Extruder	Standard interface on FB_Extruder

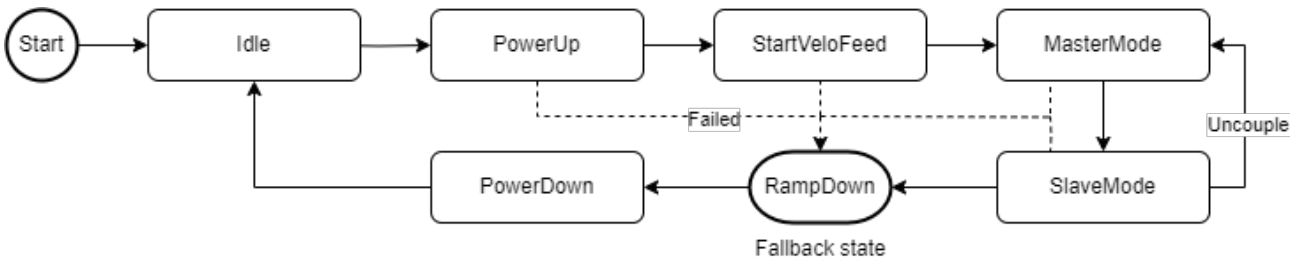
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.3.5.1 PowerStates()

Instance type	Instance Name
Master	`fbPowerStates`
Slaves	`aSeqBaseMembers[E_ExtruderPowerStates.eLength]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.4 Clamp, Carriage - Standard PTP axes

5.4.1 FB_Clamp

Implements specific functions for a clamping unit.

- Adds a closing segment to the PTP movement, which is permanently parameterized in the machine data.

Syntax:

```
FUNCTION_BLOCK FB_Clamp EXTENDS FB_PtpMotion
```

Properties

Name	Type	Access	Initial value	Description
DisableLockSegment	BOOL	Get, Set	FALSE	Disables the use of the locking point segment for NC clamp axes.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
PtpSeq() [▶ 43]	Procedure for loading a PTP movement (extended)

Interfaces

Type	Description
I_Clamp	Standard interface on FB_Clamp

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

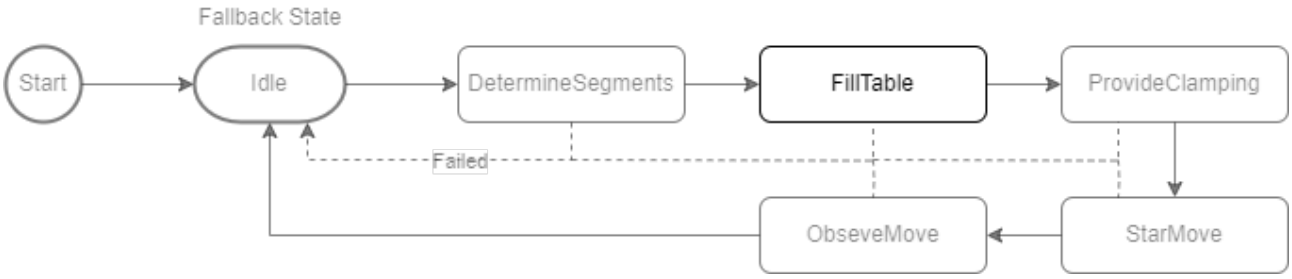
5.4.1.1 PtpSeq()

Clamp PTP variant only active with NC transformer axes

The modified state `eFillTable` is only used when using an NC transformer axis!

Instance type	Instance Name
Master	`fbPtpSeq`
Slaves	`aBaseSeqMembers[E_PtpBaseSeq.eFillTable]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.4.2 FB_Carriage

Implements additional functions for the operation of a carriage movement (e.g. using a crank drive).

Syntax:

```
FUNCTION_BLOCK FB_Carriage EXTENDS FB_PtpMotion
```

Methods

Name	Description
DynamicScaler()	Scales the axis dynamics based on the velocity in relation to the maximum velocity. The methods are implemented as PRIVATE. The call in the cycle method is activated via the properties of FB_CarriageHmi.
SmoothStart()	Enables startup with reduced dynamics in the range outside the end positions. The methods are implemented as PRIVATE. The call in the cycle method is activated via the properties of FB_CarriageHmi.

Interfaces

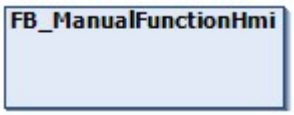
Type	Description
I_Carriage	Standard interface on FB_Carriage

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.5 ManualFunction - Manual functions

5.5.1 FB_ManualFunctionHmi



The class is used to connect to a TF8550 ManualOperation control.

Syntax:

```
FUNCTION_BLOCK FB_ManualFunctionHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Cmd [▶ 45]	I_ManualFunctionCmdHmi	Get	-	Commands from the HMI
State [▶ 46]	I_ManualFunctionStateHmi	Get	-	Feedback signals to the HMI

Interfaces

Type	Description
I_ManualFunctionHmi	Standard interface on FB_ManualFunctionHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.5.1.1 FB_ManualFunctionCmdHmi

Summarizes the commands of the HMI in FB_ManualFunctionHmi.

Syntax:

```
FUNCTION_BLOCK FB_ManualFunctionCmdHmi
```

Properties

Name	Type	Access	Initial value	Description
ToBasePos	BOOL	Get, Set	FALSE	Command to move to the base position
ToWorkPos	BOOL	Get, Set	FALSE	Command for controlling the working position

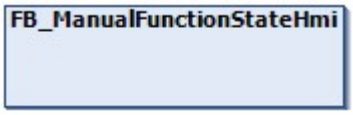
Interfaces

Type	Description
I_ManualFunctionCmdHmi	Standard interface on FB_ManualFunctionCmdHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.5.1.2 FB_ManualFunctionStateHmi



Summarizes the feedback signals of the HMI in FB_ManualFunctionHmi.

Internal functions:

- The `MovingToXy` properties set automatically when the respective signal of the FB_ManualFunctionCmd [▶ 45] is set

Syntax:

```
FUNCTION_BLOCK FB_ManualFunctionCmdHmi
```

Properties

Name	Type	Access	Initial value	Description
EnableBasePos	BOOL	Get, Set	FALSE	Enables the button of the base position.
EnableWorkPos	BOOL	Get, Set	FALSE	Releases the button of the working position.
FaultBasePos	BOOL	Get, Set	FALSE	Signals an error in the base position.
FaultWorkPos	BOOL	Get, Set	FALSE	Signals an error in the working position.
InBasePos	BOOL	Get, Set	FALSE	Signals that the base position has been reached.
InWorkPos	BOOL	Get, Set	FALSE	Signals that the working position has been reached.
MovingToBasePos	BOOL	Get, Set	FALSE	Signals the execution to the base position.
MovingToWorkPos	BOOL	Get, Set	FALSE	Signals the execution to the working position.

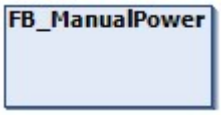
Interfaces

Type	Description
I_ManualFunctionStateHmi	Standard interface on FB_ManualFunctionStateHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.5.2 FB_ManualPower



Implements a predefined version of a manual function for switching on the axis control.

Syntax:

```
FUNCTION_BLOCK FB_ManualPower EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Enable	BOOL	Get, Set	FALSE	Releases the manual function.

Methods

Name	Description
Cyclic()	Cycle method
SetFeedEnableUse()	Configures the direction enable of the axis control.

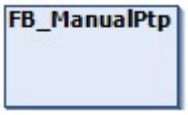
Interfaces

Type	Description
I_ManualFunction	Universal interface to a manual function

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.5.3 FB_ManualPtp



Implements a predefined version of a manual function for controlling a PTP axis.

Syntax:

```
FUNCTION_BLOCK FB_ManualPtp EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Enable	BOOL	Get, Set	FALSE	Releases the manual function.
TrigCmdBasePos	I_Trigger	Get	-	Trigger on the base position command
TrigCmdWorkPos	I_Trigger	Get	-	Trigger on the working position command

Methods

Name	Description
Cyclic()	Cycle method
SetWorkBasePos()	Sets the base and working position for indication on the HMI

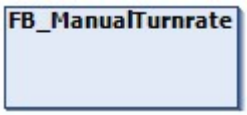
Interfaces

Type	Description
I_ManualFunction	Universal interface to a manual function

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.5.4 FB_ManualTurnrate



Implements a standard limit and display for tactile manual functions to adjust the turnrate of an axis.

Syntax:

```
FUNCTION_BLOCK FB_ManualTurnrate EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Enable	BOOL	Get, Set	FALSE	Releases the manual function.
CmdFaster	BOOL	Get	FALSE	Passes the command of the pressed button in the working position.
CmdSlower	BOOL	Get	FALSE	Passes the command of the pressed button in the base position.

Methods

Name	Description
Cyclic()	Cycle method
SetTurnrateLimits()	Configures the minimum and maximum number of revolutions and the use of the limit options.

Interfaces

Type	Description
I_ManualFunction	Universal interface to a manual function

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.6 Temperature - TF8540 Temperature Interface

5.6.1 FB_Temperature

Main class of temperature control to manage all temperature channels and groups.

Internal functions:

- Handling of the TcPfw temperature control (TF8540)
- Integrated data handling of grouping settings
- Management of individual sub-elements such as:
  - Temperature groups
  - Supply channels
  - Scheduler

Syntax:

```
FUNCTION_BLOCK FB_Temperature EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
DisableAlarms	BOOL	Get, Set	FALSE	Suppresses alarms triggered by errors in a temperature channel.
DisableMessages	BOOL	Get, Set	FALSE	Suppresses debug messages of the TF8540 library.
EnableCallback	BOOL	Get, Set	TRUE	Enables communication with the I/O components.
EnableConfig	BOOL	Get, Set	TRUE	Enables the configuration of all temperature channels.
EnableLooptest	BOOL	Get, Set	FALSE	Enables current monitoring of all zones.
LibScopeVars	REFERENCE TO FB_Scope_TempCtrlVariables	Get	-	Access to an overview of TF8540 live data.
Timer	I_TempSchedule	Get	NULL	Access to the connected scheduler.

Methods

Name	Description
Channels(x)	Returns an interface to the xth temperature channel of TF8540
CreateDefaultParams() [▶ 50]	Creates a default parameterization for all temperature channels
EnableAll() [▶ 51]	Enables all temperature channels on the PLC side.
Groups(x)	Returns an interface to the xth temperature group
LinkGroup() [▶ 52]	Assigns a linear arrangement of temperature channels to a group.
LinkSupply() [▶ 53]	Assigns a group to a supply channel.
LinkZone() [▶ 53]	Assigns a temperature channel to a group.
SetScheduler()	Assigns a schedule to the temperature control.
StandbyAll() [▶ 54]	Sets all temperature channels to standby.
Supply(x)	Returns an interface to the xth supply unit
SupplyLines(x)	Returns an interface to the xth supply channel
UnlinkGroup()	Removes all temperature channels from a group

Exceptions avoidance

The list access functions (e.g. Channels(x), Groups(x), etc...) return the first element (root) of the list if the index is invalidly requested.

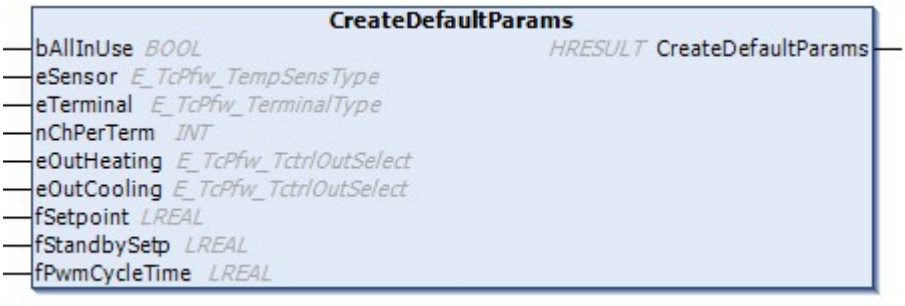
Interfaces

Type	Description
I_Temperature	Standard interface on FB_Temperature
I_TempTaskInterface	Runtime interface for a slow PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.1 CreateDefaultParams()



Creates a default parameterization for all temperature channels

Syntax:

```
METHOD CreateDefaultParams : HRESULT
VAR_INPUT
    bAllInUse:              BOOL;
    eSensor:                E_TcPfw_TempSensType;
    eTerminal:              E_TcPfw_TerminalType;
    nChPerTerm:             INT;
    eOutHeating:            E_TcPfw_TctrlOutSelect;
    eOutCooling:            E_TcPfw_TctrlOutSelect;
```

```
    fSetpoint:              LREAL;
    fStandbySetp:           LREAL;
    fPwmCycleTime:          LREAL;
END_VAR
```

Inputs

Name	Type	Description	Recommended standard
bAllInUse	BOOL	All channels are initialized as "InUse".	TRUE
eSensor	E_TcPfw_TempSensType	Sensor type - NoSensor in simulation mode	eTcPfwTempSensT_NoSensor
eTerminal	E_TcPfwTerminalType	Terminal type - NoTerminal in simulation mode	eTcPfwTermT_NoTerminal
nChPerTerm	INT	Number of channels per terminal	8
eOutHeating	E_TcPfw_TctrlOutSelect	Output type of the heating output - NoSignal, to disable the heating function (e.g. for measuring zones)	eTcPfwTcOut_PWM
eOutCooling	E_TcPfw_TctrlOutSelect	Cooling output type	eTcPfwTcOut_NoSignal
fSetpoint	LREAL	Temperature setpoint for all channels	180.0
fStandbySetp	LREAL	Temperature setpoint for standby temperature of all channels	18.0
fPwmCycleTime	LREAL	PWM cycle time for all outputs (dutyCycle = fPwmCycleTime * 0.1)	1.0

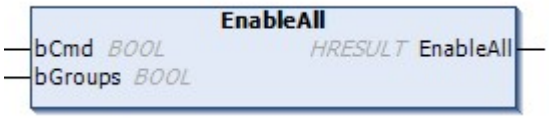
Outputs

Name	Type	Description
CreateDefaultParams	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.2 EnableAll()



Enables all temperature channels on the PLC side.

Syntax:

```
METHOD EnableAll
VAR_INPUT
    bCmd:       BOOL;
    bGroups:    BOOL;
END_VAR
```

Inputs

Name	Type	Description
bCmd	BOOL	TRUE to grant the release, FALSE to withdraw the release.
bGroups	BOOL	The enable only takes into account channels that are assigned to a group.

Outputs

Name	Type	Description
EnableAll	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.3 LinkGroup()

Assigns a number of temperature channels to a group.

Syntax:

```
METHOD LinkGroup : HRESULT
VAR_INPUT
    nStartIdx:          INT;
    nEndIdx:            INT;
    nGroupIdx:          INT;
    bOverwrite:         BOOL;
END_VAR
```

Inputs

Name	Type	Description
nStartIdx	INT	Index of the first channel to be assigned
nEndIdx	INT	Index of the last channel to be assigned
nGroupIdx	INT	Index of the group to which the channels are to be assigned
bOverwrite[1]	BOOL	Zones are assigned even if the group contains already assigned zones.

[1] Obsolete and will be ignored

Outputs

Name	Type	Description
LinkGroup	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.4 LinkSupply()

Assigns a supply unit to a group.

Syntax:

```
METHOD LinkSupply : HRESULT
VAR_INPUT
    nGroupIdx:              INT;
    nSupplyIdx:             INT;
    nLine:                  INT;
END_VAR
```

Inputs

Name	Type	Description
nGroupIdx	INT	Index of the group to which a supply unit is to be assigned
nSupplyIdx	INT	Index of the supply unit to be assigned to the group
nLine	INT	Supply channel to which the group members are connected • 1; 2; 3 - phase L1, L2 or L3 • 4 – Between phases without connection to N

Outputs

Name	Type	Description
LinkSupply	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.5 LinkZone()

Assigns a single temperature channel (zone) to a group.

Syntax:

```
METHOD LinkZone : HRESULT
VAR_INPUT
    nLinearIdx:          INT;
    nGroupIdx:           INT;
    nGroupMemberIdx:     INT;
```

```
    bOverwrite:         BOOL;
END_VAR
```

Inputs

Name	Type	Description
nLinearIdx	INT	Index of the channel to be assigned
nGroupIdx	INT	Index of the group to which the channel is to be assigned
nGroupMemberIdx[1]	INT	*Index in the target group*
bOverwrite[1]	BOOL	Zone is assigned even if the index is already occupied in the target group.

[1] Obsolete, will be ignored

Outputs

Name	Type	Description
LinkZone	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.1.6 StandbyAll()

Sets all temperature channels to standby temperature.

Syntax:

```
METHOD StandbyAll : HRESULT
VAR_INPUT
    bCmd:       BOOL;
    bGroups:    BOOL;
END_VAR
```

Inputs

Name	Type	Description
bCmd	BOOL	TRUE to enable the standby temperature, FALSE to disable.
bGroups	BOOL	The function only considers channels that are assigned to a group.

Outputs

Name	Type	Description
StandbyAll	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.2 FB_TemperatureHmi

HMI parallel class to the `FB_Temperature`.

Syntax:

```
FUNCTION_BLOCK FB_TemperatureHmi EXTENDS FB_BaseMdHmi
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	A channel (InUse = TRUE) has exceeded the absolute temperature maximum.
AlarmAbsoluteLow	BOOL	Get	FALSE	One channel (InUse = TRUE) has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	At least one channel with active control has exceeded the HighHigh tolerance.
AlarmHigh	BOOL	Get	FALSE	At least one channel with active control has exceeded the High tolerance.
AlarmLow	BOOL	Get	FALSE	At least one channel with active control has fallen below the Low tolerance.
AlarmLowLow	BOOL	Get	FALSE	At least one channel with active control has fallen below the LowLow tolerance.
CountPfwChannels	LREAL	Get	20.0	Number of available TF8540 temperature channels
ParamTempSupply	REFERENCE TO ARRAY[] OF FB_MdTempSupplyHmi	Get	-	Parameter interface for parameterization via the HMI
ParamTempZone	REFERENCE TO ARRAY[] OF FB_MdTempZoneHmi	Get	-	Parameter interface for parameterization via the HMI
TempAmbient	LREAL	Get, Set	18.0	Standard ambient temperature (for simulation)

Methods

Name	Description
Groups(x)	Returns an interface to the xth temperature group (HMI class)

Exceptions avoidance

The list access functions (e.g. Channels(x), Groups(x), etc...) return the first element (root) of the list if the index is invalidly requested.

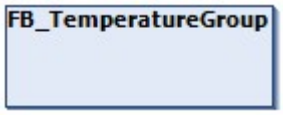
Interfaces

Type	Description
I_TemperatureHmi	Standard interface on FB_TemperatureHmi
I_Recipe	Interface for managing structured recipe values

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.3 FB_TemperatureGroup



Class for group control of the temperature channels.

Syntax:

```
FUNCTION_BLOCK FB_TemperatureGroup EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the absolute maximum temperature.
AlarmAbsoluteLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the HighHigh tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the High tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the Low tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLowLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the LowLow tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmNoResponse	BOOL	Get	FALSE	Temperature value of the group (at least one channel) does not respond to the control.
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
ConfigHash	T_SHA256	Get	0x0	Hash value of the current group configuration
ConfigID	UINT	Get	0	ID of the group configuration (incremental)
EnableLimitAlarms	BOOL	Get, Set	FALSE	Exceeding a tolerance value/ limit triggers an alarm.
Fault	BOOL	Get	FALSE	Group (at least one channel) has an error.
Index	INT	Get	0	Index (ID) of the group
IsActive	BOOL	Get	FALSE	Group (at least one channel) is actively controlled.
IsEnabled	BOOL	Get	FALSE	All channels of the group are enabled.
IsStandby	BOOL	Get	FALSE	All channels of the group are in standby mode.
LoadHash	T_SHA256	Get	0x0	Hash value of the last configuration loaded from a file
ZonesCount	INT	Get	0	Number of zones in the group

Methods

Name	Description
EnablePLC()	Enables all zones of the group on the PLC side.
Force()	Forces all zones of the group to heating/cooling 100%.
UpdateGroup()	Updates the internal listing of the assigned groups
Zones(x)	Returns an interface to the xth zone of the group

Exceptions avoidance

The list access functions (e.g. Channels(x), Groups(x), etc...) return the first element (root) of the list if the index is invalidly requested.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Type	Description
GroupStates() [▶ 59]	Procedure for loading, executing and saving tuning

Possible events

ID	Description	Alarm/Message
401x	Temperature operation monitoring	Alarm
402x	Tuning parameters loading process	Alarm
403x	Tuning parameters saving process	Message

Interfaces

Type	Description
I_TemperatureGroup	Standard interface on FB_TemperatureGroup
I_AttachableMdInterface	Interface for containerless machine data components

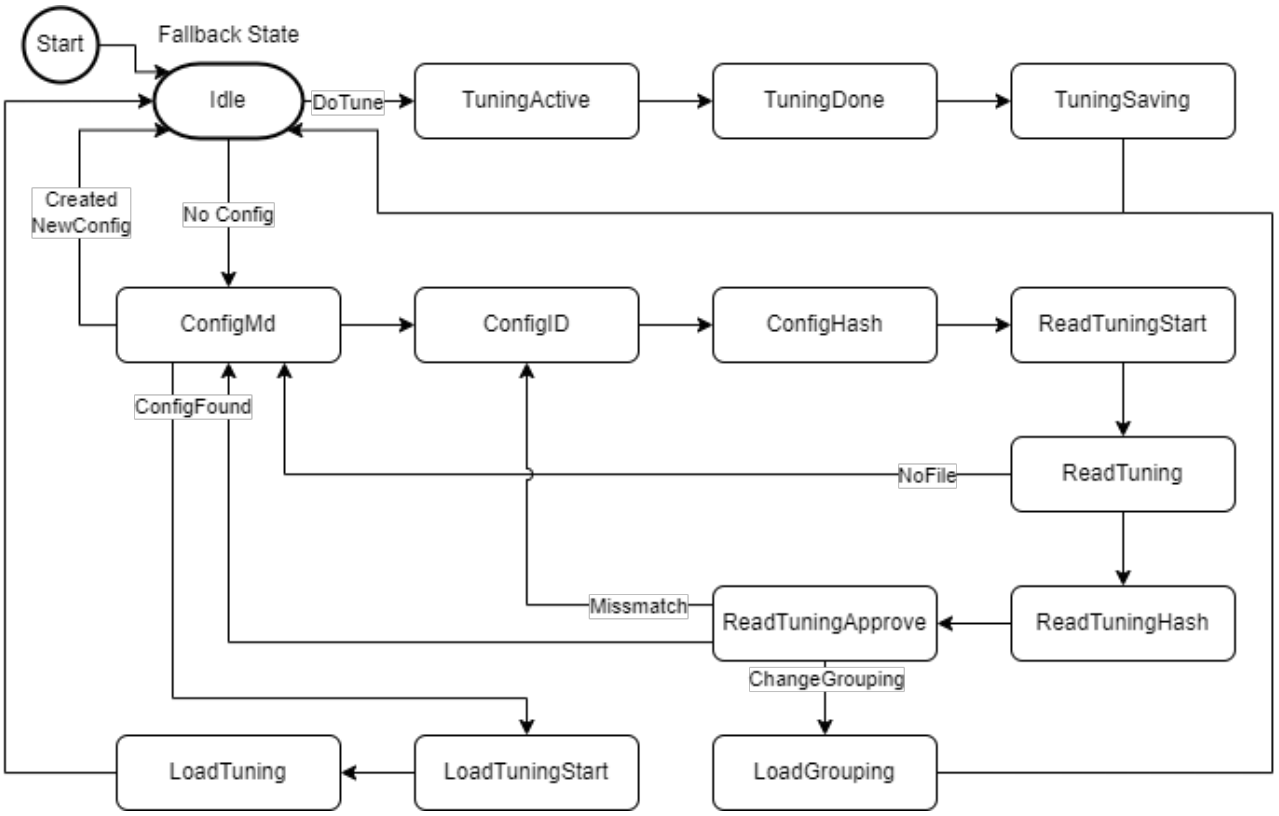
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.3.1 GroupStates()

Instance type	Instance Name
Master	`fbGroupStates`
Slaves	`aBaseSeqMembers[E_GroupStates.eLength]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.4 FB_TemperatureGroupHmi

HMI parallel class to the `FB_TemperatureGroup` class.

Syntax:

```
FUNCTION_BLOCK FB_TemperatureGroupHmi EXTENDS FB_BaseMdHmi
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the absolute maximum temperature.
AlarmAbsoluteLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the HighHigh tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmHigh	BOOL	Get	FALSE	Group (at least one channel) has exceeded the High tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the Low tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLowLow	BOOL	Get	FALSE	Group (at least one channel) has fallen below the LowLow tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmState	PlasticStatusHmi	Get	0	Alarm Status for display with a TcHMI StateIndicator
BootAsScheduled	BOOL	Get, Set	FALSE	The temperature group is to start in the scheduled operation mode
ConfigHash	T_SHA256	Get	0x0	Hash value of the loaded group configuration
ConfigNote	STRING(127)	Get, Set	‘‘	Note on the loaded group configuration
DoTune	BOOL	Get, Set	FALSE	Starts tuning of all active zones of the group.
GroupName	STRING	Get, Set	‘‘	Temperature group name
Index	INT	Get	0	Index (ID) of the group
IsOff	BOOL	Get	FALSE	Group is switched off
IsOn	BOOL	Get	FALSE	Group is switched on
IsScheduled	BOOL	Get	FALSE	Group is in scheduled operation mode
IsStandby	BOOL	Get	FALSE	Group is in standby mode
SetOff	BOOL	Get, Set	FALSE	Switch off group
SetOn	BOOL	Get, Set	FALSE	Switch on group
SetScheduled	BOOL	Get, Set	FALSE	Set group to scheduled mode
SetStandby	BOOL	Get, Set	FALSE	Set group to standby mode
TuningActive	BOOL	Get	FALSE	The tuning of the group is active

Name	Type	Access	Initial value	Description
TuningDone	BOOL	Get	FALSE	The tuning of the group is completed.
TuningFailed	BOOL	Get	FALSE	The tuning of the group has failed
ZonesCount	INT	Get	0	Number of zones in the group

Methods

Name	Description
CollectRemainingSave()	[INTERNAL] Queries whether a property to be saved persistently has been set (.Set)
UpdateState()	[INTERNAL] Sets the active operation mode

Interfaces

Type	Description
I_TemperatureGroupHmi	Standard interface on FB_TemperatureGroupHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.5 FB_TempChannelBase

Base class for a single temperature channel

Syntax:

```
FUNCTION_BLOCK FB_TempChannelBase EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Index	INT	Get, Set	0	References a PlasticBaseApplication channel to a TF8540 zone.

Class contains significantly more properties than listed

The properties of the FB_TempChannelBase class overlap with the TF8540 Global Variables aaaPfwTempToHmi, aaaPfwTempMparamFromHmi, aaaPfwTempPparamFromHmi and stPfwTempCtrl. For more information on overlapping features, it is recommended to use the TF8540 documentation.

Interfaces

Name	Description
I_TempZoneHmi	Interface compatible with FB_TempZoneHMI.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.6 FB_TempChannel

Class for single control of a temperature channel.

Syntax:

```
FUNCTION_BLOCK FB_TempChannel EXTENDS FB_TempChannelBase
```

Properties

Name	Type	Access	Initial value	Description
AlarmNoResponse	BOOL	Get	FALSE	Temperature value of the channel does not respond to the control.
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
ConfigID	UINT	Get, Set	0	ID of the active group configuration
EnableLimitAlarms	BOOL	Get, Set	FALSE	Exceeding a tolerance value/ limit triggers an alarm.
Error	BOOL	Get	FALSE	Alias to 'Fault'
ExtruderID	INT	Get, Set	0	ID of the extruder used
Fault	BOOL	Get	FALSE	An error has occurred in this temperature channel
GroupID	INT	Get, Set	0	ID of the assigned group
IsActive	BOOL	Get	FALSE	Channel is actively controlled
IsEnabled	BOOL	Get	FALSE	Channel is enabled
IsLinked	BOOL	Get	FALSE	Channel is assigned to a group
IsStandby	BOOL	Get	FALSE	Channel is in standby mode
SupplyID	INT	Get	0	ID of the supply channel used
TuningLastExecution	DATE_AND_TIME	Get, Set	DT#1900-01-01T00:00:00Z	Date of the last successful execution of a tuning
TuningRequired	BOOL	Get, Set	TRUE	The current configuration requires autotuning

Methods

Type	Description
EnablePLC()	Enables the temperature channel on the PLC side
Force()	Forces the temperature channel to heating/cooling 100%.
Standby()	Sets the temperature channel to standby

Possible events

ID	Description	Alarm/Message
400x	Temperature operation monitoring	Alarm

Interfaces

Type	Description
I_TempChannel	Standard interface on FB_TempChannel
I_AttachableMdInterface	Interface for containerless machine data components
I_TempZone	Compatible interface with FB_TempZone

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.7 FB_TempChannelHmi

HMI parallel class to the `FB_TempChannel` class

Syntax:

```
FUNCTION_BLOCK FB_TempChannelHmi EXTENDS FB_TempChannelBase
```

Properties

Name	Type	Access	Initial value	Description
GroupID	INT	Get, Set	0	ID of the assigned group

Interfaces

Type	Description
I_TempChannelHmi	Standard interface on FB_TempChannelHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.8 FB_TempRecipe

Recipe management class of temperature control.

Syntax:

```
FUNCTION_BLOCK FB_TempRecipe EXTENDS FB_Recipe
```

Local HMI variables

Name	Data type	Description
Setpoint	LREAL	Temperature set point
Standby	LREAL	Temperature standby setpoint
ThresholdM	LREAL	Negative inner tolerance
ThresholdMM	LREAL	Negative external tolerance
ThresholdP	LREAL	Positive inner tolerance
ThresholdPP	LREAL	Positive external tolerance

Methods

Name	Description
Init(ipChannel)	Initialization for linking with a temperature channel

Interfaces

Type	Description
I_Recipe	Standard interface on FB_Recipe.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.9 FB_TempSupply

Implements a supply unit of temperature control

Syntax:

```
FUNCTION_BLOCK FB_TempSupply EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
Index	INT	Get, Set	0	Supply unit index

Methods

Type	Description
Line(x)	Returns the xth supply channel of the supply group

Exceptions avoidance

The list access functions (e.g. Channels(x), Groups(x), etc...) return the first element (root) of the list if the index is invalidly requested.

Interfaces

Type	Description
I_TempSupply	Standard interface on FB_TempSupply

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.10 FB_TempSupplyLine

Implements a supply unit of temperature control

Syntax:

```
FUNCTION_BLOCK FB_TempSupplyLine EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
ActSupplyCurrent	LREAL	Get	0.0	Actual current of the channel
ActSupplyLoad	LREAL	Get	0.0	Actual output of the channel
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
CalcErrorCurrent	LREAL	Get	0.0	Calculated error current of the channel
CalcSupplyLoad	LREAL	Get	0.0	Calculated output of the channel
CalcSupplyMatch	LREAL	Get	0.0	Deviation of the calculated and actual output of the channel
Frequency	LREAL	Get	0.0	Channel frequency
PwmCycleTime	LREAL	Get, Set	0.0	Cycle time of the PWM signal
PwmFactorC	INT	Get, Set	0	Factor PWM cycle time from cooling to heating
PwmMaxOnTime	LREAL	Get, Set	0.1	Maximum PWM switch-on time
PwmMaxOnTimeC	LREAL	Get, Set	0.1	Maximum PWM switch-on time of the cooling system
PwmMaxRampLoad	LREAL	Get, Set	0.0	Reserved
PwmMinOnTime	LREAL	Get, Set	0.0	Minimum PWM switch-on time
SupplyID	INT	Get, Set	0	ID of the channel

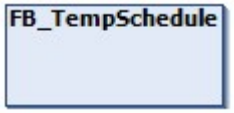
Interfaces

Type	Description
I_TempSupplyLine	Standard interface on FB_TempSupplyLine
I_AttachableMdInterface	Interface for containerless machine data components

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.11 FB_TempSchedule



Class for the use of weekday timers in connection with temperature control

Syntax:

```
FUNCTION_BLOCK FB_TempSchedule EXTENDS FB_TimerWeekdayMaster
```

Methods

Type	Description
MemberSchedule(x)	Returns an interface to the first timer assigned to the group (member) x
MemberScheduledActive(x)	Returns an interface to the first timer assigned to the group (member) x and currently active

Exceptions avoidance

The member access functions return a dummy instance in the absence of an overlap with member x.

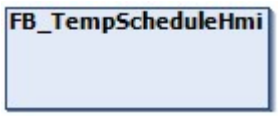
Interfaces

Type	Description
I_TempSchedule	Standard interface on FB_TempSchedule
I_OneTaskInterface	Runtime interface for a PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.12 FB_TempScheduleHmi



HMI parallel class to FB_TempSchedule class.

Syntax:

```
FUNCTION_BLOCK FB_TempScheduleHmi EXTENDS FB_BaseMdHmi
```

Properties

Name	Type	Access	Initial value	Description
Timer [1]	REFERENCE TO ARRAY[] OF FB_TimerWeekdayHmi	Get	-	Interface to the individual dates of the schedule

[1] Obsolete

Methods

Type	Description
Timers(x)	Returns an interface to the xth timer

Exceptions avoidance

The list access functions (e.g. Channels(x), Groups(x), etc...) return the first element (root) of the list if the index is invalidly requested.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.13 FB_TimerTempHmi

Extension of `FB_TimerWeekdayHmi` with additional temperature-relevant attributes.

Syntax:

```
FUNCTION_BLOCK FB_TimerTempHmi EXTENDS FB_TimerWeekdayHmi
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
MembersCount	INT	Get	0	Number of assigned members (groups)
Standby	BOOL	Get, Set	FALSE	Timer is to be evaluated as standby operation

Methods

Type	Description
Clear()	Empties the list of members (groups)
Exists(x)	Checks whether member (group) x is assigned to the timer
Members() [▶ 70]	Returns an interface to the member
Subscribe(x)	Makes group x become a member of the timer

Interfaces

Type	Description
I_TimerTempHmi	Standard interface on FB_TimerTempHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.6.13.1 Members()

Returns an interface to the idx member (group)

Syntax:

```
METHOD Members : I_TemperatureGroup
VAR_INPUT
    idx:        INT;
    iTemp:      I_Temperature;
END_VAR
```

Inputs

Name	Type	Description
Idx	INT	Index of the member in the list all members
iTemp	I_Temperature	Reference of the temperature control to determine the temperature group instance

Outputs

Name	Type	Description
Members	I_TemperatureGroup	Requested temperature group

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.7 Recipe - recipe management

The recipe management of the TF8550 Recipe Helper is based on the implementation of the FB_PlcStateToHmi [▶ 71] class and the implemented handshake procedure. For this purpose, the class is already instantiated in the GVL `HmiCommunication`.

```
{attribute 'qualified_only'}
VAR_GLOBAL

// ============================================
// communication PLC <> HMI

    fbPlcStateToHmi:                    FB_PlcStateToHmi;

// ============================================
```

```
END_VAR
```

5.7.1 FB_PlcStateToHmi

Implements the handshake procedure for the TF8560 recipe management on the PLC side.

Syntax:

```
FUNCTION_BLOCK FB_PlcStateToHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description	PLC / HMI[1]
AxesInitialised	BOOL	Get, Set	FALSE	Axes have been initialized, the recipe may be initialized.	PLC
ClientID	UINT	Get, Set	0	ID of the currently connected HMI client	HMI
DataReqFailed	BOOL	Get, Set	FALSE	Errors in communication. The PLC is waiting for a reset.	PLC
DataRequestPlc	BOOL	Get, Set	FALSE	Requesting data during the start-up phase or a product change.	PLC
DataRequestPlcActiveHmi	BOOL	Get, Set	FALSE	Writing recipe data to the PLC active.	HMI
DataRequestPlcQuitHmi	BOOL	Get, Set	FALSE	Writing of recipe data to the PLC completed.	HMI
DataValidPlc	BOOL	Get, Set	FALSE	The recipe data has been loaded completely.	PLC
LiveSignHMI	UINT	Get, Set	0	Changing value of the HMI client to signal an active connection	HMI
PlcInitialized	BOOL	Get, Set	FALSE	Alias to AxesInitialized	PLC
ProductChangeConfirmPlc	BOOL	Get, Set	FALSE	The request to change the product is accepted.	PLC
ProductChangeEnable	BOOL	Get, Set	FALSE	Signals the possibility to change the recipe to the HMI.	PLC
ProductRequestHmi	BOOL	Get, Set	FALSE	Requests of a product change	HMI
Reset	BOOL	Get, Set	FALSE	Request for reinitialization of the recipe	HMI
SaveDataQuitPlc	BOOL	Get, Set	FALSE	Confirms the processing of a recipe storage.	HMI
SaveDataRequestPlc	BOOL	Get, Set	FALSE	Request to save the recipe	PLC
VersionBaseApplication	STRING	Get	'v0.0.0.0'	Plastic Base Application version	PLC
VersionPalsticFunctions	STRING	Get	'v0.0.0.0'	Plastic Technology Functions version (TF8560)	PLC

[1]The PLC/HMI column describes the assignment of the write access.

Methods

Name	Description
DeclareBeta()	Adds a beta label to the display version of the Plastic Base Application.

Interfaces

Type	Description
I_RecipeState	Interface for the handshake variables to be processed in the main procedure

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.7.2 I_RecipeState

Interface to the PLC variables of the TF8550 Beckhoff.TwinCAT.HMI.Plastic.RecipeHelper class `FB_PlcStateToHmi`

Syntax:

```
INTERFACE I_RecipeState EXTENDS I_BaseEmpty
```

Properties

Name	Type	Access	Initial value	Description
DataRequestPlc	BOOL	Get	FALSE	Requesting data during the start-up phase or a product change.
DataValidPlc	BOOL	Get	FALSE	The recipe data has been loaded completely.
PlcInitialized	BOOL	Get, Set	FALSE	Data has been initialized, the recipe may be initialized.
ProductChangeEnable	BOOL	Get, Set	FALSE	Signals the possibility to change the recipe to the HMI.
Reset	BOOL	Get, Set	FALSE	Request for reinitialization of the recipe

Methods

Name	Description
DeclareBeta()	Adds a beta label to the display version of the Plastic Base Application.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.7.3 FB_Recipe

Recipe management class for summarizing structured recipe values. These are used in integrated TcHMI controls.

Class is abstract

Since the class is defined as ABSTRACT, the class cannot be instantiated and must be implemented using inheritance.

Syntax:

```
FUNCTION_BLOCK ABSTRACT FB_Recipe EXTENDS FB_Base
```

Methods

Name	Description
Lock()	Locks the recipe variables • The next execution of .Update() writes the recipe values to the PLC • Can be called/executed from the HMI
Reset()	Resets the lock state
Update()	[ABSTRACT] Updates the recipe variables to the current PLC value • Can be called/executed from the HMI

Interfaces

Type	Description
I_Recipe	Standard interface on FB_Recipe.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.8 MachineData - Machine data

Base

The \MachineData\Base\ subfolder contains the base classes of the machine data handling.

Classes

Name	Description
FB_MdBaseContainer [▶ 74]	Represents the base container for machine data.
FB_MdBaseComponent [▶ 76]	Represents the base class for machine components.

Components

The \MachineData\Components\ subfolder contains all the machine data components that have already been implemented. All included components inherit from the FB_MdBaseComponent class and have an enumeration that defines the index (without offset) of the individual parameters.

Classes

Name	Use	Description
FB_MdFileInfo	FB_MdBaseContainer	Contains the information of a saved file.
FB_MdAnalogValue	FB_Monitoring FB_Setpoints FB_Blowing	Contains scaling parameters for analog values .
FB_MdBlowpin	FB_Blowpin	Contains blowpin-specific parameters.
FB_MdClamp	FB_Clamp	Contains clamp-specific parameters.
FB_MdWtc	FB_Wtc	Contains Wtc-specific parameters.
FB_MdAxis	FB_Axis	Contains general axis parameters.
FB_MdContinuousMotion	FB_Extruder	Contains parameters for axes with continuous rotary motion.
FB_MdHoming	FB_Homing	Contains setting parameters for homing procedures.
FB_MdNc	FB_NcFunctions	Contains NC-specific parameters.
FB_MdPtpMotion	FB_PtpMotion	Contains parameters for PTP-based motion.
FB_MdTrafo	FB_TrafoFunctions	Contains parameters for transforming axes.
FB_MdTempSupply	FB_TempSupply	Contains parameters for a temperature supply unit.
FB_MdTempZone	FB_TempZone	Contains parameters for a temperature zone.
FB_MdWeekdayTiming	FB_TimerWeekdayHmi	Contains data about the TimeSchedule of the Weekday Timer.

Create your own machine data components

The standard of TF8560 machine data applies for creating your own machine data components. A new component must implement the abstract methods MdNextParameter() and MdSetParameter() and set some internal variables. For this it is recommended to use the existing classes as a template and to follow the instructions of the TF8560 documentation.

Hmi

In the \MachineData\Hmi\ subfolder, classes are defined that summarize the data stored per component for the HMI. Here, only reference accesses (interfaces) are used and no data is copied locally into the class. Among other things, these are used in the HMI on the configuration pages.

Subitems

The \MachineData\Subitems\ subfolder contains several classes, interfaces and enumerations that contribute to the functionality of the machine data classes.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.8.1 FB_MdBaseContainer

The class extends the TF8560 FB_MdContainer by the machine data encoding of the TwinCAT 3 Plastic Application. In addition, errors that occur can be evaluated using Boolean values and file storage is simplified.

Syntax:

```
FUNCTION_BLOCK FB_MdBaseContainer EXTENDS FB_MdContainer
```

Properties

Name	Type	Access	Initial value	Description
AllowFolderCreation	BOOL	Get, Set	TRUE	Allows the container to create the folder structure for storing the machine data.
CrcInvalid	BOOL	Get	FALSE	The error that occurred was caused by an incorrect CRC checksum.
CreationDate	DATE_AND_TIME	Get	DT#1970-1-1-0:0:0	Date of the first creation of a file
Details	I_MdBaseContainerDetails	Get	THIS^	Summarizes the properties of the file details.
ErrorDetails	I_MdBaseContainerErrors	Get	THIS^	Summarizes the properties of the error information.
ErrorId	UDINT	Get	0	Error ID of the last occurred error
FileAccessDenied	BOOL	Get	FALSE	The error that occurred was caused by missing file access rights.
FileNotfound	BOOL	Get	FALSE	The error that occurred was caused by the absence of the file.
IgnoreMissmatches	BOOL	Get, Set	FALSE	Forces the container to load a file despite version collision.
MissmatchDetected	BOOL	Get	FALSE	The error that occurred was caused by a version conflict.
MissmatchBaseApp	BOOL	Get	FALSE	The version conflict that has occurred is in the versions of the TwinCAT Base Application library.
MissmatchIdxFormat	BOOL	Get	FALSE	The version of the index coding causes the version conflict that occurred.
StoreCount	UDINT	Get	0	Counts the iterations of file write operations since the file was created.
StoreDate	DATE_AND_TIME	Get	DT#1970-1-1-0:0:0	Date of the last saving process
UnknownParameter	BOOL	Get	FALSE	The occurred error was caused by an unknown parameter in the file.
VersionBaseApp	I_LibVersion	Get	-	Version of the Plastic Base Application library
VersionIdxFormat	I_IdxFormatVersion	Get	-	Index coding version

Methods

Name	Description
AddComponent()	Adds another component to the container.
CompareFileVersion()	[INTERNAL] Compares the passed versions with the defined version in the source code.
CreateFilepath()	Creates a new file path based on the container name
OverwriteFromFile()	[INTERNAL] Used by the FB_MdFileInfo class to update the loaded version in the container.

Interfaces

Type	Description
I_MdBaseContainer	Standard interface on FB_MdBaseContainer
I_MdBaseContainerDev	Extended interface with access to the methods marked as "[INTERNAL]"
I_MdBaseContainerDetails	Interface to the properties with file-related information (e.g. CreationDate)
I_MdBaseContainerErrors	Interface to the properties with error information

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.5)

5.8.2 FB_MdBaseComponent

The class extends the standard component FB_MdComponent of the TF8560 utilities by functions of the index coding of the Plastic Base Application.

Syntax:

```
FUNCTION_BLOCK FB_MdBaseComponent EXTENDS FB_MdComponent
```

Properties

Name	Type	Access	Initial value	Description
LinkedContainer	I_MdBaseContainer	Get	NULL	Points to the container to which the component was appended.

Methods

Name	Description
ConfigCompType()	Used with inheriting classes to initialize the encoding offset to E_StandardCompType.
ConfigCompTypeEx()	Used with the inheriting class to initialize the encoding offset.

Both methods are declared as **PROTECTED** and can only be used within the class.

Interfaces

Type	Description
I_MdBaseComponent	Standard interface on FB_MdBaseComponent

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.8.3 FB_MdCollection

The class can store a collection of components of a class type. When attaching the class to `FB_BaseRuntime`, components of objects implementing the interface `I_AttachableMdInterface` can be collected and attached to the collection.

Syntax:

```
FUNCTION_BLOCK FB_MdCollection EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
MachineData	I_MdBaseContainer	Get	-	Interface to the internal machine data container

Methods

Name	Description
AddComponent()	Adds a component to the collection: • Automatically increments `.ComponentIndex` of the attached components • Refuses components that do not match the prototype
CheckType()	[PROTECTED] Returns an ID used among the supported types
SetPrototype()	Fixes the type of the component to be collected by the runtime

Interfaces

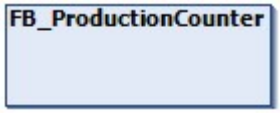
Type	Description
I_MdCollection	Standard interface on FB_MdCollection
I_OneTaskInterface	Runtime interface for a PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.9 OperationData - Production data and statistics

5.9.1 FB_ProductionCounter



Implements a general counter for production-dependent data such as a piece counter or a production time counter.

Syntax:

```
FUNCTION_BLOCK FB_ProductionCounter EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
SavingInterval	LREAL	Get, Set	10.0	Interval for saving the counter value in a machine data file
SinceLastReset	I_ProductionDataComponent	Get	-	Counter value since the last execution of the Reset() method
SinceProducing	I_ProductionDataComponent	Get	-	Counter value since the start of the current counting process
SinceStart	I_ProductionDataComponent	Get	-	Counter value since machine start
Total	I_ProductionDataComponent	Get	-	Counter value since the beginning of the machine production time

Methods

Name	Description
NewPart()	Logs a new part for the part counter.
Producing()	Indicates that production is actively running.

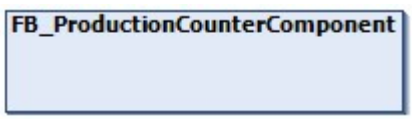
Interfaces

Type	Description
I_ProductionCounter	Standard interface on FB_ProductionCounter

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.9.2 FB_ProductionCounterComponent



Contains the counter data for a defined period.

Syntax:

```
FUNCTION_BLOCK FB_ProductionDataComponent EXTENDS FB_MdBaseComponent
```

Properties

Name	Type	Access	Initial value	Description
OperationTime	LREAL	Get, Set	0.0	Production time
Parts	ULINT	Get, Set	0	Number of items
PartsPerHour	LREAL	Get, Set	0	Parts per hour
TimePerCycle	LREAL	Get, Set	0.0	Production cycle time
ComponentType	USINT	Get	0	Type of component as identifier
Connected	BOOL	Get	FALSE	Component is connected with its access dependencies.

Methods

Name	Description
Connect()	Connects references to the class.

Event-driven methods (callback methods)

Name	Description
MdNextParameter()	Container call to save the parameters to a file
MdSetParameter()	Container call to load the parameters into the runtime

Interfaces

Type	Description
I_ProductionCounterComponent	Standard interface on FB_ProductionCounterComponent

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.10 EventLogger - logging of events and errors

5.10.1 FB_AlarmHandler

The class implements a handling of pending alarms to influence the process sequence. This means, for example, that an alarm with the severity "Error" can be responded to by stopping the machine directly.

Syntax:

```
FUNCTION_BLOCK FB_AlarmHandler EXTENDS FB_ListenerBase2
```

Properties

Name	Type	Access	Initial value	Description
AlarmCritical	BOOL	Get	FALSE	A critical alarm is present.
AlarmError	BOOL	Get	FALSE	There is an error alarm.
AlarmWarning	BOOL	Get	FALSE	There is a warning alarm.
AlarmInfo	BOOL	Get	FALSE	There is an information alarm pending.
Error	BOOL	Get	FALSE	The handler is in an error state.

Methods

Name	Description
ClearAlarms()	Confirms and clears all pending alarms.

Interfaces

Type	Description
I_AlarmHandler	Standard interface on FB_AlarmHandler
I_OneTaskInterface	Runtime interface for a PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.10.2 FB_AppMessage

This class is part of the FB_Base class and provides an interface to the EventLogger for almost every TC3 Plastic Base Application object.

Syntax:

```
FUNCTION_BLOCK EXTENDS FB_Message
```

Methods

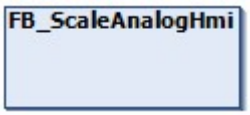
Name	Description
ClearAlarm()	Clears a specific alarm.
ConfirmAlarm()	Acknowledges a specific alarm.
Reset()	Resets all alarms generated by this instance.
SendAlarm()	Triggers an alarm.
SendEqualMessage()	Logs a message in the EventLogger.
SendHresult()	Logs a message in the EventLogger with the hexadecimal representation of the error codes.
SendMessage()	Logs a message in EventLogger, unless the message has been sent before.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.11 Analog - analog value scaling

5.11.1 FB_ScaleAnalogHmi



Implements functions for scaling analog values with parameterization via the HMI.

Syntax:

```
FUNCTION_BLOCK FB_ScaleAnalogHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
ParamAnalogValue	REFERENCE TO FB_MdAnalogValueHmi	Get	-	Interface for parameterization of analog value scaling
ScalingElectricalMax	LREAL	Get, Set	10.0	Maximum of the electrical voltage input
ScalingElectricalMin	LREAL	Get, Set	0.0	Minimum of the electrical voltage input
ScalingRawMax	LREAL	Get, Set	32767	Maximum of the raw analog measured value (bit)
ScalingRawMin	LREAL	Get, Set	0	Minimum of the raw analog measured value (bit)
ScalingValueMax	LREAL	Get, Set	100.0	Maximum of the scaled end unit
ScalingValueMin	LREAL	Get, Set	0.0	Minimum of the scaled end unit

Methods

Name	Description
ElectricalToRaw()	Scales the electrical voltage to the raw analog measured value (bit).
ElectricalToValue()	Scales the electrical voltage to the unit of the final scaling.
RawToElectrical()	Scales the raw analog measured value (bit) to the electrical voltage.
RawToValue()	Scales the raw analog measured value (bit) to the unit of the final scaling.
ValueToElectrical()	Scales the unit of the final scaling to the electrical voltage.
ValueToRaw()	Scales the unit of the final scaling to the raw analog measured value (bit).

Interfaces

Type	Description
I_ScaleAnalogHmi	Standard interface on FB_ScaleAnnalogHmi
I_ScaleAnalogScalings	Interface to the scaling minima and maxima

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.11.2 FB_Monitoring

Implements a monitoring function for analog signals. The threshold values for monitoring can be set variably on the HMI.

Syntax:

```
FUNCTION_BLOCK FB_Monitoring EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
AutoClearAlarms	BOOL	Get, Set	FALSE	Triggered alarms are automatically cleared after the cause has been eliminated.
EnableAlarms	BOOL	Get, Set	FALSE	Falling below/exceeding the limit triggers pre-implemented alarms.
Input	I_InputBase	Get, Set	NULL	Interface of the analog signal to be read
Value	LREAL	Get, Set	0	Returns the scaled analog value. Can be set if no input has been assigned.

Possible events

ID	Description	Alarm/Message
300x	Exceeding/falling below the set tolerances	Alarm

Interfaces

Type	Description
I_Monitoring	Standard interface on FB_Monitoring
I_AttachableMdInterface	Interface for containerless machine data components
I_OneTaskInterface	Runtime interface for a PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.11.2.1 FB_MonitoringTemp

Implements monitoring of a temperature channel with direct connection to TF8540. The adjustable limits are synchronized with the temperature zone.

Syntax:

```
FUNCTION_BLOCK FB_MonitoringTemp EXTENDS FB_Monitoring
```

Properties

Name	Type	Access	Initial value	Description
TempChannel	I_TempChannel	Get, Set	NULL	Assigned temperature channel

Interfaces

Type	Description
I_MonitoringTemp	Standard interface on FB_MonitoringTemp

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.11.2.2 FB_MonitoringIPC

Implements the monitoring of the CPU temperature of a Beckhoff IPC.

Syntax:

```
FUNCTION_BLOCK FB_MonitoringIPC EXTENDS FB_Monitoring
```

Properties

Name	Type	Access	Initial value	Description
RefreshRate	LREAL	Get, Set	5.0	Rate [s] of the asynchronous request of the IPC value

Interfaces

Type	Description
I_MonitoringIPC	Standard interface on FB_MonitoringIPC

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.11.3 FB_Setpoints

Implements scaling of analog setpoints that can be adjusted via the HMI.

Syntax:

```
FUNCTION_BLOCK FB_Setpoints EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
Output	I_OutputBase	Get, Set	NULL	Interface of the analog output to be assigned

Interfaces

Type	Description
I_Setpoints	Standard interface on FB_Setpoints
I_AttachableMdInterface	Interface for containerless machine data components
I_OneTaskInterface	Runtime interface for a PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.12 Timer - process timing

5.12.1 FB_TimerHmi

HMI interface for IEC 61131-3 timer to set process timings.

Syntax:

```
FUNCTION_BLOCK FB_TimerHmi EXTENDS FB_BaseHmi
```

Properties

Name	Type	Access	Initial value	Description
ActualValue	LREAL	Get, Set	0.0	Current time value in seconds
LatchedValue	LREAL	Get, Set	0.0	Stored time value of the last execution in seconds
Out	BOOL	Get, Set	FALSE	Timer output (depending on TON, TOF, TP)
SetValue	LREAL	Get, Set	0.0	Preset time setpoint in seconds

Interfaces

Type	Description
I_TimerHmi	Standard interface on FB_TimerHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.12.2 FB_TimerTon

Implements IEC 61131-3 timer function with integrated interface for PLC and HMI.

Syntax:

```
FUNCTION_BLOCK FB_TimerTon EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Elapsed	LREAL	Get	0.0	Current time value in seconds
Latched	LREAL	Get	0.0	Stored time value of the last execution in seconds
Preset	LREAL	Get, Set	0.0	Preset time value in seconds
Et	TIME	Get	T#0ms	Current time value in milliseconds
In	BOOL	Get, Set	FALSE	Activation input
L	TIME	Get	T#0ms	Stored time value of the last execution in milliseconds
Pt	TIME	Get, Set	T#0ms	Preset time value in milliseconds
Q	BOOL	Get	FALSE	Timer output

Interfaces

Type	Description
I_Timer	General interface for IEC 61131-3 timer

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.12.3 FB_TimerTof

Implements IEC 61131-3 timer function with integrated interface for PLC and HMI.

Syntax:

```
FUNCTION_BLOCK FB_TimerTof EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Elapsed	LREAL	Get	0.0	Current time value in seconds
Latched	LREAL	Get	0.0	Stored time value of the last execution in seconds
Preset	LREAL	Get, Set	0.0	Preset time value in seconds
Et	TIME	Get	T#0ms	Current time value in milliseconds
In	BOOL	Get, Set	FALSE	Activation input
L	TIME	Get	T#0ms	Stored time value of the last execution in milliseconds
Pt	TIME	Get, Set	T#0ms	Preset time value in milliseconds
Q	BOOL	Get	FALSE	Timer output

Interfaces

Type	Description
I_Timer	General interface for IEC 61131-3 timer

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.12.4 FB_TimerTp

Implements IEC 61131-3 timer function with integrated interface for PLC and HMI.

Syntax:

```
FUNCTION_BLOCK FB_TimerTp EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Elapsed	LREAL	Get	0.0	Current time value in seconds
Latched	LREAL	Get	0.0	Stored time value of the last execution in seconds
Preset	LREAL	Get, Set	0.0	Preset time value in seconds
Et	TIME	Get	T#0ms	Current time value in milliseconds
In	BOOL	Get, Set	FALSE	Activation input
L	TIME	Get	T#0ms	Stored time value of the last execution in milliseconds
Pt	TIME	Get, Set	T#0ms	Preset time value in milliseconds
Q	BOOL	Get	FALSE	Timer output

Interfaces

Type	Description
I_Timer	General interface for IEC 61131-3 timer

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.12.5 FB_TimerWeekdayHmi

HMI interface for a weekday timer for use with the TF8550 Control TimeScheduler.

Syntax:

```
FUNCTION_BLOCK FB_TimerWeekdayHmi EXTENDS FB_TimerHmi
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInterface	I_MdComponent	Get	-	Interface to the machine data component
Duration	LREAL	Get	0.0	Time difference between start and end time
EndTime	LREAL	Get, Set	0.0	End time in seconds
StartTime	LREAL	Get, Set	0.0	Start time in seconds
Weekday	E_Weekday [▶ 88]	Get, Set	eNone	Weekday of the start time

Interfaces

Type	Description
I_TimerWeekday	Standard interface on FB_TimerWeekdayMaster
I_TimerWeekdayHmiQuery	Interface for internal queuing of multiple weekday timers
I_AttachableMdInterface	Interface for containerless machine data components

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.12.5.1 E_Weekday

Enumeration of the weekdays on which a schedule is to start.

Syntax:

```
TYPE E_Weekday :
(
    // invalid - inactive
    eNone              := 2#0000_0000,

    // One day
    eMonday             := 2#0000_0001,
    eTuesday            := 2#0000_0010,
    eWednesday          := 2#0000_0100,
    eThursday           := 2#0000_1000,
    eFriday             := 2#0001_0000,
    eSaturday           := 2#0010_0000,
    eSunday             := 2#0100_0000
```

Values

Name	Description
eNone	Inactive or no weekday
eMonday	Monday
eTuesday	Tuesday
eWednesday	Wednesday
eThursday	Thursday
eFriday	Friday
eSaturday	Saturday
eSunday	Sunday

5.12.6 FB_TimerWeekdayMaster

Implements a weekday timer, compatible with IEC 61131-3 timers.

Syntax:

```
FUNCTION_BLOCK FB_TimerWeekdayMaster EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
Elapsed	LREAL	Get	0.0	Current time value in seconds
Latched	LREAL	Get	0.0	Stored time value of the last execution in seconds
Preset	LREAL	Get, Set	0.0	Preset time value in seconds
Et	TIME	Get	T#0ms	Current time value in milliseconds
In	BOOL	Get, Set	FALSE	Activation input
L	TIME	Get	T#0ms	Stored time value of the last execution in milliseconds
Pt	TIME	Get, Set	T#0ms	Preset time value in milliseconds
Q	BOOL	Get	FALSE	Timer output

Interfaces

Type	Description
I_TimerWeekday	Standard interface on FB_TimerWeekdayMaster

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13 BlowMolding - Blow molding specific classes

5.13.1 FB_Blowing

Implements a typical blowing sequence in two blowing phases with adjustable pressure.

Syntax:

```
FUNCTION_BLOCK FB_Blowing EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AttachableMdInteface	I_MdComponent	Get	-	Interface to the machine data component
Done	BOOL	Get	FALSE	The blowing process was carried out successfully.
EnableOutput	BOOL	Get, Set	FALSE	Releases the output of the blowing pressure.
Output	I_OutputBase	Get, Set	NULL	Represents the interface to the analog output.

Methods

Name	Description
Start()	Starts the blowing process.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
BlowSeq() [▶ 90]	Procedure for blowing pressure output

Interfaces

Type	Description
I_Blowing	Standard interface on FB_Blowing
I_AttachableMdInterface	Interface for containerless machine data components

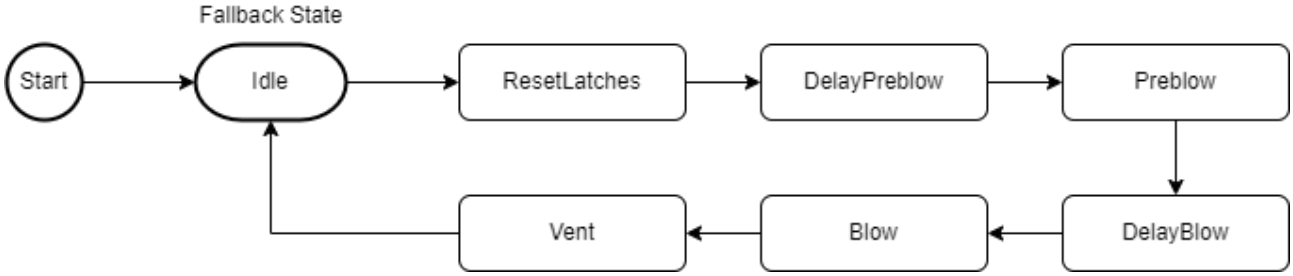
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.1.1 BlowSeq()

Instance type	Instance Name
Master	`fbBlowSeq`
Slaves	`aBaseSeqMembers[E_BlowingSequence.eLength]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.2 FB_IntervalBlowing

Extends the FB_Blowing class with an alternating blowing interval during the main blowing phase.

Syntax:

```
FUNCTION_BLOCK FB_IntervalBlowing EXTENDS FB_Blowing
```

Properties

Name	Type	Access	Initial value	Description
EnableInterval	BOOL	Get, Set	-	Turns on the interval extension.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
BlowSeq() [▶ 91]	Procedure for blowing pressure output (advanced)

Interfaces

Type	Description
I_IntervalBlowing	Standard interface on FB_IntervalBlowing

Requirements

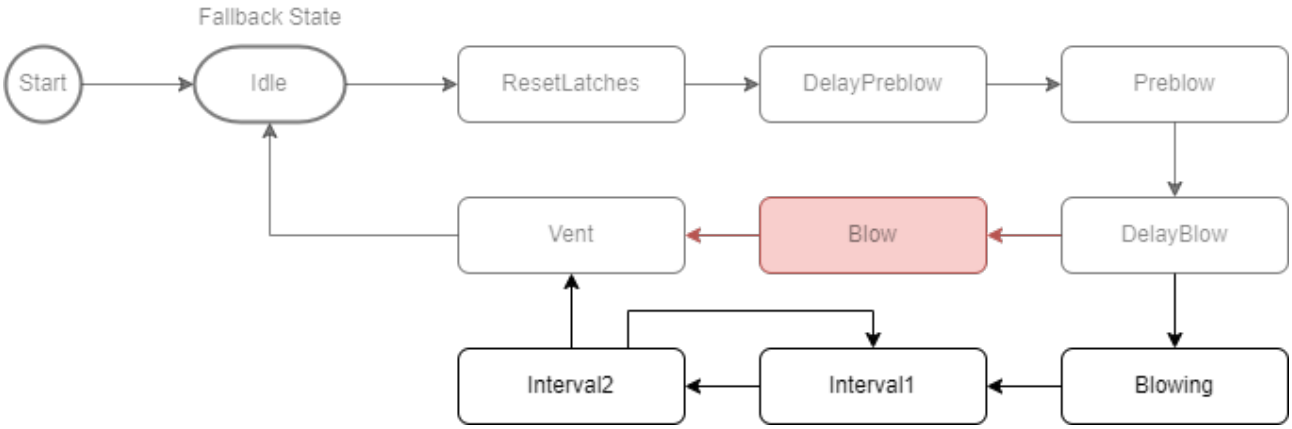
Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.2.1 BlowSeq()

Instance type	Instance Name
Master	`fbBlowSeq`

Slaves	`fbStateBlowing`
	`fbStateInterval1`
	`fbStateInverval2`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.3 FB_Blowpin

Implements blowpin-specific functions.

Syntax:

```
FUNCTION_BLOCK FB_Blowpin EXTENDS FB_PtpMotion
```

Properties

Name	Type	Access	Initial value	Description
DisableHoldingTorque	BOOL	Get, Set	FALSE	Locks the holding torque for NC blowpin axes.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
PtpSeq() [▶ 93]	Procedure for loading a PTP movement (extended)

Interfaces

Type	Description
I_Blowpin	Standard interface on FB_Blowpin

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

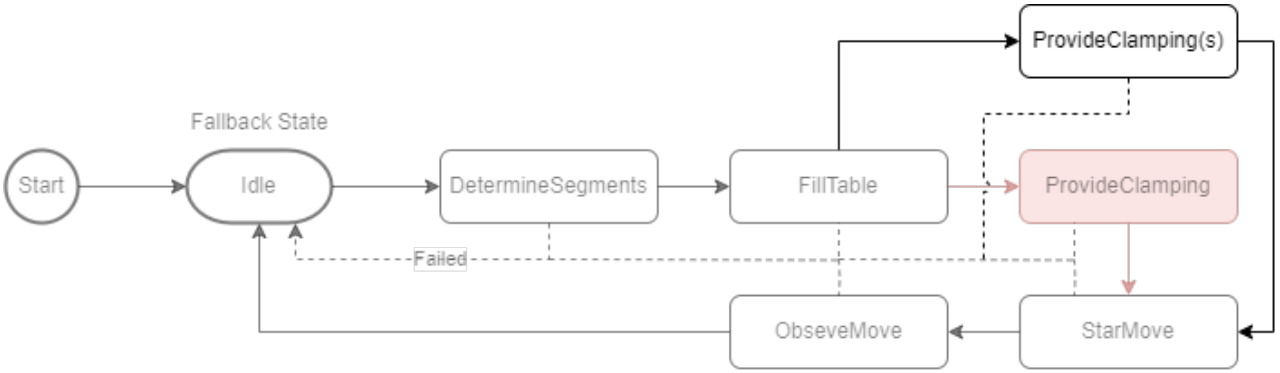
5.13.3.1 PtpSeq()

Blowpin PTP variant only active for NC axes

`fbSeqProvideClamping` is only inserted when using an NC axis. For hydraulic axes, the normal clamping of the `FB_PtpMotion` class is used.

Instance type	Instance Name
Master	`fbPtpSeq`
Slaves	`fbSeqProvideClamping`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.13.4 FB_BlowMoldingExtruder

Implements specific functions for extruders in the blow molding process.

Syntax:

```
FUNCTION_BLOCK FB_BlowMoldingExtruder EXTENDS FB_Extruder
```

Properties

Name	Type	Access	Initial value	Description
ParisonLengthControl [▶ 94]	I_ParisonLengthControl	Get	-	Interface to the integrated parison length control

Interfaces

Type	Description
I_BlowMoldingExtruder	Standard interface on FB_BlowMoldingExtruder

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.4.1 FB_ParisonLengthControl

Implements parison length control in the typical use case of the blow molding process.

Syntax:

```
FUNCTION_BLOCK FB_ParisonLengthControl EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Photocell	BOOL	Set	FALSE	Input for the signal of the photocell
TurnrateDown	BOOL	Get	FALSE	Control output for lowering the turnrate
TurnrateUp	BOOL	Get	FALSE	Control output for raising the turnrate
WtcStart	BOOL	Get, Set	FALSE	Input signal for the start of a new cycle

Methods

Name	Description
Activate()	Activates the parison length control.

Interfaces

Type	Description
I_ParisonLengthControl	Standard interface on FB_ParisonLengthControl

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.5 FB_Wtc

Implements wall thickness control for the extruded hose in blow molding applications.

Syntax:

```
FUNCTION_BLOCK FB_Wtc EXTENDS FB_Axis
```

Methods

Name	Description
CamIn()	Loads the current curve and starts camming.
DieWaitPosition()	Commands the WTC to the waiting position.
Power()	Activates the axis including the connection to the cam plate.
Testposition()	Commands the WTC to the test position.

Procedure controlling methods (FB_AdaptableSequence [▶ 97])

Name	Description
WtcStates() [▶ 95]	Procedure for executing commands and coupling and decoupling from the cam plate

Interfaces

Type	Description
I_Wtc	Standard interface on FB_Wtc

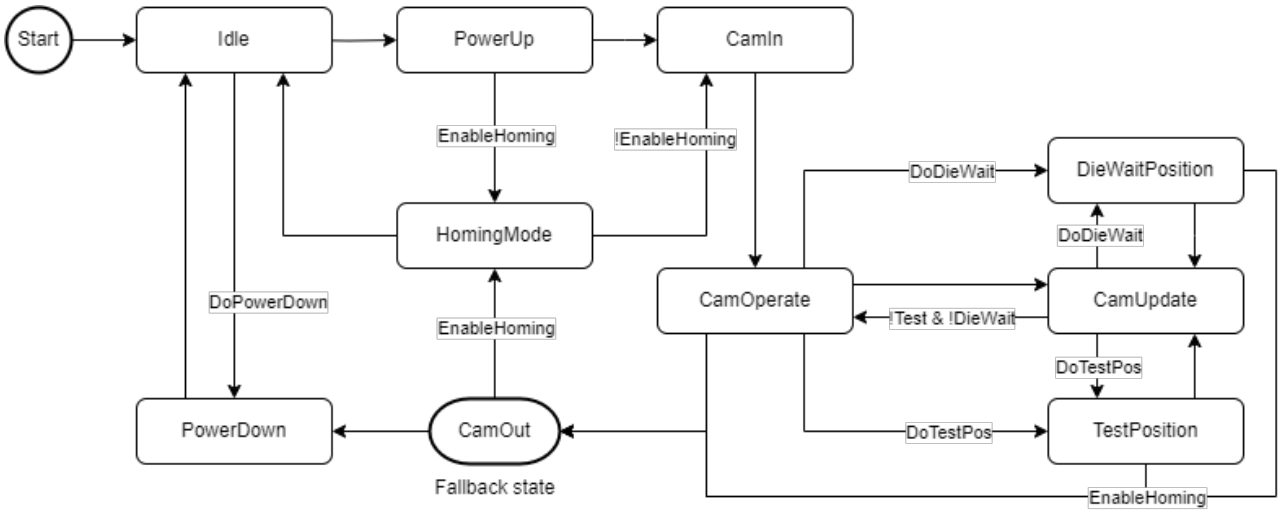
Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.5.1 WtcStates()

Instance type	Instance Name
Master	`fbWtcStates`
Slaves	`aBaseSeqMembers[E_WtcState.eLength]`

State diagram:



Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.13.6 FB_WtcTimeMaster

Implements a timer as master for WTC camming.

Syntax:

```
FUNCTION_BLOCK FB_WtcTimeMaster EXTENDS FB_Axis
```

Properties

Name	Type	Access	Initial value	Description
ActualTime	LREAL	Get	0.0	Elapsed time since start of WTC cycle in seconds
ActualTimeLatched	LREAL	Get	0.0	Stored time of the last WTC cycle in seconds
FirstStart	BOOL	Get	FALSE	WTC is in its first cycle since the last launch.
GuidingValue	LREAL	Get	0.0	Time value as resulting camming master value
ProfileStarted	BOOL	Get	FALSE	TRUE if the timer has been started.

Methods

Name	Description
ProfileStartAck()	Resets the "ProfileStarted" feedback signal.
Start()	Starts the timer.

Interfaces

Type	Description
I_WtcTimeMaster	Standard interface on FB_WtcTimeMaster
I_WtcMaster	General interface of a WTC master axis

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.13.7 FB_WtcAccuMaster

Implements an accumulator as the master for WTC camming.

Syntax:

```
FUNCTION_BLOCK FB_WtcAccuMaster EXTENDS FB_Axis
```

Properties

Name	Type	Access	Initial value	Description
FillingDone	BOOL	Get	FALSE	The accumulator has reached the filling volume.
GuidingValue	LREAL	Get	0.0	Position as resulting camming master value
PushoutDone	BOOL	Get	FALSE	The discharge is completed (the remaining filling volume is below the buffer volume).

Methods

Name	Description
Filling()	Starts the filling process of the accumulator.
Pushout()	Starts the pushing-out the filling volume.

Interfaces

Type	Description
I_WtcAccuMaster	Standard interface on FB_WtcAccuMaster
I_WtcMaster	General interface of a WTC master axis

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14 Utilities

5.14.1 FB_AdaptableSequence

Allows variable extension of implemented procedures of a class. The class itself is used to manage all sequences and to indicate the state of a single sequence with reference to the management (master).

Syntax:

```
FUNCTION_BLOCK FB_AdaptableSequence
```

Properties

Name	Type	Access	Initial value	Description
ActiveSeq	I_AdaptableSeqItf	Get	THIS^	Interface to the current step of the procedure
Done	BOOL	Get, Set	FALSE	Indicates successful processing of a procedure step.
Failed	BOOL	Get, Set	FALSE	Indicates the failed execution of a procedure step.
Index	BOOL	Get	INT	Index of the class in relation to the overall procedure
IsActive	BOOL	Get	FALSE	The class is active as the current sequence.
IsMaster	BOOL	Get	TRUE	The class is the management object of a procedure.
Length	INT	Get	0	Length of the list of attached procedure steps
Next	I_AdaptableSeqItf	Get	NULL	Interface to the next step of the procedure
Tag	I_FlexValue	Get	-	Arbitrary value for saving user-defined information

Methods

Name	Description
Check() [▶ 99]	Checks whether the current step of the procedure has been processed.
Clear() [▶ 99]	Deletes all sequences from the master.
Exists() [▶ 99]	Checks whether a sequence is included in the master.
Idx() [▶ 100]	Returns the xth step from the procedure.
Insert() [▶ 101]	Adds another step to the procedure.
Jump() [▶ 101]	Requests jumping to a sequence that does not follow.
Reset()	Slave: Resets the state of the sequence. Master: Resets the progress of the procedure.
SetMaster()	[INTERNAL] Assigns a master to the sequence. Used by the Insert() method and does not need to be called separately.

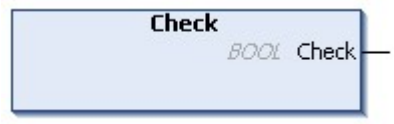
Interfaces

Type	Description
I_AdaptableSeqItf	Interface for implementation as slave
I_AdaptableSeqState	Interface for reading the sequence state
I_AdaptableSeqQuery	Advanced interface for editing by the master
I_AdaptableSeqMaster	Interface for implementation as master

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.1.1 Check()



Checks whether the current step of the procedure has been processed. If this is the case, the method also causes switching to the next sequence.

Use is already integrated

This method is already implemented for existing instances in the TwinCAT 3 Plastic Base Application. It is not recommended to use this method for checking within a sequence. A call of the function is only necessary when a procedure is restarted.

Syntax:

```
METHOD Check : BOOL
```

Outputs

Name	Type	Description
Check	BOOL	TRUE if the current step has been completed and the next step is initiated.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.2 Clear()

Deletes all sequences from the master.

Syntax:

```
METHOD Clear : HRESULT
```

Outputs

Name	Type	Description
Clear	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.3 Exists()

Checks whether a process step is included in the master.

Syntax:

```
METHOD Exists : HRESULT
VAR_INPUT
    iSeq:        I_AdaptableSeqItf;
END_VAR
```

Inputs

Name	Type	Description
iSeq	I_AdaptableSeqItf	Process step to be checked

Outputs

Name	Type	Description
Exists	HRESULT	Return value with feedback on the success of the check

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.4 Idx()

Returns the xth step from the procedure.

Syntax:

```
METHOD Idx : I_AdaptableSeqItf
VAR_INPUT
    nIdx:        INT;
END_VAR
```

Inputs

Name	Type	Description
nIdx	INT	The requested index

Outputs

Name	Type	Description
Idx	I_AdaptableSeqItf	Found step of the procedure

Invalid indexes

To avoid exceptions caused by an invalid value on input nIdx, the function returns the master in case of an error. Therefore, the method should be treated comparable to an array index by considering the total length of the sequence list.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.5 Insert()

Adds another step to the procedure.

Syntax:

```
METHOD Insert : HRESULT
VAR_INPUT
    iCurrent:        I_AdaptableSeqItf;
    iNew:            I_AdaptableSeqItf;
    bOverwrite:      BOOL;
END_VAR
```

Inputs

Name	Type	Description
iCurrent	I_AdaptableSeqItf	The current participant to be moved behind the new participant.
iNew	I_AdaptableSeqItf	New participant to be added
bOverwrite	BOOL	TRUE if the current participant is not to be moved but replaced.

Outputs

Name	Type	Description
Insert	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.6 Jump()

Requests jumping to a sequence that does not follow.

Request does not lead directly to execution

When the method is executed, only the request is stored in the master. To execute the jump, the Done or Failed property of the active process step must be set and the Check() method must be called.

See the Check() [▶ 99] method for more important notes.

Syntax:

```
METHOD Jump : HRESULT
VAR_INPUT
    iTarget:        I_AdaptableSeqItf;
END_VAR
```

Inputs

Name	Type	Description
iTarget	I_AdaptableSeqItf	Sequence to be jumped to

Outputs

Name	Type	Description
Jump	HRESULT	Return value with feedback on the success of the request

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.1.7 I_AdaptableSeqExt

Allows the implementation of procedure steps outside the procedure-implementing class

Syntax:

```
INTERFACE I_AdaptableSeqExt EXTENDS I_BaseEmpty
```

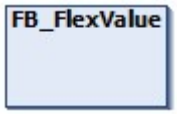
Event-driven methods (callback methods)

Name	Description
ExtAdaptSeq()	Called from the implemented procedure to process the procedure steps outside

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.14.2 FB_FlexValue



Class (FB) comparable to the structured data type ST_FlexValue from TF8560. Represents a variable without a specified data type.

Syntax:

```
FUNCTION_BLOCK FB_FlexValue
```

Properties

Name	Type	Access	Initial value	Description
Value	U_FlexValue	Get	0	Value as Union (all types)
ValueType	E_FlexValue	Get	eBOOL	Last assigned value data type (except if assigned via Union)
_BOOL	BOOL	Get, Set	FALSE	Value as BOOL
_INT	INT	Get, Set	0	Value as INT
_LREAL	LREAL	Get, Set	0.0	Value as LREAL

Interfaces

Type	Description
I_FlexValue	Standard interface on FB_FlexValue

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.3 FB_Parameter

Allows concatenation of multiple 64-bit floating point (LREAL) values with assignable name.

Syntax:

```
FUNCTION_BLOCK FB_Parameter EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
Name	STRING	Get, Set	''	Name of the parameter
Next	I_Parameter	Get, Set	NULL	Next item in the list
Root	I_Parameter	Get, Set	THIS^	First element of the list
Value	LREAL	Get, Set	0.0	Value of the parameter

Methods

Name	Description
Idx()	Returns the element of the xth position of the list.

Exceptions avoidance

The Idx() function returns the first element (root) of the list on invalid requested index.

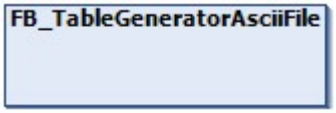
Interfaces

Type	Description
I_Parameter	Standard interface on FB_Parameter
I_ParameterQuery	Extension of the I_Parameter interface with set access to next and root properties

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.4 FB_TableGeneratorAsciiFile



Implements a file read mechanism for a user-defined transformation table.

Formatting the file:

Drive Load

```
0.0         100.0
1.0         110.0
2.0         121.0
3.0         131.0
4.0         142.0
…
```

Drive points must be equidistant

The points of the drive side must be equidistant! Otherwise, all points between start and end will shift to equidistant distances, resulting in unwanted inaccuracy.

Syntax:

```
FUNCTION_BLOCK FB_TableGeneratorAsciiFile EXTENDS Tc3_PlasticFunctions.FB_TrafoTableGenerator
```

Properties

Name	Type	Access	Initial value	Description
FilePath	STRING	Get, Set	‘‘	File path on the target system to the stored description file
LoadHighEnd	LREAL	Get	0	Read highest point of the load side
LoadLowEnd	LREAL	Get	0	Read lowest point of the load side

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.5 FB_TrendHmi

Support class for the TwinCAT HMI SQLiteTrend with pre-implemented views. The TF8550 function TcHmi.Functions.Plastic.UpdateTrend() is required for use. The selected view is additionally determined by the property `SelectedView`. The following objects with corresponding interface can be attached to the trend:

Type	Display value
I_ExtruderHmi	Actual turnrate of the extruder [RPM]
I_MonitoringHmi	Monitoring value [*any*]
I_TempChannel	Actual temperature of a temperature channel [°C]
I_Temperature	Actual temperature of each grouped temperature channel [°C]
I_TempCtrlHmi [1]	Temperature of each zone [°C]

[1] Obsolete

Internal functions:

- The first `x` views are pre-reserved for displaying the temperature groups
  - Default `x := 5`
  - Automatically adapting to the set number of groups [▶ 116]
  - When the FB_Temperature [▶ 49] group configuration is changed, the views are automatically adapted as well
- All values attached via `Append()` are available for configuration by the `ConfigXyz()` methods

Syntax:

```
FUNCTION_BLOCK FB_TrendHmi EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
SelectedView	INT	Get, Set	0	Selected view

Methods

Name	Description
Append() [▶ 106]	Append a value to be recorded in the trend and selected in views.
AppendTempChannels() [▶ 107]	Append the temperature zones of an FB_Temperature instance
CheckSupport()	[PROTECTED] Checks the support of the appended object
ConfigDisplayName() [▶ 107]	Overwrites the instance name of a display value.
ConfigDisplayLocalisation() [▶ 108]	Overwrites the instance name with a localization key.
ConfigDisplayLocalisationNuget()	Overwrites the instance name with a localization key from the TF8550.Localisation package.
ConfigView() [▶ 109]	Configures a value in a selectable view.
Idx()	Returns the object at the xth position.
AppendTempZones() [1]	Append the temperature zones of an FB_TempCtrl instance.

[1] Obsolete

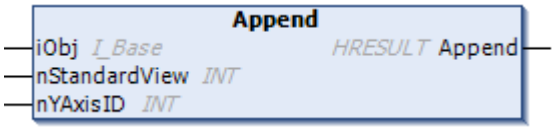
Exceptions avoidance

The Idx() function returns the first element (root) of the list on invalid requested index.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.5.1 Append()



Appends a value to the trend.

Syntax:

```
METHOD Append : HRESULT
VAR_INPUT
    iObj:               I_Base;
    nStandardView:      INT;
    nYAxisID:           INT;
END_VAR
```

Inputs

Name	Type	Description
iObj	I_Base	Object to be included in the trend
nStandardView	INT	Standard view in which the value is displayed
nYAxisID	INT	Y-axis in the HMI on which the value is displayed

The `YAxisID` is predefined for the following units:

ID	Unit	Description
1	°C	Temperature
2	A	Current
3	Bar	Pressure
4	RPM	Turnrate

Outputs

Name	Type	Description
Append	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.5.2 AppendTempChannels()

Appends the temperature configuration to the trend. If the group configuration of all zones changes, this is automatically taken over by `FB_TrendHmi`.

Syntax:

```
METHOD AppendTempChannels : HRESULT
VAR_INPUT
    ipTemp:              I_Temperature;
    nYAxisID:            INT;
END_VAR
```

Inputs

Name	Type	Description
ipTemp	I_Temperature	Instance of the temperature control class (FB)
nYAxisID	INT	Y-axis in the HMI on which the value is displayed

The `YAxisID` is predefined for the following units:

ID	Unit	Description
1	°C	Temperature
2	A	Current
3	Bar	Pressure
4	RPM	Turnrate

Outputs

Name	Type	Description
AppendTempChannels	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.5.3 ConfigDisplayName()

Adjusts the display name for a trend value.

Syntax:

```
METHOD ConfigDisplayName : HRESULT
VAR_INPUT
    iObj:            I_Base;
    sName:           STRING(255);
    bAppend:         BOOL;
END_VAR
```

Inputs

Name	Type	Description
iObj	I_Base	Object whose name is to be adjusted
sName	STRING(255)	String to be displayed
bAppend	BOOL	Passed string is to be appended to the existing display name.

Outputs

Name	Type	Description
ConfigDisplayName	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.5.4 ConfigDisplayLocalisation()

Adjusts the display name for a trend value.

Syntax:

```
METHOD ConfigDisplayName : HRESULT
VAR_INPUT
    iObj:           I_Base;
    sLocalisation:  STRING(255);
    bAppend:        BOOL;
END_VAR
```

Inputs

Name	Type	Description
iObj	I_Base	Object whose name is to be adjusted
sLocalization	STRING(255)	Localization key to be displayed
bAppend	BOOL	Passed localization key is to be appended to the existing display name.

Outputs

Name	Type	Description
ConfigDisplayLocalisation	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.5.5 ConfigView()

Changes the composition of a selectable view.

Syntax:

```
METHOD ConfigView : HRESULT
VAR_INPUT
    iObj:                I_Base;
    nViewIdx:            INT;
    bShow:               BOOL;
END_VAR
```

Inputs

Name	Type	Description
iObj	I_Base	Object to be adjusted in a view
nViewIdx	INT	View (ID) to be customized
bShow	BOOL	Object is displayed (TRUE).

Outputs

Name	Type	Description
ConfigView	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.6 FB_Trigger

Combines triggers of type R_TRIG and F_TRIG with object-oriented interface.

Syntax:

```
FUNCTION_BLOCK FB_Trigger
```

Properties

Name	Type	Access	Initial value	Description
CLK	BOOL	Get, Set	FALSE	Sampled input signal
FQ	BOOL	Get	FALSE	Input signal has a falling edge.
Q	BOOL	Get	FALSE	Input signal has a rising or falling edge.
RQ	BOOL	Get	FALSE	Input signal has a rising edge.

Methods

Name	Description
Cyclic()	Cycle method by which the signal is sampled

Interfaces

Type	Description
I_Trigger	Standard interface on FB_Trigger

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.7 FB_LatchTrigger

Extends the FB_Trigger class by a memory function of the edges.

Syntax:

```
FUNCTION_BLOCK FB_LatchTrigger EXTENDS FB_Trigger
```

Properties

Name	Type	Access	Initial value	Description
LF	BOOL	Get	FALSE	A falling edge was present at the input signal.
LR	BOOL	Get	FALSE	A rising edge was present at the input signal.

Methods

Name	Description
Reset()	Resets LF and LR.

Interfaces

Type	Description
I_LatchTrigger	Standard interface on FB_LatchTrigger

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.8 FB_LibVersion

Defines the structure of the version number of a library

Syntax:

```
FUNCTION_BLOCK FB_LibVersion
```

Properties

Name	Type	Access	Initial value	Description
Build	UDINT	Get	0	Third digit of the version number
Major	UDINT	Get	0	First digit of the version number
Minor	UDINT	Get	0	Second digit of the version number
Released	BOOL	Get	FALSE	Version is marked as 'Released'
Revision	UDINT	Get	0	Fourth digit of the version number
Version	STRING	Get	'v0.0.0.0'	Version number as ascii string
Version3	STRING	Get	'v0.0.0'	.Version without the first digit of the version number

Methods

Name	Description
IsEqualTo()	Compares if another version matches
IsNewerThan()	Checks if this version is newer than the passed comparison version
IsOlderThan()	Checks if this version is older than the passed comparison version
SetVersion()	Sets the version number
SetVersionStruct()	Sets the version number based on a version of type ST_LibVersion

Interfaces

Type	Description
I_LibVersion	Standard interface on FB_LibVersion

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.9 FB_LibVersionBeta

Defines the structure of the version number of a library, plus a beta tag. The property `FB_LibVersionBeta.Version` thus presents itself as 'v5.3.32.9-beta554', for example

Syntax:

```
FUNCTION_BLOCK FB_LibVersion
```

Properties

Name	Type	Access	Initial value	Description
BetaIteration	UDINT	Get, Set	0	Determines the beta iteration of the version

Interfaces

Type	Description
I_LibVersionBeta	Standard interface on FB_LibVersionBeta

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.6.0)

5.14.10 F_SecondsToTime()

Converts a time in seconds of type LREAL to milliseconds of type TIME.

Syntax:

```
FUNCTION F_SecondsToTime : TIME
VAR_INPUT
    fTime:        LREAL;
END_VAR
```

Inputs

Name	Type	Description
fTime	LREAL	Time value in seconds as floating point number

Outputs

Name	Type	Description
F_SecondsToTime	TIME	Time value in milliseconds

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.11 F_TimeToSeconds()

Converts a time in milliseconds of type TIME to seconds of type LREAL.

Syntax:

```
FUNCTION F_SecondsToTime : LREAL
VAR_INPUT
    tTime:        TIME;
END_VAR
```

Inputs

Name	Type	Description
tTime	TIME	Time value in milliseconds

Outputs

Name	Type	Description
F_TimeToSeconds	LREAL	Time value in seconds as floating point number

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.12 F_GetCycleTime()

Returns the cycle time of the calling task as LREAL floating point value in seconds.

Syntax:

```
FUNCTION F_GetCycleTime : LREAL
```

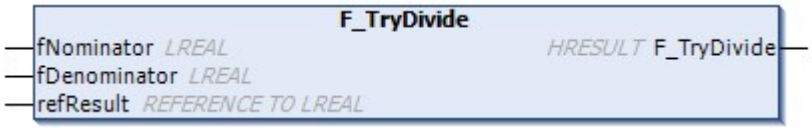
Outputs

Name	Type	Description
F_GetCycleTime	LREAL	Cycle time in seconds as floating point value

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.14.13 F_TryDivide()



Divides two values without throwing an exception.

Return value mathematically invalid

The function defines the mathematically invalid case x / 0 as 0. This is a mathematically invalid result, but is sufficient for many use cases. Check for your use case whether this definition does not lead to unforeseen misbehavior.

Syntax:

```
FUNCTION F_TryDivide : HRESULT
VAR_INPUT
    fNominator:      LREAL;
    fDenominator:    LREAL;
    refResult:       REFERENCE TO LREAL;
END_VAR
```

Inputs

Name	Type	Description
fNominator	LREAL	Value to be divided
fDenominator	LREAL	Value by which to divide
refResult	REFERENCE TO LREAL	Result of the division

Outputs

Name	Type	Description
F_TryDivide	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.14.14 F_GetLocalSystemtime

Returns the local system time, taking into account the time zone. Is suitable for creating system time-related timestamps.

Syntax:

```
FUNCTION F_GetLocalSystemtime : ULINT
```

Outputs

Name	Type	Description
F_GetLocalSystemtime	ULINT	System time, based on the definition of T_FILETIME

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.14.15 F_GetLocalSystemtimeEx

Returns the system time of a device with AMS-NetID, taking into account the time zone. Is suitable for creating system time-related timestamps.

Syntax:

```
FUNCTION F_GetLocalSystemtimeEx : ULINT
```

Inputs

Name	Type	Description
sNetID	T_AmsNetID	Net-ID of the system to be read

Outputs

Name	Type	Description
F_GetLocalSystemtimeEx	ULINT	System time, based on the definition of T_FILETIME

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.4)

5.14.16 PlasticStatusHmi

Status variable for linking to an HMI StateIndicator. This variable is interpreted bit by bit.

Syntax:

```
TYPE PlasticStatusHmi : BYTE; END_TYPE
```

Values

bit	Description
0	Successful (Green)
1	Warning (Orange)
2	Error (Red)
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	State is invalid

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

5.15 Setting parameters - Tc3_PlasticBaseAppStaticParams

Parameter list for the use of the Tc3_PlasticBaseApplication.

Syntax:

```
{attribute 'qualified_only'}
VAR_GLOBAL CONSTANT
    // Static motion paramters
    {attribute 'hide'}
    {attribute 'TcHmiSymbol.Hide'}
    cnNoOfTrafoPoints       :              INT := 181;
    {attribute 'TcHmiSymbol.Hide'}
    cnNoOfCammmingPoints    :              INT := 400;
    {attribute 'TcHmiSymbol.Hide'}
    cnMaxTrafoPoints          :            INT := 181;
    {attribute 'TcHmiSymbol.Hide'}
    cnMaxPtpPoints          :              INT := 6;
    {attribute 'TcHmiSymbol.Hide'}
    cnMaxMoveCluster        :              INT := 2;
    {attribute 'TcHmiSymbol.Hide'}
    cnMaxPtpCams            :              INT := 5;


    // Static temperature parameters
    {attribute 'TcHmiSymbol.Hide'}
    cnTempGroups            :              INT := 10;
    {attribute 'TcHmiSymbol.Hide'}
    cnTempZonesPerGroup     :              INT := 20;
    {attribute 'TcHmiSymbol.Hide'}
    cnTempTimers            :              INT := 30;


    // Trend
    {attribute 'TcHmiSymbol.Hide'}
    cnTrendSize             :              INT := 50;


    // Runtime handling
    {attribute 'TcHmiSymbol.Hide'}
    cnRuntimeObjects        :              INT := 200;


    // Machine Data
    {attribute 'TcHmiSymbol.Hide'}
    csHardDisk              :           STRING := 'C:';
    {attribute 'TcHmiSymbol.Hide'}
    csDataFolderName        :           STRING := 'Data\Machine';
    {attribute 'TcHmiSymbol.Hide'}
    csMachineName           :           STRING := 'Beckhoff';        // left empty for using object
name only in machine-data-filename
END_VAR
```

Values

Name	Description	Default
cnNoOfCammingPoints	Standard number of points of a cam plate (e.g. for using the TF8550 CurveEditor)	400
cnMaxPtpPoints	Maximum number of PTP segments	6
cnMaxTrafoPoints	Maximum number of transformation points for the buffer for loading a table from a file	181
cnMaxMoveCluster	Maximum number of clusters (Grouped PTP segments)	2 (positive/ negative)
cnMaxPtpCams	Maximum number of PTP cams per cluster	5
cnTempGroups	Number of available temperature groups	10
cnTempZonesPerGroup	Number of available zones per temperature group	20
cnTempTimers	Number of weekday timings for temperature control scheduling	30
cnTrendSize	Maximum number of trend values that can be historized	50
cnRuntimeObjects	Number of control objects that can be attached to the runtime	200
csHardDisk	Machine data: Target drive	C:
csDataFolderName	Machine data: Destination folder on the target drive	Data\Machine
csMachineName	Machine data: Abbreviation for the identification of a machine data file	Beckhoff

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.42	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.5)

6 PLC samples / instructions

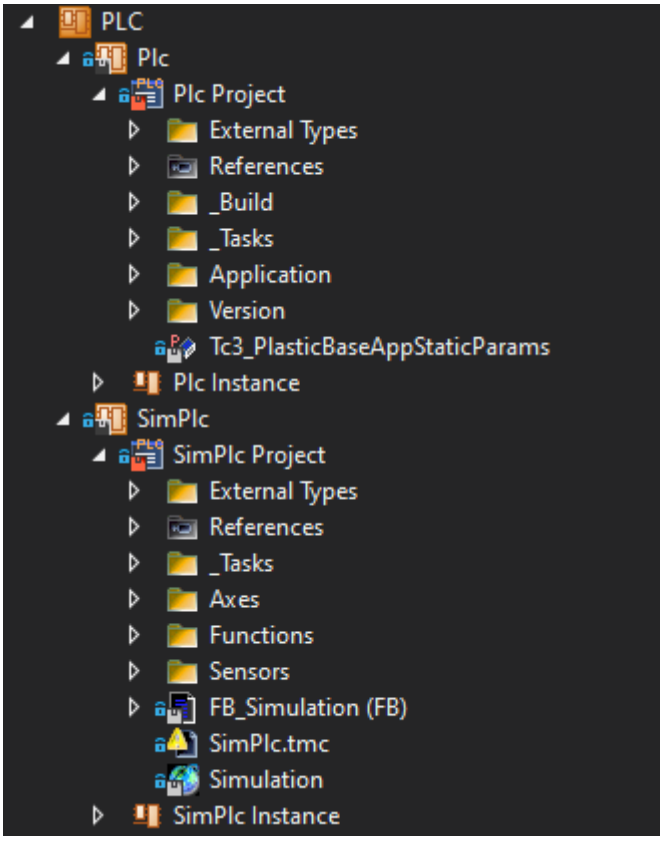
6.1 General

6.1.1 Set up new TwinCAT project

If you want to start a new TwinCAT project with the Plastic Application, you have several options to set up the project. This sample shows the setup steps for a new project based on the supplied ApplicationSample.

1. Place the submitted project file *Tc3_PlasticApplication_V12.5.0.zip* in a folder with the shortest possible project path.
   In this example, the path *C:\Projects* is chosen.
   Avoid long file paths - a too long file path can lead to errors when exporting the ZIP file or opening the project. Therefore, avoid storing the project under long file paths such as the user folders (C:\Users\{UserName}\Documents\TcXaeShell\).
2. Unpack the ZIP file.
3. Open the *Tc3_PlasticApplicationPlc*.tsproj* file in one of the subdirectories using TcXaeShell.
   Blow molding machine:
   Tc3_PlasticApplicationPlc\BlowMolding\Tc3_PlasticBaseApplication\Tc3_PlasticApplicationPlcBMM.tsproj
   Extruder:
   Tc3_PlasticApplicationPlc\Extruder\Tc3_PlasticBaseApplication\Tc3_PlasticApplicationPlcEXTR.tsproj
   The warning message that appears when opening the project, stating that a tmc file was not found, can be ignored. The missing file is created automatically when the project is created.

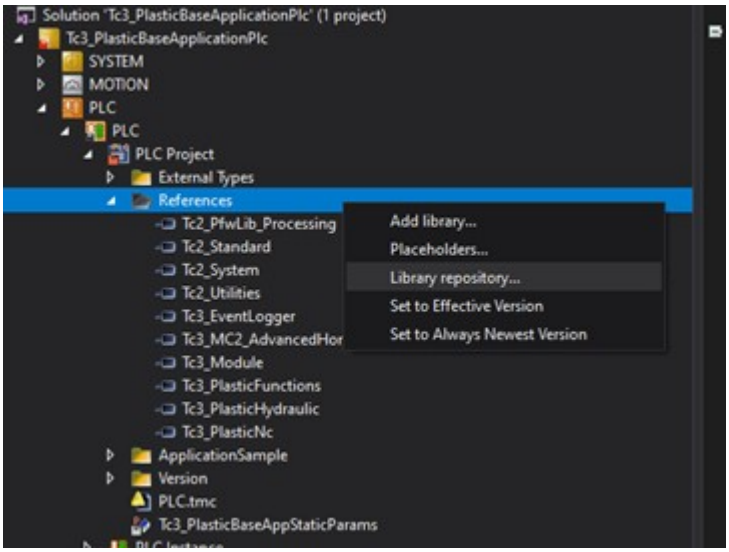
⇨ The project is ready to start.
You see two PLC projects:
The application and a simulation PLC. In its form, the simulation acts as a stand-alone project like the I/O of a machine. Accordingly, the simulation can be mapped to the control PLC like a machine.

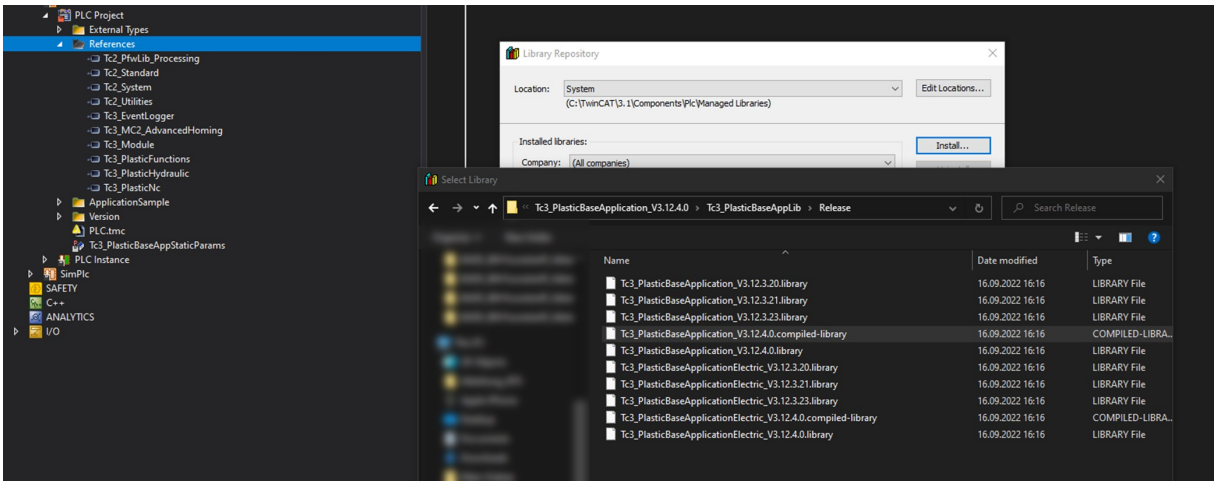


6.1.2 Set up empty project / extend existing project

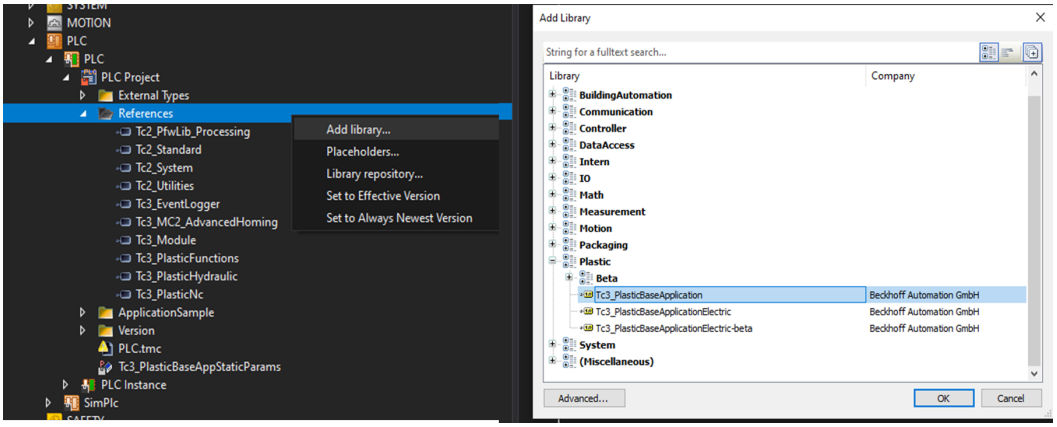
In addition to the possibility of setting up a project on top of the supplied project, the Plastic Base Application can be integrated into an existing or empty project. This is made possible by adding the Plastic Base Application as a library to an existing or new project.

1. Install the library in the TwinCAT Library Manager. The file is located in the subdirectory *\Dependencies\Tc3_PlasticBaseApplication_V3.12.5.0.compiled-library*





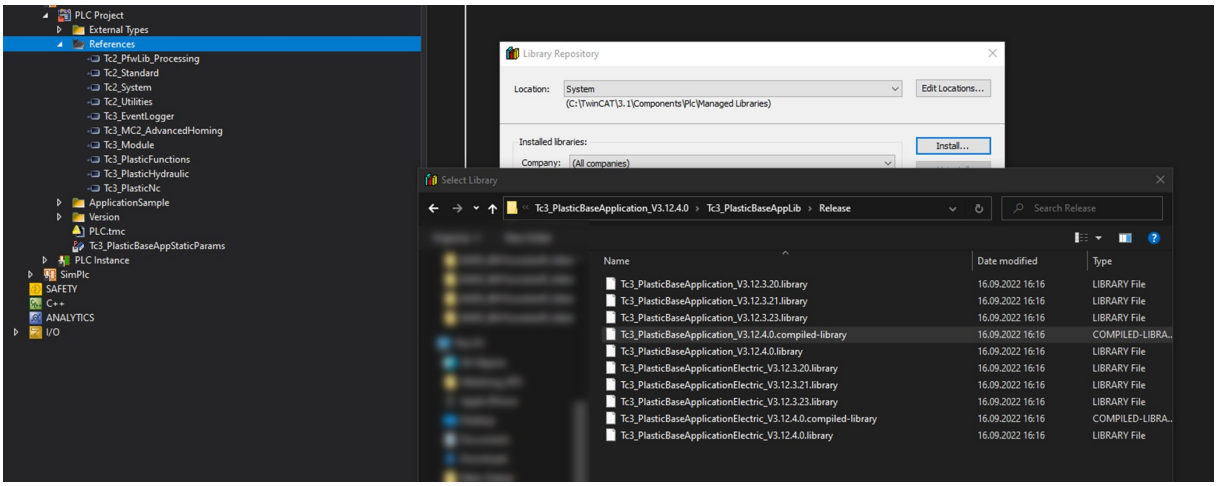
2. Add the library to the project



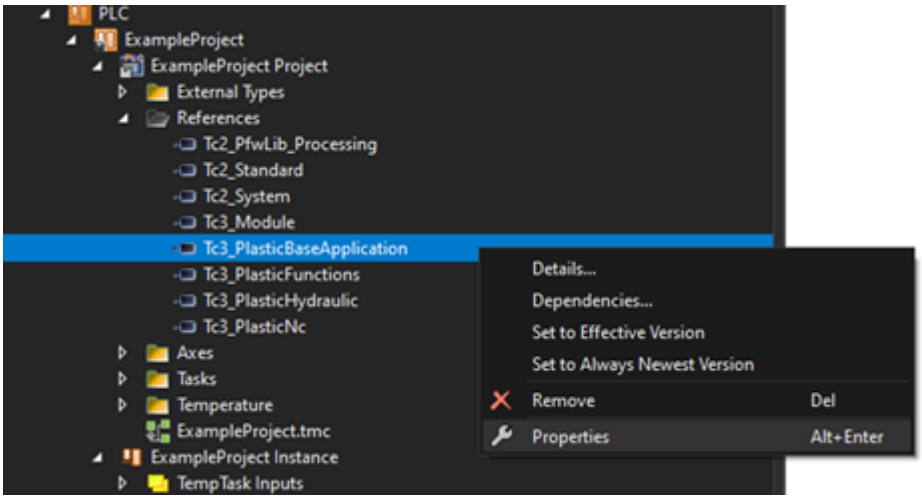
6.1.3 Update Plastic Base Application subsequently in the project

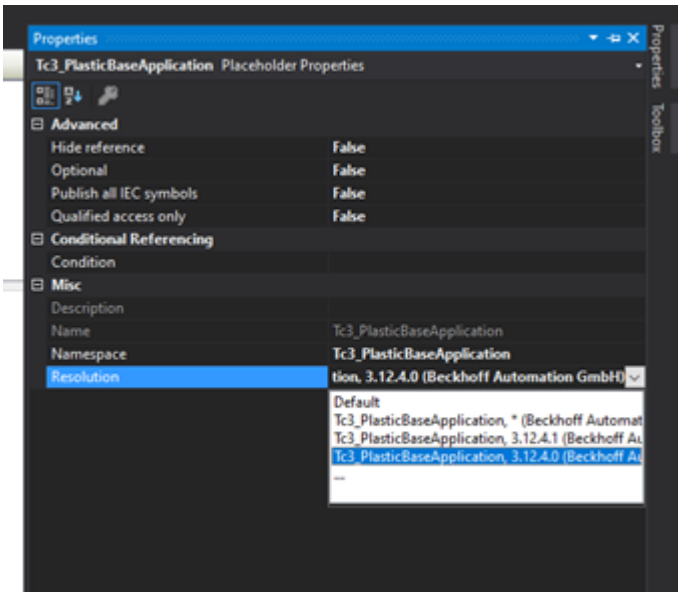
If you want to integrate new features of following Plastic Base Application versions into your existing project, the TwinCAT 3 Plastic Base Application offers the possibility of an update.

1. Install the newer version of the Plastic Base Application version in your Library Manager.



2. Change the fixed version of the library to the newly installed version.





3. When you create the project, check that the new library has been adopted correctly.

6.2 Object orientation

6.2.1 Adding a variable to a class (FB)

If the scope of the internal variables of a class (FB) is to be extended, proceed as follows. In the following example, a variable is added to the default axis type `FB_Axis`.

1. Create a new class (FB) and remove `VAR_INPUT` and `VAR_OUTPUT`.

```
FUNCTION_BLOCK FB_CustomAxis
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
END_VAR
```

2. Add the class to be inherited to the class definition using the `EXTENDS` keyword.

```
FUNCTION_BLOCK FB_CustomAxis EXTENDS FB_Axis
VAR

END_VAR
```

3. Create the new variables in the `VAR` section.

```
FUNCTION_BLOCK FB_CustomAxis EXTENDS FB_Axis
VAR
    bNewVariable1:        BOOL;
    nNewVariable2:        INT;
    fNewVariable3:        LREAL;
END_VAR
```

4. Compile the project to check the implementation for correct syntax.

⇨ You have successfully added new variables to a class.

Example result in the logged in PLC:

fbCustomAxis	FB_CustomAxis	
eBaseState	E_BASESTATE	eInit
hrLastInternalError	HRESULT	16#00000000
sObjectName	STRING	'fbCustomAxis'
fbMessage	FB_AppMessage	
bError	BOOL	FALSE
sNamespace	STRING	'PRG_AxisApplica...
fbMachineData	FB_MdBaseContainer	
MachineData	I_MdBaseContainer	16#FFFFDE87FC...
iAxisBase	I_AxisBase	16#0000000000...
iAxisHmi	I_AxisHmi	16#0000000000...
eErrorAlarmTL	TCEVENTSEVERITY	Verbose
fbAxisHoming	FB_Homing	
fbAxisSpecific	FB_AxisSpecific	
fbAxisData	FB_MdAxis	
bNewVariable1	BOOL	FALSE
nNewVariable2	INT	0
fNewVariable3	LREAL	0

6.2.2 Adding a property or method to a class (FB)

In many cases, the scope of methods and properties of a class should be changed. This includes adding new elements as well as changing or removing existing elements. In the following samples, these three procedures are explained using the standard axle type `FB_Axis` as an example.

The following steps must be completed in advance for all three procedures:

1. Create a new class (FB) and remove `VAR_INPUT` and `VAR_OUTPUT`.

```
FUNCTION_BLOCK FB_CustomAxis
VAR_INPUT
END_VAR
VAR_OUTPUT
```

```
END_VAR
VAR
END_VAR
```

2. Add the class to be inherited to the class definition using the `EXTENDS` keyword.

```
FUNCTION_BLOCK FB_CustomAxis EXTENDS FB_Axis
VAR

END_VAR
```

3. So that you can also address the class and the added elements via an interface, create an interface with the same name.

```
INTERFACE I_CustomAxis
```

4. Let the interface inherit from the interface of the inherited class.

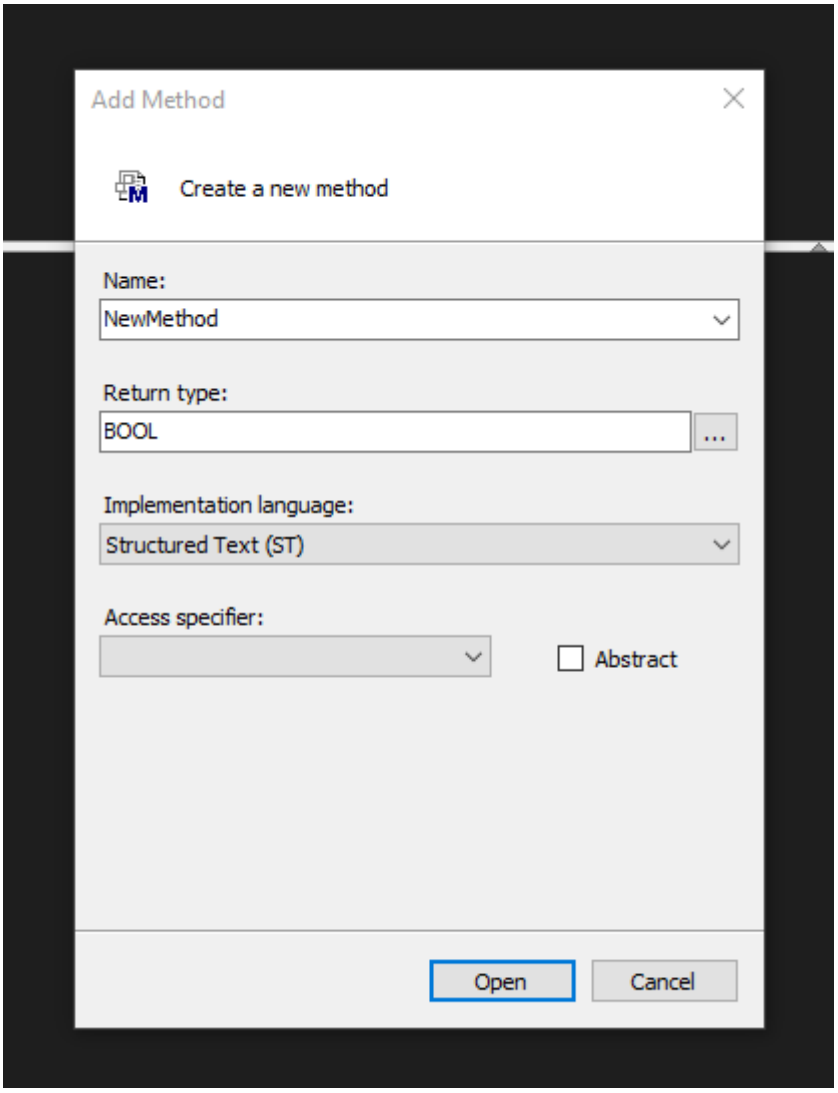
```
INTERFACE I_CustomAxis EXTENDS I_Axis
```

5. Implement the interface in the previously created class (FB).

```
FUNCTION_BLOCK FB_CustomAxis EXTENDS FB_Axis IMPLEMENTS I_CustomAxis
VAR
END_VAR
```

Adding a new method/property

1. Add a new method/property to the class.



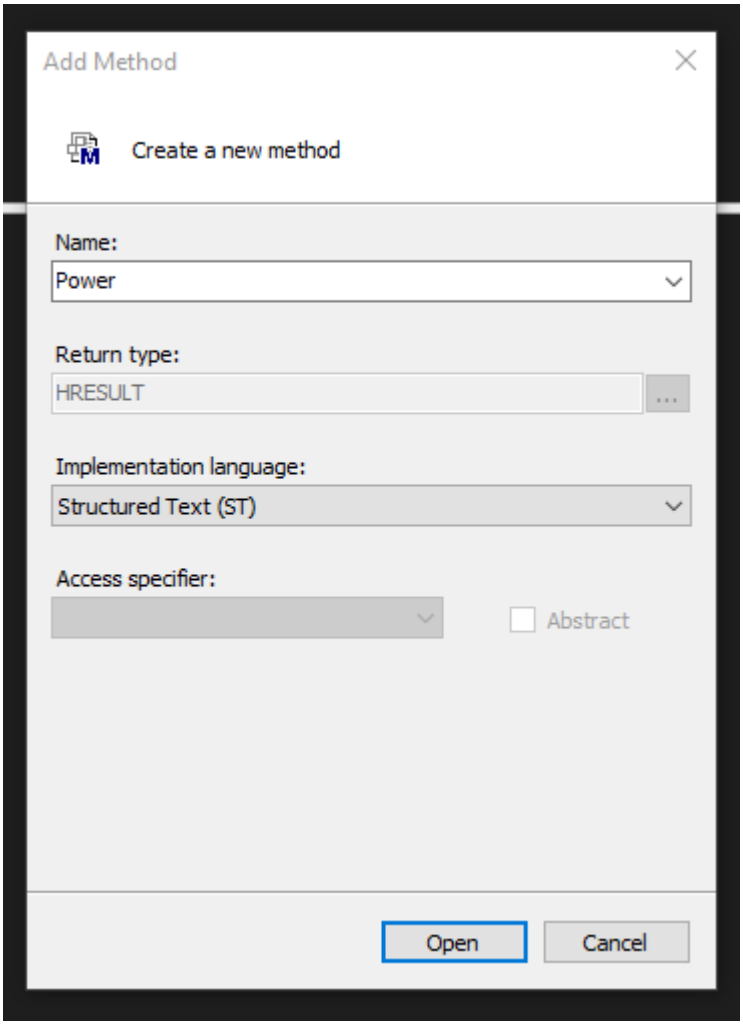

```
METHOD NewMethod : HRESULT
VAR
END_VAR
```

2. Copy the method into the created interface, if the method should be accessible from outside.

⇨ You have successfully added a new method and can start the implementation.

Extend or overwrite a method/property from the inherited class

1. Add a method/property to the class with the name of the method/property to be extended.




```
METHOD Power : HRESULT
VAR_INPUT
    bCommand:        BOOL;
END_VAR
VAR
END_VAR
```

2. If you want to extend the method/property rather than overwrite it, you must call the base implementation at the appropriate place.

```
METHOD Power : HRESULT
VAR_INPUT
    bCommand:        BOOL;
END_VAR
VAR
    nValue:          INT;
END_VAR


IF bCommand THEN
    nValue := 10;
END_IF

SUPER^.Power(bCommand);
```

3. Check the return value and the `INPUT` variables for consistency. You can view the base implementation by selecting the `SUPER^.MethodName()` and then pressing the F12 key.
4. Implement their own lines of code in the method.

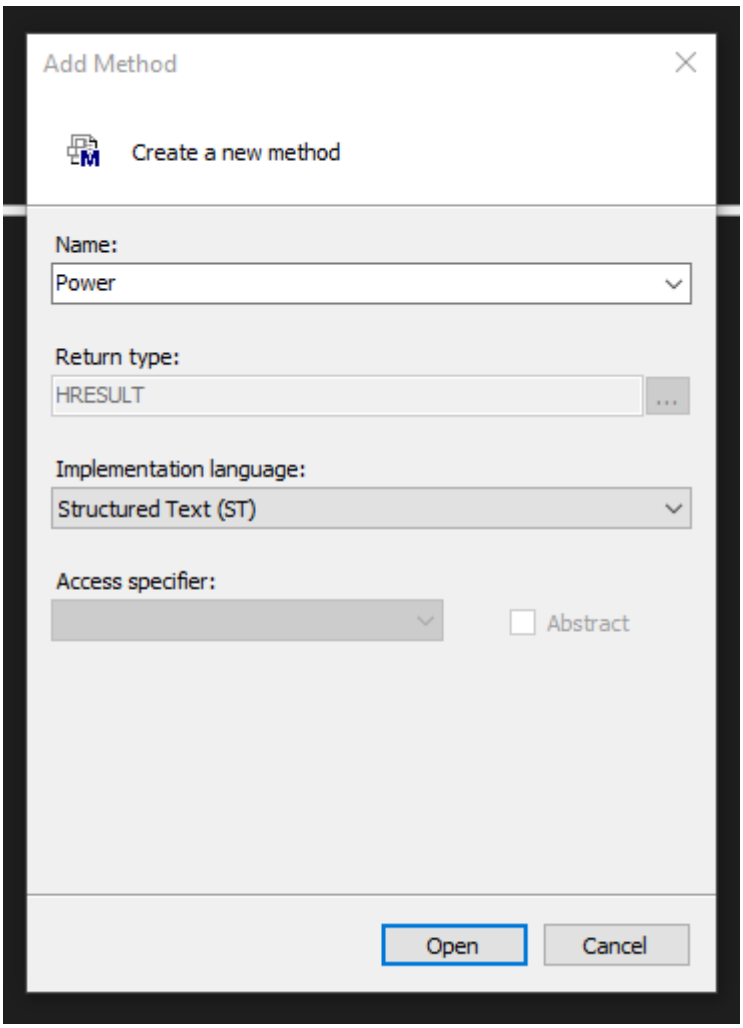
⇨ You have successfully extended a method/property.

Removing a method/property from the inherited class

Removing a method/property is only possible indirectly

Note that you cannot completely remove a method/property! The steps described below only result in the call of the "removed" method/property not causing a reaction.

1. Add a method/property to the class with the name of the method/property to be removed.



2. Leave the contents of the method/property empty and do **not** call the `SUPER^.Method()`.

```
METHOD Power : HRESULT
VAR_INPUT
    bCommand:        BOOL;
END_VAR
VAR
END_VAR
```

3. Optional: Add `{attribute 'hide'}` to the method/property to hide the method/property in the development environment.

⇨ You have successfully disabled a method/property.

6.2.3 Adapting inner procedures of a class (FB)

Some classes contain inner flows/procedures which are to be extended/changed by inheritance levels or adapted by the application. This is realized with the FB_AdaptableSequence [▶ 97] class.

1. Create a new class (FB) and let it inherit from a class with inner procedure.

```
FUNCTION_BLOCK FB_AdaptableClass EXTENDS FB_Extruder
VAR
END_VAR
```

2. Overwrite the internal callback method with the integrated procedure.
   In the case of the `FB_Extruder` class, the internal procedure is implemented in the `PowerStates()` method.

```
METHOD PROTECTED PowerStates
VAR_INPUT
END_VAR
```

3. Add a query to the callback method for its process step, whether you want to extend an existing sequence or add a new one, and if you have an existing implementation, whether you want to execute it or skip it.
4. Evaluate whether calling the existing implementation before or after its new implementation makes sense. Calling the SUPER^ method is only necessary if you continue to use the existing sequences.

```
// React on existing sequence state
IF aSeqBaseMembers[E_ExtruderPowerStates.eStartVeloFeed].IsActive THEN
    ;
    // Call return to replace exisiting implementation
    RETURN;
END_IF

// Define additional/replacing sequence state
IF fbSetProdTurnrate.IsActive THEN
    ;
    RETURN;
END_IF
Platzieren Sie den SUPER^ Aufruf der Callback Methode.
```

5. Define the condition at which the sequence is completed.

```
// React on existing sequence state
IF aSeqBaseMembers[E_ExtruderPowerStates.eStartVeloFeed].IsActive THEN
    // Set FB_AdaptableSequence interface locally
    iSeq := aSeqBaseMembers[E_ExtruderPowerStates.eStartVeloFeed];

    IF bAdditionalAction THEN
        nSaveValueToThis := 10;
        // command a jump to a state that is not the default "next" element
        iSeq.Jump(fbSetProdTurnrate);
        // feedback on finishing the sequence state
        iSeq.Done := TRUE;
    END_IF

    // Call return to replace exisiting implementation
    RETURN;
END_IF

// Define additional/replacing sequence state
IF fbSetProdTurnrate.IsActive THEN
    // Set FB_AdaptableSequence interface locally
    iSeq := fbSetProdTurnrate;

    IF bAdditionalAction THEN
        nSaveValueToThis := 10;
        iSeq.Done := TRUE;
    END_IF

    RETURN;
END_IF

// Call implementation of other sequence steps
SUPER^.PowerStates();
```

6. [Only when adding]: Instantiate an instance of type `FB_AdaptableSequence` in the class with the name of the sequence.

```
FUNCTION_BLOCK FB_AdaptableClass EXTENDS FB_Extruder
VAR
    fbSetProdTurnrate:        FB_AdaptableSequence;
END_VAR
```

7. Insert the sequence in the initialization at the desired position.

```
IF NOT F_SucceededHr(SUPER^.Init(), Init) THEN
    RETURN;
END_IF

fbPowerStates.Insert [▶ 101](
    iCurrent     := aSeqBaseMembers[E_ExtruderPowerStates.eMasterMode],
    iNew         := fbSetProdTurnrate,
    bOverwrite   := FALSE);
```

8. Apply the changes to your target system and restart the PLC.

⇨ You have successfully extended an inner procedure of a class.

6.2.4 Extending the HMI parallel class (FB)

In many use cases, the number of displayed values on the surface is to be extended. For this purpose, the HMI parallel class can be extended so that the new values are accessible in the HMI environment at a suitable location.

✓ For the complete implementation the extension of the base class (FB) is necessary.

1. Perform steps 1 to 5 of Base class extension (FB) for the base class.
2. Create a new class (FB) and remove VAR_INPUT and VAR_OUTPUT.

```
FUNCTION_BLOCK FB_CustomAxisHmi
VAR
END_VAR
```

3. Add the class to be inherited to the class definition using the EXTENDS keyword.

```
FUNCTION_BLOCK FB_CustomAxisHmi EXTENDS FB_AxisHmi
VAR
END_VAR
```

4. So that you can also address the class and the added elements via an interface, create an interface with the same name.

```
INTERFACE I_CustomAxisHmi
```

5. Let the interface inherit from the interface of the inherited class.

```
INTERFACE I_CustomAxisHmi EXTENDS I_AxisHmi
```

6. Implement the interface in the previously created class (FB).

```
FUNCTION_BLOCK FB_CustomAxisHmi EXTENDS FB_AxisHmi IMPLEMENTS I_CustomAxisHmi
VAR
END_VAR
```

7. Instantiate the interface in the base class (FB).

```
FUNCTION_BLOCK FB_CustomAxis EXTENDS FB_Axis IMPLEMENTS I_CustomAxis
VAR
    iCustomAxisHmi:        I_CustomAxisHmi;
END_VAR
```

8. Overwrite the SetHMI() method of the base class.

```
// Setter method for HMI-Class
METHOD SetHMI : HRESULT
VAR_INPUT
    ipBaseHmi:        I_BaseHmi;        // interface on hmi object
END_VAR



IF NOT __QUERYINTERFACE(ipBaseHmi, iCustomAxisHmi) THEN
    SetHMI := F_HresultFailure(E_AdsErr.DEVICE_INVALIDINTERFACE);
    RETURN;
END_IF

SetHmi := S_OK;
```

9. Extend the Init() method and add a __QUERYINTERFACE() operation for the new interface.

```
METHOD Init : HRESULT



IF NOT __QUERYINTERFACE(iCustomAxisHmi, iAxisHmi) THEN
```

```
    RETURN;
ELSIF NOT F_SucceededHr(SUPER^.Init(), Init) THEN
    RETURN;
END_IF
```

Add a new setting or command value for the HMI

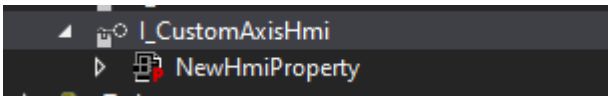
10. Add a new property to the HMI class.

```
PROPERTY NewHmiProperty : LREAL
```

11. Add the Monitoring attribute to the declaration to make the property visible to the HMI. This makes the property visible even when the PLC is logged in.

```
{attribute 'monitoring' := 'call'}
PROPERTY NewHmiProperty : LREAL
```

12. Copy the property to the interface with the same name.



13. Create a similarly named variable in the HMI class where the value can be cached.

```
FUNCTION_BLOCK FB_CustomAxisHmi EXTENDS FB_AxisHmi IMPLEMENTS I_CustomAxisHmi
VAR
    fNewHmiProperty:            LREAL;
END_VAR
```

14. Add the 'TcHmiSymbol.Hide' attribute to the declaration so that the variable is not seen by the HMI. This ensures that the variable is not mistakenly used by the HMI instead of the property. If you set the ADS mapping in the HMI to Use whitelisting, you can basically hide all variables. To continue showing the HMI classes, you must add the {attribute 'TcHmiSymbol.ShowRecursively'} to the declaration.

```
FUNCTION_BLOCK FB_CustomAxisHmi EXTENDS FB_AxisHmi IMPLEMENTS I_CustomAxisHmi
VAR
    {attribute 'TcHmiSymbol.Hide'}
    fNewHmiProperty:            LREAL;
END_VAR
```

15. Implement the property write and read operation in the *Get* and *Set* method.

⇨ You have successfully added a property.

Making an existing property of the base class accessible to the HMI

1. Instantiate an interface of the base class type in the HMI parallel class.

```
FUNCTION_BLOCK FB_CustomAxisHmi EXTENDS FB_AxisHmi IMPLEMENTS I_CustomAxisHmi
VAR
    {attribute 'TcHmiSymbol.Hide'}
    iCustomAxis:                I_CustomAxis;
    {attribute 'TcHmiSymbol.Hide'}
    fNewHmiProperty:            LREAL;
END_VAR
```

2. Add the Init() method to the HMI class.

```
// Init method for linking to a control class (FB)
METHOD Init : HRESULT
VAR_INPUT
    ipBase:    I_Base;    // Base interface on linked control class (FB)
END_VAR
```

3. In the Init() method, perform a `__QUERYINTERFACE()` operation from the base interface passed to the last interface instantiated.

```
Init := F_HresultFailure(E_AdsErr.DEVICE_NOTINIT);

RETURN(NOT __QUERYINTERFACE(ipBase, iCustomAxis));
RETURN(NOT F_SucceededHr(SUPER^.Init(ipBase), Init));

Init :=  F_HresultSuccess(NOERR);
```

4. [If inheriting from FB_BaseHmi] Implement the call to the HMI-Init() in the Init() method of the base class.

```
Init:=F_HresultFailure(E_AdsErr.DEVICE_NOTINIT);

IF iCustomAxisHmi = 0 THEN
    RETURN;
ELSIF NOT F_SucceededHr(iCustomAxisHmi.Init(THIS^), Init) THEN
    RETURN;
ELSIF NOT F_SucceededHr(SUPER^.Init(), Init) THEN
    RETURN;
END_IF
```

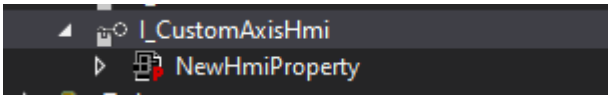
5. Add a new property to the HMI class.

```
PROPERTY SecondNewHmiProperty : LREAL
```

6. Add the Monitoring attribute to the declaration to make the property visible to the HMI.
   This makes the property visible even when the PLC is logged in.

```
{attribute 'monitoring' := 'call'}
PROPERTY NewHmiProperty : LREAL
```

7. Copy the property to the interface with the same name.



8. Program the access to the property of the interface in the *Get* (and if necessary *Set*) method of the property.

```
IF iCustomAxis <> 0 THEN
    SecondNewHmiProperty := iCustomAxis.ExistingValue;
END_IF
```

9. For further properties with reference access, steps 1 to 4 are omitted accordingly.

⇨ You have successfully added a property with reference access to the main class.

6.3 Axes

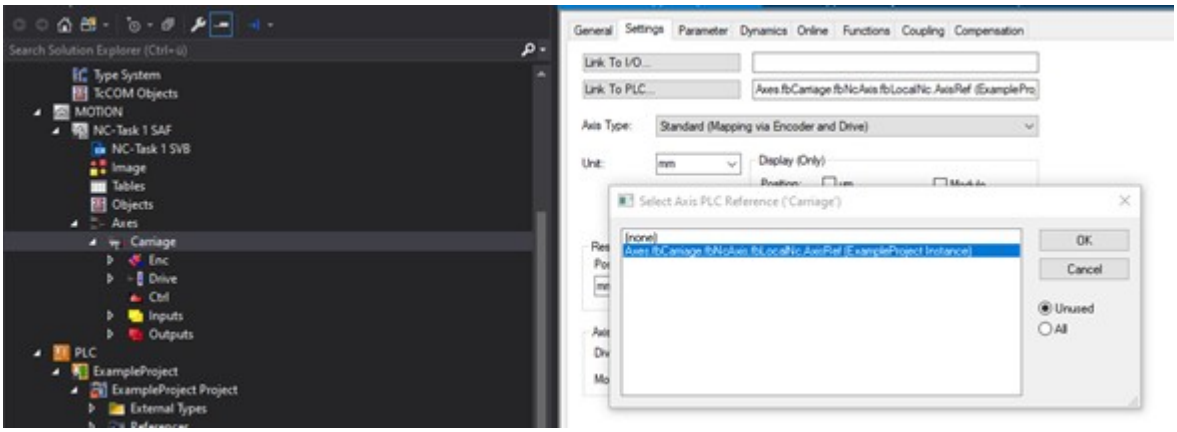
6.3.1 Creating and initializing NC axis

There are several ways to create an NC axis. One possible procedure is shown below as an example.

1. Create the three instances of the following objects:
   fbCarriageAxis (FB_AxisNcBase): TF8560 axis (here NC)
   fbCarriage (FB_Carriage): Specifically implemented axis type (here carriage axis)
   fbCarriageHmi (FB_CarriageHmi): Parallel HMI interface of the axis type (here carriage axis)

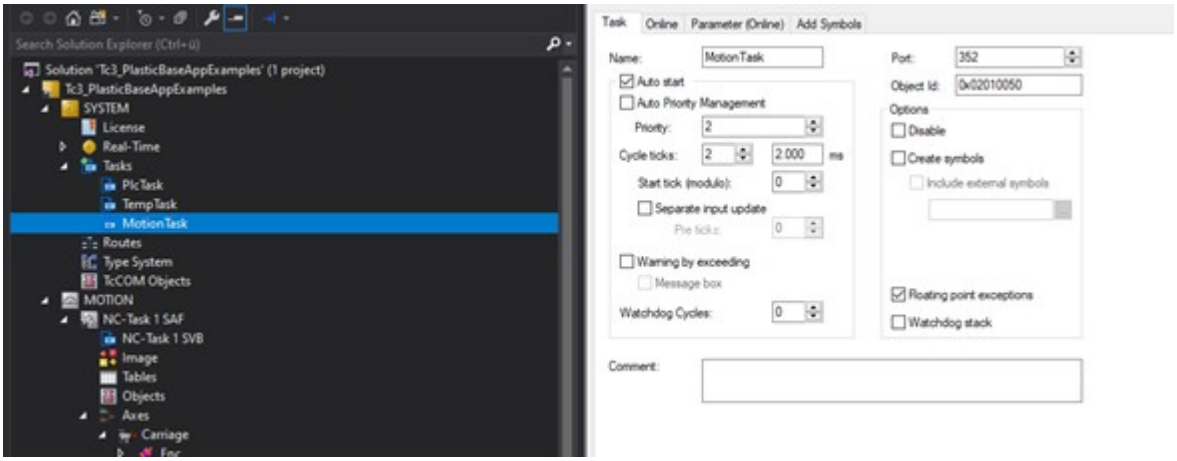
```
{attribute 'qualified_only'}
VAR_GLOBAL
    {attribute 'TcHmiSymbol.Hide'}
    {attribute 'TcContextName':='MotionTask'}
    fbCarriageAxis:        FB_AxisNcBase('', Tc3_PlasticBaseAppStaticParams.cnMaxPtpPoints, 0, 0
, 0);
    {attribute 'TcHmiSymbol.Hide'}
    fbCarriage:            FB_Carriage;

    fbCarriageHmi:         FB_CarriageHmi;
END_VAR
```

2. Create/link an NC axis in the project to the created instance of the `fbCarriage` axis.



If the TF8560 axes from the PLC do not appear in the selection dialog, the project has not been created. Only once the PLC project has been successfully created after the instance has been created, the instances become visible in the mapping.

3. It is recommended that you create a PLC task with the same cycle time of the NC (default 2 ms) for the drive control part (if an axis does not already exist).



4. Assign a TF8560 axis to the specific axis in the PLC.

```
Axes.fbCarriage.SetAxisRef(Axes.fbCarriageAxis);
```

5. Add the axis to the runtime.

```
Tasks.fbRuntime.Append(Axes.fbCarriage, Axes.fbCarriageHmi);
```

If you do not want to work with FB_BaseRuntime, the following further steps are necessary:

6. Assign the HMI interface to the specific axis.

```
Axes.fbCarriage.SetHMI(Axes.fbCarriageHmi);
```

7. Initialize the axis with a single call to the `Init()` method and check the return values.

```
IF NOT bInit THEN
    bInit := TRUE;

    Axes.fbCarriage.SetAxisRef(Axes.fbCarriageAxis);
    Axes.fbCarriage.SetHMI(Axes.fbCarriageHmi);

    hr             := Axes.fbCarriage.Init();
    bInitFailed    := FAILED(hr);
END_IF
```

8. After successful initialization, call the `CoreCyclic()` method of the axis with a fast task.

```
IF NOT PRG_AxisApplication.bInitFailed THEN

        Axes.fbCarriage.CoreCyclic();
END_IF
```

9. In parallel with the `CoreCyclic()` method, call the `Cyclic()` method in a slower task and initialize the default parameterization using the `ParamInit()` method.

```
VAR
    bInit:                  BOOL;
    bInitFailed:            BOOL;
    bParamInit:             BOOL;
    hr:                     HRESULT;
END_VAR



IF NOT bInit THEN
    bInit := TRUE;

    Axes.fbCarriage.SetAxisRef(Axes.fbCarriageAxis);
    Axes.fbCarriage.SetHMI(Axes.fbCarriageHmi);

    hr             := Axes.fbCarriage.Init();
    bInitFailed := FAILED(hr);

ELSIF NOT bInitFailed THEN
    IF NOT bParamInit THEN
        hr := Axes.fbCarriage.ParamInit();
        bParamInit := SUCCEEDED(hr);
    END_IF

    Axes.fbCarriage.Cyclic();
END_IF
```

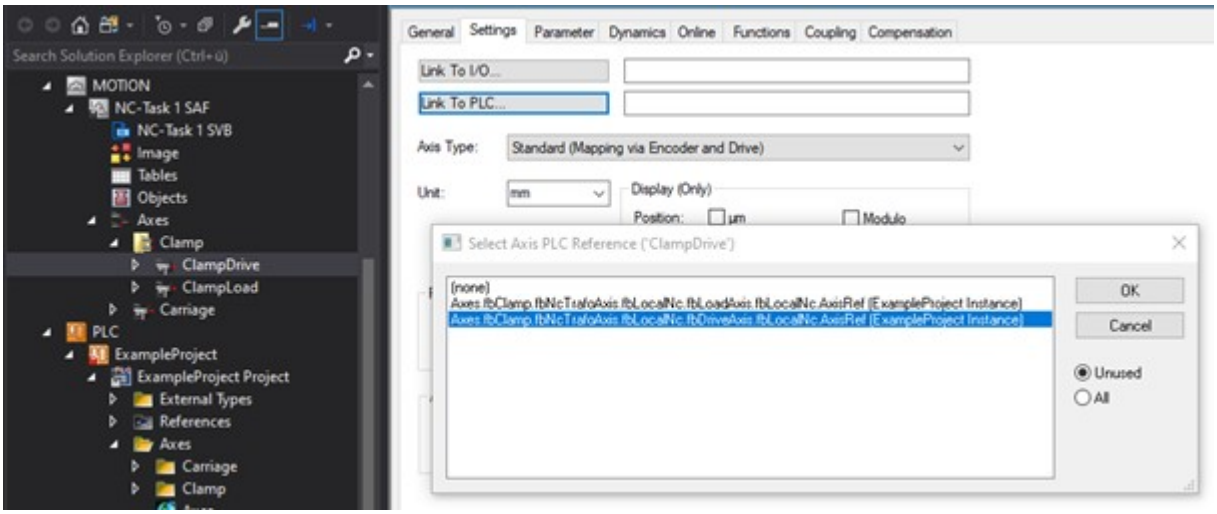
6.3.2 Creating and initializing NC transformation axis

Several steps are necessary to create a new NC transformation axis. One possible procedure is shown below as an example.

1. Create the three instances of the following objects:
   fbClampAxis (FB_AxisNcTrafoBase): TF8560 axis (here transformation NC)
   fbClamp (FB_Clamp): Specifically implemented axis type (here clamping unit)
   fbClampHmi (FB_ClampHmi): Parallel HMI interface of the axis type (here clamping unit).

```
{attribute 'qualified_only'}
VAR_GLOBAL
    {attribute 'TcHmiSymbol.Hide'}
    {attribute 'TcContextName':='MotionTask'}
    fbClampAxis:        FB_AxisNcTrafoBase('', Tc3_PlasticBaseAppStaticParams.cnMaxPtpPoints, Tc
3_PlasticBaseAppStaticParams.cnNoOfTrafoPoints, 0, 0, 0);
    {attribute 'TcHmiSymbol.Hide'}
    fbClamp:            FB_Clamp;

    fbClampHmi:         FB_ClampHmi;
END_VAR
```

2. Create/link two NC axes in the project for the drive and load sides of the created instance of the `fbClamp` axis.



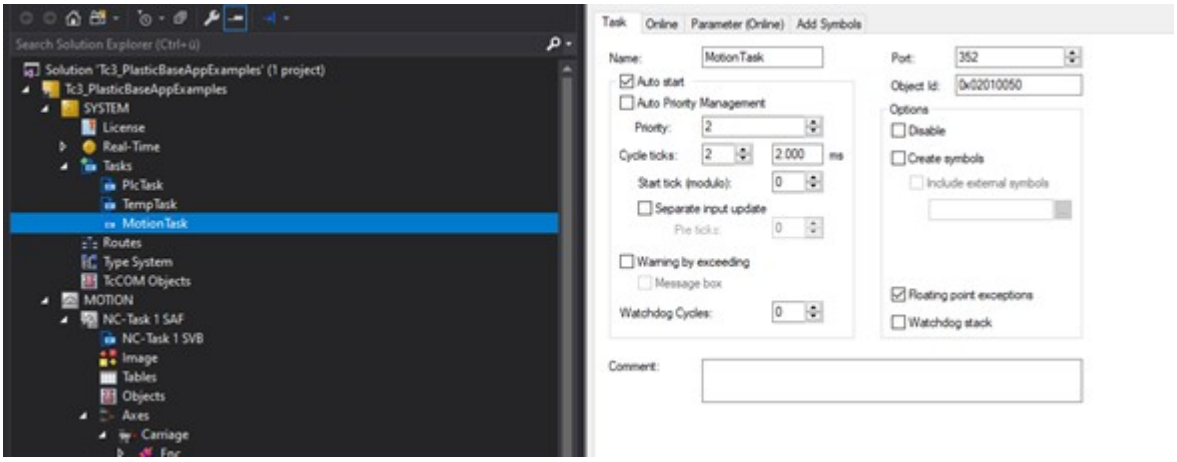
If the TF8560 axes from the PLC do not appear in the selection dialog, the project has not been created. Only once the PLC project has been successfully created after the instance has been created, the instances become visible in the mapping.

3. Set the following parameterization on the NC axis of the load side:
   - Axes > ClampLoad > Settings > Axis Type: = Standard
   - Axes > ClampLoad > Enc > NC-Encoder > Type: = Encoder SSI
   - Axes > ClampLoad > Enc > Parameter > Scaling Factor Numerator = 0.0001
   - Axes > ClampLoad > Enc > Parameter > Scaling Factor Denominator = 1.0
   - Axes > ClampLoad > Enc > Parameter > Position Bias = -1000.0
   - Axes > ClampLoad > Enc > Parameter > Encoder Mode = POSVELOACC

And map the following variables additionally between NC and PLC:

- Axes > ClampLoad > Enc.Inputs.In.nDataIn1 <--> GVL_Xyz.fbNcTrafoAxis.fbLocalNc.fbActuals.nDataIn1
- Axes > ClampLoad > Enc.Inputs.In.nState4 <--> GVL_Xyz.fbNcTrafoAxis.fbLocalNc.fbActuals.nState4

4. It is recommended that you create a PLC task with the same cycle time of the NC (default 2 ms) for the drive control part (if an axis does not already exist).



5. Additionally, instantiate a Table Generator in your application to create a transformation table.

```
VAR
    fbTableGen:                FB_TableGeneratorClampStandard_1;
END_VAR
```

6. Assign a TF8560 axis to the specific axis in the PLC.

```
Axes.fbClamp.SetAxisRef(Axes.fbClampAxis);
```

7. Assign the Table Generator to the specific axis in the PLC.

```
Axes.fbClamp.Specific.Trafo.TableGenerator := fbTableGen;
```

8. Define the geometry of the mechanics to use the transformation function. In the case of the clamping unit, a standard clamping mechanism is parameterized in this sample.

```
Method DefineTable : HRESULT



// Assign geometries
fbTableGen.BaseDistance     := 672.0;    // [mm]
fbTableGen.DriveArm         := 228.0;    // [mm]
fbTableGen.LoadArm          := 325.2;    // [mm]
fbTableGen.ToolArm_1        := 602.52;   // [mm]
fbTableGen.ToolArm_2        := 455.4;    // [mm]
fbTableGen.ToolArm_3        := 114.0;    // [mm]
fbTableGen.ToolArm_Angle    := 216.0;    // [mm]
fbTableGen.ToolOffset       := 288.0;    // [mm]
fbTableGen.DriveLowEnd      := 0.0;      // [°]
fbTableGen.DriveHighEnd     := 180.0;    // [°]

// calculate resulting point table
IF NOT fbTableGen.DefineTable() THEN
    DefineTable:= F_HresultFailure(E_AdsErr.DEVICE_INVALIDPARM);
    RETURN;
END_IF
```

```
// copy parameter to machine data file
fbTableGen.WriteToParamList();

// activates table in TF8560 axis
F_SucceededHr(Axes.fbClamp.Specific.Trafo.AssignTableToAxis(FALSE), DefineTable);
// copies table drive ends to nc-Softends
F_SucceededHr(Axes.fbClamp.Specific.Trafo.CopyTableDriveEnds(FALSE, FALSE), DefineTable);
```

9. Add the axis to the runtime.

```
Tasks.fbRuntime.Append(Axes.fbClamp, Axes.fbClampHmi);
```

If you do not want to work with FB_BaseRuntime, the following further steps are necessary:

10. Assign the HMI interface to the specific axis.

```
Axes.fbClamp.SetHMI(Axes.fbClampHmi);
```

11. Initialize the axis with a single call to the `Init()` method and check the return value.

```
IF NOT bInit THEN
    bInit := TRUE;

    Axes.fbClamp.SetAxisRef(Axes.fbClampAxis);
    Axes.fbClamp.SetHMI(Axes.fbClampHmi);

    hr             := Axes.fbClamp.Init();
    bInitFailed    := FAILED(hr);
END_IF
```

12. After successful initialization, call the `CoreCyclic()` method of the axis with a fast task.

```
IF NOT PRG_AxisApplication.bInitFailed THEN

        Axes.fbClamp.CoreCyclic();
END_IF
```

13. In parallel with the `CoreCyclic()` method, call the `Cyclic()` method in a slower task and initialize the default parameterization using the `ParamInit()` method.

```
VAR
    bInit:                  BOOL;
    bInitFailed:            BOOL;
    bParamInit:             BOOL;
    hr:                     HRESULT;

    fbTableGen:             FB_TableGeneratorClampStandard_1;
END_VAR



IF NOT bInit THEN
    bInit := TRUE;

    Axes.fbClamp.SetAxisRef(Axes.fbClampAxis);
    Axes.fbClamp.Specific.Trafo.TableGenerator := fbTableGen;
    Axes.fbClamp.SetHMI(Axes.fbClampHmi);

    hr             := Axes.fbClamp.Init();
    bInitFailed := FAILED(hr);

ELSIF NOT bInitFailed THEN
    IF NOT bParamInit THEN

        hr := DefineTable();
        bParamInit := SUCCEEDED(hr);
        hr := Axes.fbClamp.ParamInit();
        bParamInit := bParamInit AND SUCCEEDED(hr);
    END_IF

    Axes.fbClamp.Cyclic();
END_IF
```

6.3.3 Integrating manual function into an axis

A manual function can be created in several places in the application. The following sample explains how a manual function for moving and switching on the axis can be integrated into it.

1. Extend an existing class by inheritance. In this sample, the `FB_Carriage` class is used for this purpose.

```
FUNCTION_BLOCK FB_CustomCarriage EXTENDS FB_Carriage
```

2. Instantiate the `FB_ManualXyz` manual function in the class of the axis. This is not necessary for the manual function of the forward/backward movement, since this is already instantiated by the `FB_Carriage` class.

```
FUNCTION_BLOCK FB_CustomCarriage EXTENDS FB_Carriage
VAR
    fbManualPower:        FB_ManualPower;
END_VAR
```

3. The `FB_CarriageHmi` class used here already provides an HMI interface for both manual functions. For more manual functions, you need to extend the class with respective instances of the `FB_ManualFunctionHmi` class.
4. Afterwards, the newly instantiated manual function must be initialized. For convenience, you can use the `F_SucceededHr()` function to check the return value of the initialization function for success and copy it to a local variable.

```
METHOD Init : HRESULT



IF NOT F_SucceededHr(fbManualPower.Init(THIS^, iCarriageHmi), Init) THEN
    RETURN;
END_IF

Init := SUPER^.Init();
```

5. To process the inner algorithms of the manual functions, you must call the respective cycle methods in `Cyclic()`.

```
METHOD Cyclic



fbManualPower.Cyclic();
fbManualForBack.Cyclic();
```

6. In the same method you can additionally define when the manual function should be activated for use and/or which method is called by the manual function.

```
fbManualPower.Enable := bManualMode OR bSetupMode OR bAutomaticMode;
fbManualPower.Cyclic();

fbManualForBack.Enable := bManualMode OR bSetupMode;
fbManualForBack.Cyclic();

IF bSetupMode THEN
    IF fbManualForBack.TrigCmdWorkPos.RQ THEN
        JogNegative(TRUE);
    ELSIF fbManualForBack.TrigCmdWorkPos.FQ THEN
        JogNegative(FALSE);
    ELSIF fbManualForBack.TrigCmdBasePos.Q THEN
        JogPositive(fbManualForBack.TrigCmdBasePos.Q);
    END_IF

ELSIF bManualMode THEN
    IF fbManualForBack.TrigCmdWorkPos.Q THEN
        MovePtp(1, fbManualForBack.TrigCmdWorkPos.Q, 8);
    ELSIF fbManualForBack.TrigCmdBasePos.Q THEN
        MovePtp(2, fbManualForBack.TrigCmdBasePos.Q, 8);
    END_IF
END_IF
```

6.4 Data management

6.4.1 Creating machine data

At the start time of a project, an initial set of machine data must be created so that the parameters determined during commissioning can be persistently stored.

Error messages in the EventLogger

All runtime objects attached to the `FB_BaseRuntime` will inevitably trigger an error message under the following conditions:

- All `Init()` methods have been successfully executed
- No machine data file exists yet
- `FB_BaseRuntime.DisableMdInit` is not set

✓ For the sequence of steps described below, the use of the `FB_BaseRuntime` is assumed.

1. Start the PLC runtime.
2. Call the `FB_BaseRuntime.MdSaveAll()` method **once**.
3. Wait for the saving process to be completed.
   Depending on the amount of data to be saved, the processing time may vary. In PLC online mode the variable bMdSaveAll can be monitored to check the completion of the saving process.

⇨ You have successfully generated a first set of machine data

6.4.2 Integrating recipe release

Since in certain operating scenarios (e.g. in automatic mode) no recipes are to be loaded into the PLC, the release of the recipe management must be given by the PLC.

✓ The recipe release requires an initialization after machine start. For this purpose, the property `FB_PlcStateToHmi.PlcInitialized` must be set.

1. Write the Boolean condition on the following property: `FB_PlcStateToHmi.ProductChangeEnable`

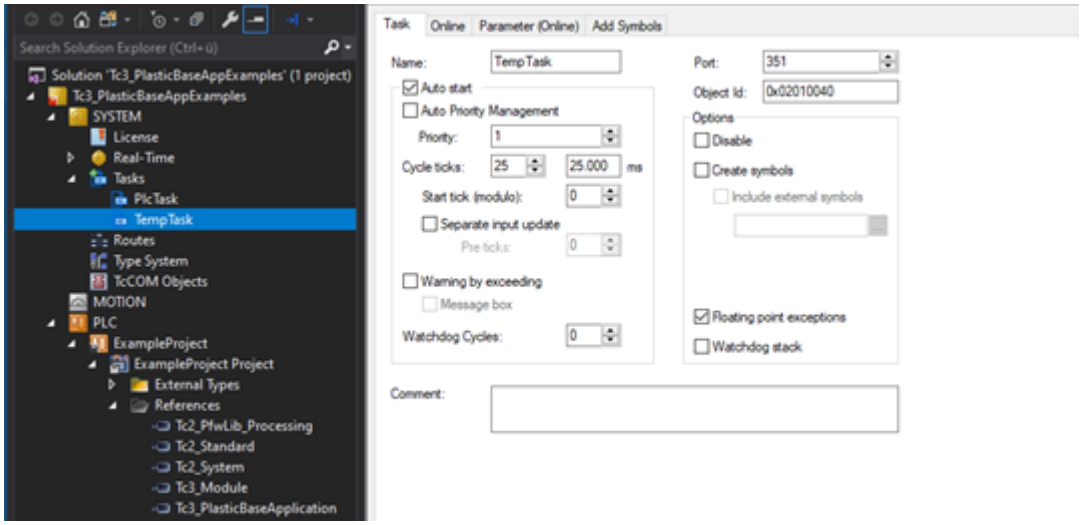
⇨ If the Boolean condition is true, recipes can be loaded into the PLC.

6.5 Temperature control

6.5.1 Instantiating and initiating temperature control

For certain applications, only the temperature control should be used from the scope of the TC3 Plastic Application. In this, but also in the normal use case, the temperature controller must be capable of being instantiated and initiated individually for this purpose. In the following sample, we will go through this process step by step.

1. Create a new task with a cycle time of 25 ms. This is not mandatory, but - based on experience - recommended.



2. Create an instance of `FB_TempCtrl` and `FB_TempCtrlHmi` and assign the mapping to the newly created task.

```
{attribute 'qualified_only'}
VAR_GLOBAL
    {attribute 'TcHmiSymbol.Hide'}
    {attribute 'TcContextName':='TempTask'}
    fbTempCtrl:             FB_TempCtrl;

    fbTempCtrlHmi:          FB_TempCtrlHmi;
END_VAR
```

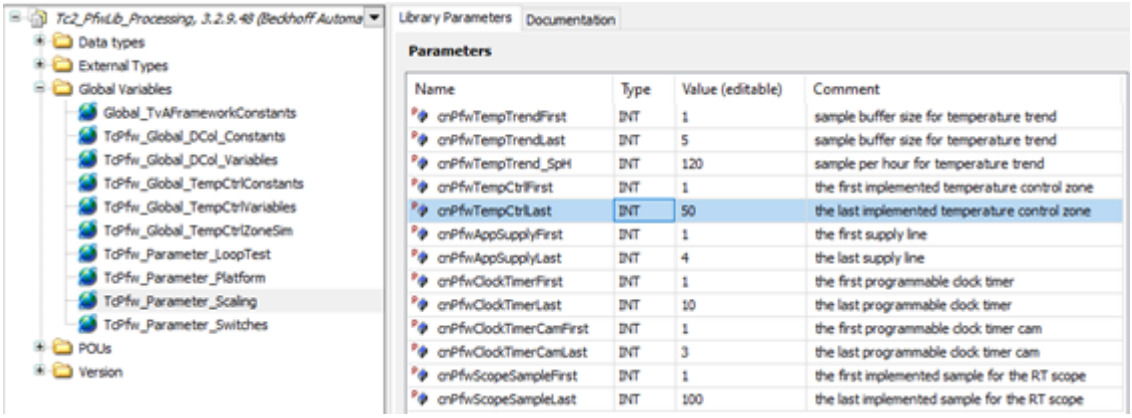
3. Call the `SetHMI()` and `Init()` method of `FB_TempCtrl` and check the return value for successful execution. If this is the case, you can call the `ParamInit()` method for parameter initialization and the cycle method.

```
VAR
    bInit:          BOOL;
    bInitFailed:    BOOL;
    bParamInit:     BOOL;
    hr:             HRESULT;
END_VAR



IF NOT bInit AND NOT bInitFailed THEN
    hr := fbTempCtrl.SetHMI(fbTempCtrlHmi);
    hr := fbTempCtrl.Init();
    IF SUCCEEDED(hr) THEN
        bInit := TRUE;
    ELSE
        bInitFailed := TRUE;
    END_IF
ELSIF bInit THEN
    IF NOT bParamInit THEN
        hr := fbTempCtrl.ParamInit();
        IF SUCCEEDED(hr) THEN
            bParamInit := TRUE;
        END_IF
    END_IF

    fbTempCtrl.Cyclic();
END_IF
```

6.5.2 Mapping and configuration of temperature zones

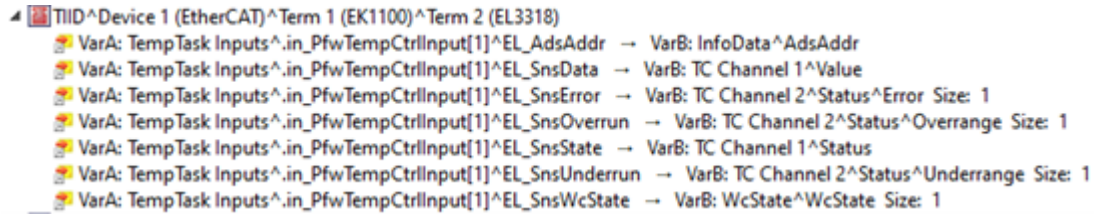
A few individual steps must be taken in the engineering to commission the temperature control. After that, setup and configuration is possible purely on the user interface. The following step-by-step instructions show you the setting necessities in TwinCAT Engineering.

1. Configure the usable number of zones in the Tc2_PfwLib_Processing.
   References > Tc2_PfwLib_Processing > Global Variables > TcPfw_Parameter_Scaling > cnPfwTempCtrlLast



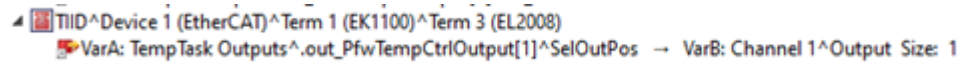
The library included in the Tc3_PlasticBaseApplication cannot be adjusted! It is necessary to add the Tc2_PfwLib_Processing library to your project to adjust the number of temperature zones.

2. Link their analog inputs of the temperature sensors to the array `.in_PfwTempCtrlInput[]`.



After setting the number of zones, the project must be created successfully so that the mapping is updated, otherwise the mapping will not have the set number of zones.

3. Link your digital outputs of the heating and cooling relays to the array `.out_pfwTempCtrlOutput[]`.



4. Group the linear arrangement of zones to match their application. You have the following options here:
   - A contiguous section of the arrangement shall be assigned to a group of temperature zones. In this sample, zones 1 to 10 are assigned to group 1.

```
hr := fbTempCtrl.LinkGroup(nStartIdx := 1, nEndIdx := 10, nGroupIdx := 1, bOverwrite := FALSE)
;
```

   - The devices of a temperature zone group are scattered over the array and must be individually assigned to the group. In this sample, zone 12 and zone 20 are assigned to group 2.

```
hr := fbTempCtrl.LinkZone(nLinearIdx := 12, nGroupIdx := 2, nGroupMemberIdx := 1, bOverwrite :
= FALSE);
hr := fbTempCtrl.LinkZone(nLinearIdx := 20, nGroupIdx := 2, nGroupMemberIdx := 2, bOverwrite :
= FALSE);
```

6.5.3 Commissioning of the temperature control

The commissioning of the temperature control includes both a TwinCAT Engineering part and a part to be performed at runtime. In this sample the individual steps that are performed at runtime are summarized.

Commissioning via HMI recommended

This sample describes the procedure solely via the PLC path. Use the section Commissioning of the temperature control [▶ 194] for commissioning using the HMI.

Before the subsequent steps, the preparation of the TwinCAT project has to be done according to the example from section Mapping and configuration of temperature zones [▶ 135].

Set the appropriate input and output signal types and devices for all their temperature zones

Hardware parameterization without grouping

This step refers to the linear mapping of the TF8540 library. Already configured groupings are ignored in this step.

1. Create a program section to assign the parameters once.
2. Assign the following parameters according to your hardware configuration.

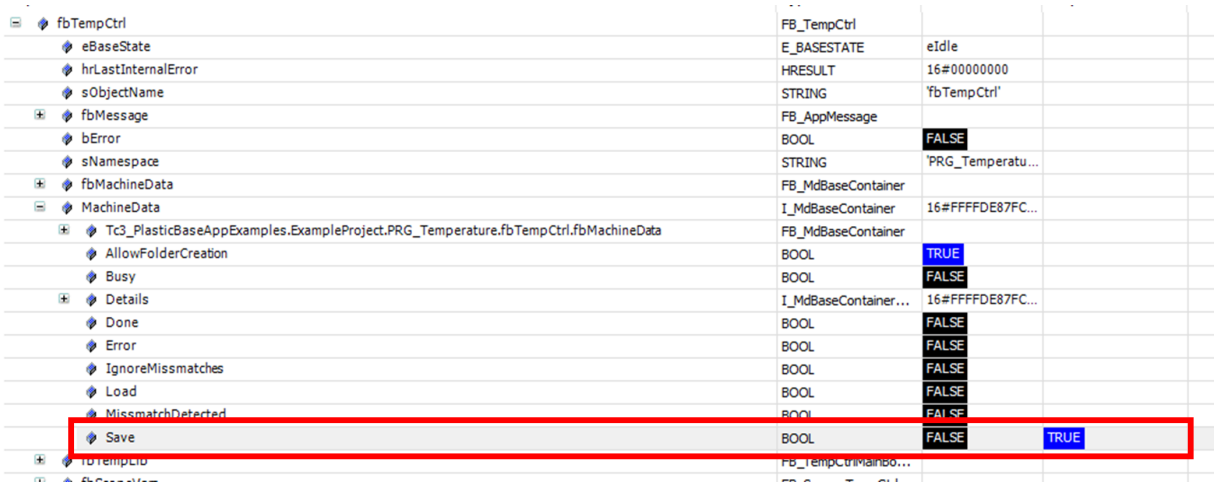
```
// set cooling output type to 'no cooling'
fbTempCtrlHmi.ParamTempZone[1].OutputSel_C  := E_TcPfw_TctrlOutSelect.eTcPfwTcOut_NoSignal;
// set heating output type to 'pwm'
fbTempCtrlHmi.ParamTempZone[1].OutputSel_H  := E_TcPfw_TctrlOutSelect.eTcPfwTcOut_PWM;
// set sensor terminal type to 'EL3314'
fbTempCtrlHmi.ParamTempZone[1].TempSensTerm := E_TcPfw_TerminalType.eTcPfwTermT_EL331x;
// set sensor type to 'ThermoCouple Typ K'
fbTempCtrlHmi.ParamTempZone[1].SensorType   := E_TcPfw_TempSensType.eTcPfwTempSensT_TC_K;
// set channel of the sensor on the linked terminal to 'Channel 1'
fbTempCtrlHmi.ParamTempZone[1].TermChannel  := 1;
```

3. Set the 'InUse' flag to validate the use of the zone.

```
fbTempCtrlHmi.ParamTempZone[1].InUse := TRUE
```

4. Repeat step 2 and 3 for all used zones
5. Execute the created code segment once

6. Log on to the controller
7. Save the parameters via `fbTempCtrl.MachineData.Save`



Check the reaction of the hardware inputs on the machine

1. Log on to the controller
2. Go in the tree structure `fbTempCtrlHmi.fbGroups[...].fbZones[...]` to the first used zone (`nIndex <> 0`)
3. Heat the sensor of the zone via an external heat source
4. Observe via the value `ActualTemperature` whether the temperature change occurs in the expected zone
   Repeat step 2 to 4 for each zone

Check the response of the hardware outputs on the machine

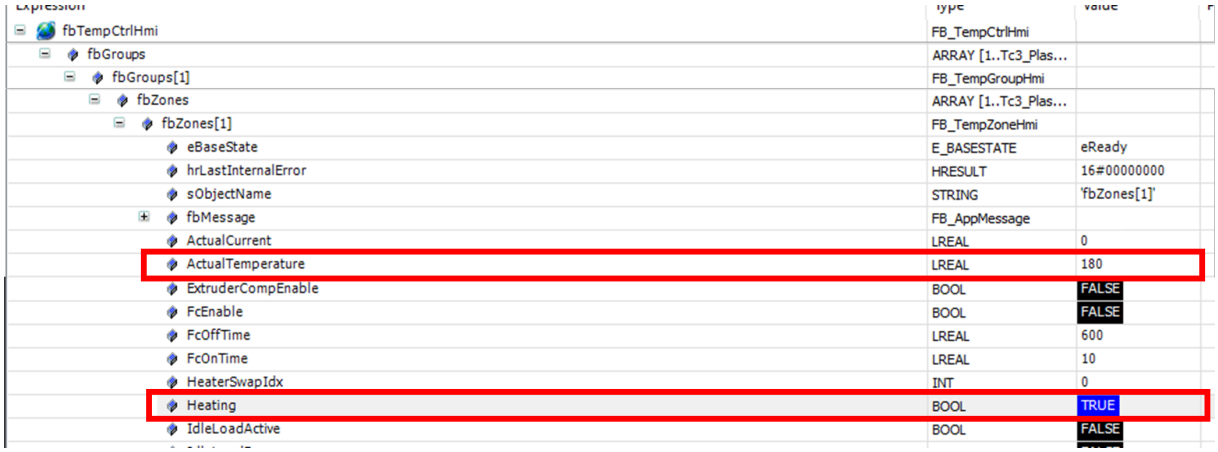
Switching on a zone does not generate a power level

Make sure that at the time of this step the temperature control has been enabled by the PLC!

- To enable all zones execute the method FB_TempCtrl.EnableAll(...) [▶ 51]
- To enable exactly one group execute the method FB_TempCtrl.Groups[...].Enable(...) [▶ 56]

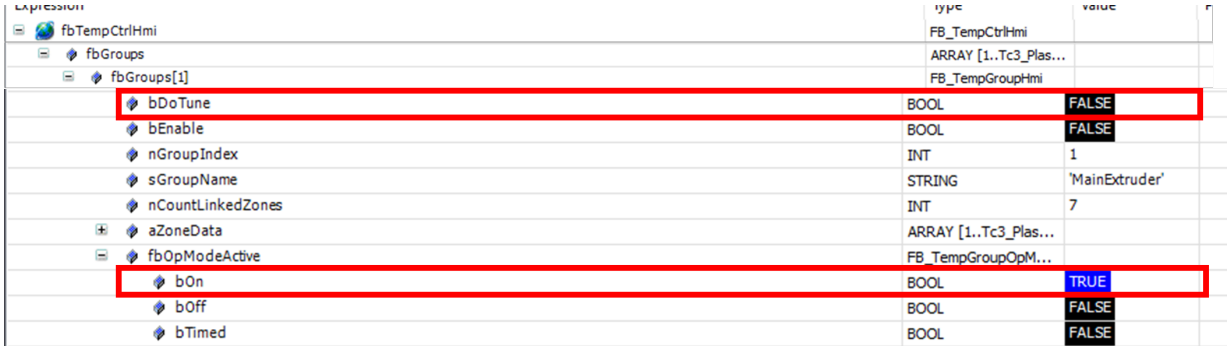
1. Switch on a single temperature zone via `fbTempCtrlHmi.fbGroups[...].fbZones[...].Enable`
2. Check if in the same zone the value of the variable `Heating` = `TRUE` and the value of the variable `ActualTemperature` (3) changes
3. Switch the zone off again as soon as possible to keep the temperature rise to a minimum

4. Repeat steps 1 to 3 for each zone



Start the automatic tuning of the control parameters

1. Switch on a temperature group via `fbTempCtrlHmi.fbGroups[…].fbOpModeActive.On`
2. Activate the tuning of a group via `fbTempCtrlHmi.fbGroups[…].DoTune` = TRUE directly afterwards
3. Carry out steps 1 and 2 for all groups to be commissioned



Monitor the automatic tuning until it completes successfully

As soon as the value of the variable fbTempCtrlHmi.fbGroups[...].TuningActive is reset the tuning of the group is finished

The value of the variable fbTempCtrlHmi.fbGroups[...].TuningDone indicates whether the tuning was successful or not

You have successfully commissioned your temperature control

7 HMI project structure

The Plastic Application HMI project is a project developed in TwinCAT HMI with basic functionalities for creating user interfaces for plastics processing machines. The project structure is based on the standard of a TwinCAT HMI project.

Use and further development of the Plastic Application HMI

Since the Plastic Application HMI project does not provide a fully comprehensive user interface for every plastics processing machine, the project must be adapted to specific machines. For this purpose, the project can be extended and modified as desired.

Update capability of the Plastic Application HMI

Editing the Plastic Application HMI can lead to an impairment of the update capability. This may affect the support provided by Beckhoff Automation.

7.1 References

NuGet packages can be added to the project under the References folder via the NuGet package management system.

TF8550 | Plastic HMI Framework

The following NuGet packages from the product can be used specifically for plastics processing machines:

NuGet package	Description
Beckhoff.TwinCAT.HMI.Plastic.Controls	Addition to the NuGet package Beckhoff.TwinCAT.HMI.Controls to extend the toolbox with specific controls for plastic processing machines (e.g. ArrowMotionGraph, CamControl, PfwSingleTempControl etc.).
Beckhoff.TwinCAT.HMI.Plastic.Functions	Complementary functions for the project.
Beckhoff.TwinCAT.HMI.Plastic.Images	Contains the icons and graphics used in the Plastic Application HMI project, as well as other resources for plastic processing machines.
Beckhoff.TwinCAT.HMI.Plastic.Localizations	Contains the language keys used in the Plastic Application HMI project with translations in English and German specifically for plastics processing machines.
Beckhoff.TwinCAT.HMI.Plastic.RecipeHelper	Addition to the NuGet package Beckhoff.TwinCAT.HMI.RecipeManagement to extend the Recipe Management extension.
Beckhoff.TwinCAT.HMI.Plastic.Temperature	Addition to the NuGet package Beckhoff.TwinCAT.HMI.Plastic.Controls to extend the toolbox with controls and functions in the temperature area.
Beckhoff.TwinCAT.HMI.Plastic.Themes	Adds a Plastic theme to the themes and includes special design adjustments for the Plastic Application HMI project.

Using TwinCAT HMI with the TF8550 NuGet packages

The NuGet packages of the TF8550 product can be integrated into any TwinCAT HMI project. For this purpose, the corresponding license for TF8550 is required in the PLC. The package provides the NuGet packages described in chapters. The NuGet packages can be included via the NuGet package management system. To do this, the package source under which the NuGet packages are located must be included. After that, the NuGet packages can be located and installed via the search. The installed NuGet packages are available in the project after a restart of the project.

The **Configurator control** is part of the `Beckhoff.TwinCAT.HMI.Plastic.Controls` NuGet package and can be dragged onto a view or content via the toolbox. The control is used to set specific functions and the appearance of some other controls that are in the same NuGet package. The control only needs to exist with the desired parameters, but has no other functionality and therefore does not need to be visible.

Requirements for TC3 Plastic Application

Version	TE2000 | TwinCAT 3 HMI Engineering	TF8550 | Plastic HMI Framework
v12.6.0	v12.760.44 • Beckhoff.TwinCAT.HMI.Controls (12.760.44) • Beckhoff.TwinCAT.HMI.EventLogger (19.0.102) • Beckhoff.TwinCAT.HMI.Framework (12.760.44) • Beckhoff.TwinCAT.HMI.Functions (12.760.44) • Beckhoff.TwinCAT.HMI.RecipeManagement (19.0.102) • Beckhoff.TwinCAT.HMI.ResponsiveNavigation (12.760.44) • Beckhoff.TwinCAT.HMI.SqliteHistorize (19.0.102)	v12.8.0 • Beckhoff.TwinCAT.HMI.Plastic.Controls (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.Functions (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.Images (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.Localizations (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.RecipeHelper (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.Temperature (12.8.0) • Beckhoff.TwinCAT.HMI.Plastic.Themes (12.8.0)
v12.5.1	v12.758.8 • Beckhoff.TwinCAT.HMI.Controls (12.758.8) • Beckhoff.TwinCAT.HMI.EventLogger (19.0.0) • Beckhoff.TwinCAT.HMI.Framework (12.758.8) • Beckhoff.TwinCAT.HMI.Functions (12.758.8) • Beckhoff.TwinCAT.HMI.RecipeManagement (19.0.0) • Beckhoff.TwinCAT.HMI.ResponsiveNavigation (12.758.8) • Beckhoff.TwinCAT.HMI.SqliteHistorize (19.0.0)	v12.6.0 • Beckhoff.TwinCAT.HMI.Plastic.Controls (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.Functions (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.Images (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.Localizations (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.RecipeHelper (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.Temperature (12.6.0) • Beckhoff.TwinCAT.HMI.Plastic.Themes (12.6.0)

7.2 Themes

In the folder *Themes* you will find the Plastic theme. The Plastic theme references the `Beckhoff.TwinCAT.HMI.Plastic.Themes` NuGet package and includes design adaptations of existing controls and classes used in the Plastic Application HMI project. The file can be customized to change the design as desired, as long as it has been loaded into the project.

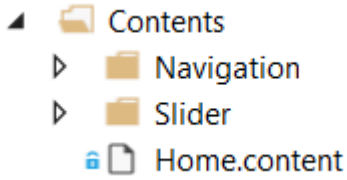
Update capability of the Plastic Application HMI

Editing the Plastic Application HMI can lead to an impairment of the update capability. This may affect the support provided by Beckhoff Automation.

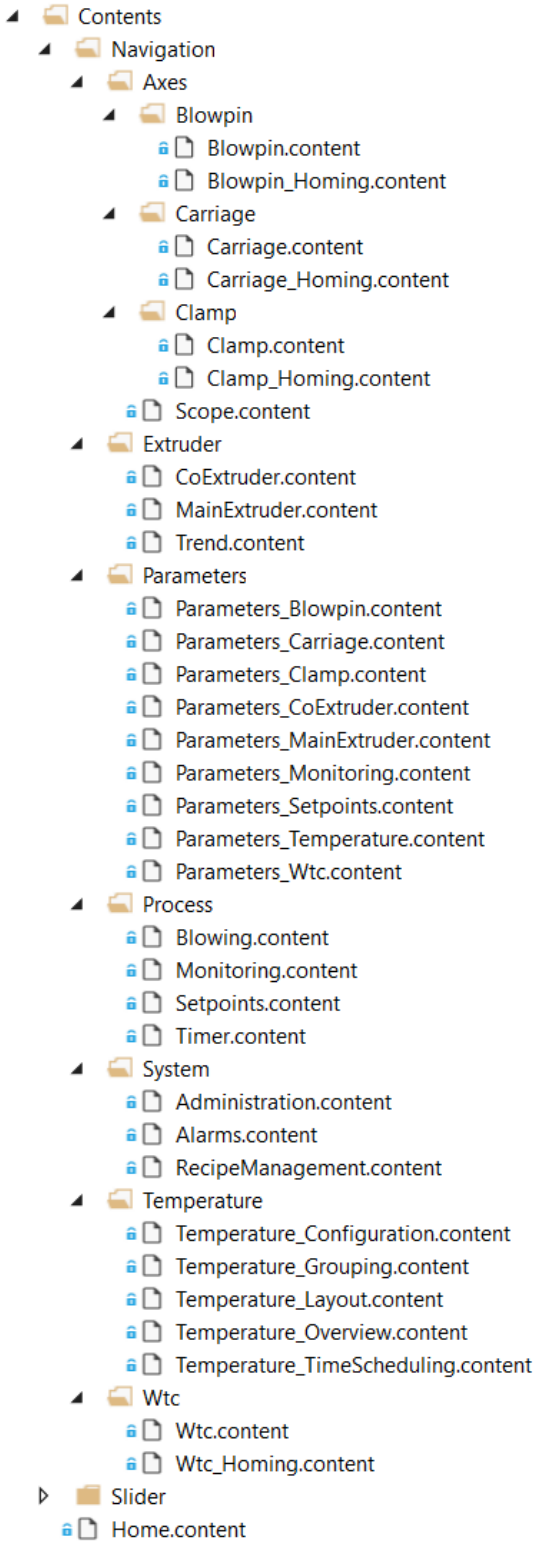
Available from version 12.5.1

7.3 Contents

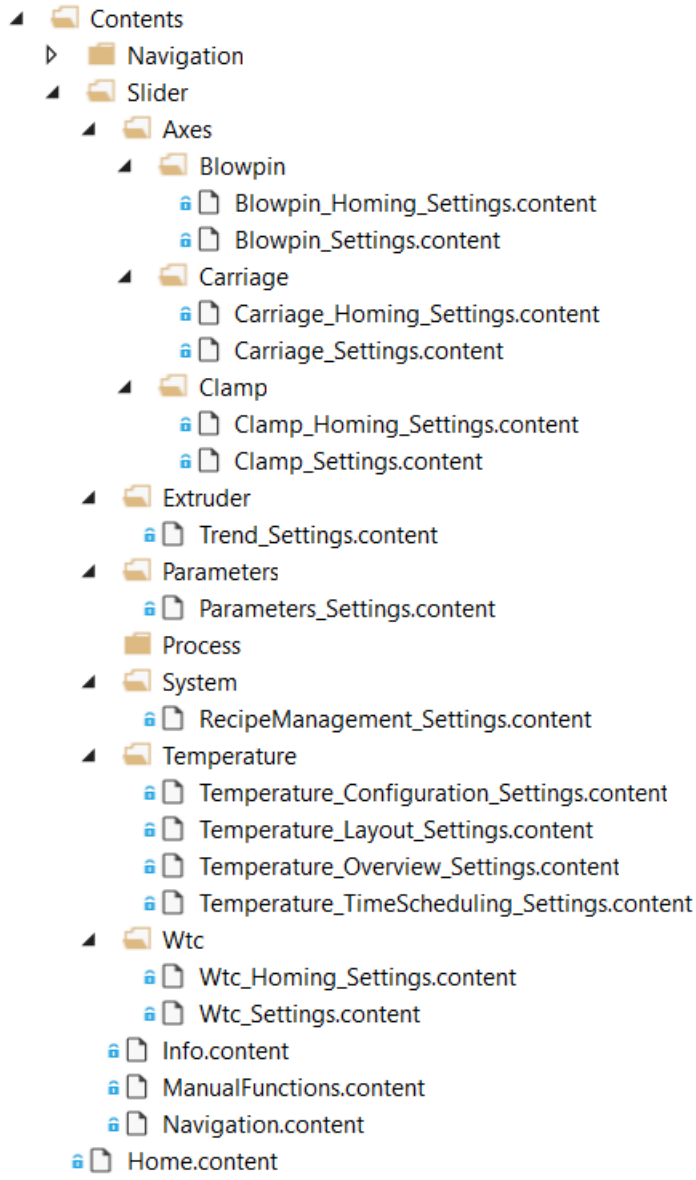
The folder *Contents* contains in further subfolders all existing pages or `.content` files of the project. The contents are built according to certain structures. These are only recommendations.



The folder *Contents/Navigation* contains all Contents categorized in folders that can be found in the navigation of the Plastic Application HMI project.



The folder *Contents/Slider* contains all contents categorized in folders that are applied in the slider area of the Plastic Application HMI project.



The subfolders of the two folders *Contents/Navigation* and *Contents/Slider* have the same structure in order to recognize the affiliation of the contents from the slider area and the contents of the main area (contained in the navigation).

As desired, additional files of the type `.content` can be created or the existing contents can be deleted or edited.

Update capability of the Plastic Application HMI

Editing the Plastic Application HMI can lead to an impairment of the update capability. This may affect the support provided by Beckhoff Automation.

Naming controls

The naming (*Identifier*) of the controls within a content is done according to a special scheme to avoid duplicate naming, as this leads to errors.

The naming in the Plastic Application HMI project follows the following scheme:
ContentName_XYZ_ControlType

1. **ContenName**

   The name, i.e. the ID of the content (the identifier of the control of type `TcHmi.Controls.System.TcHmiContent`).

   - Example: *Temperature_Parameter_Settings*

2. **XYZ**

   Any description of the control is possible. The hierarchical arrangement within the content can be observed.

   ◦ Example: *Group1_Active_On*

3. **ControlType**

   Type of the control. An abbreviation is used.

   ◦ Example: For the control of type `TcHmi.Controls.Beckhoff.TcHmiButton` the abbreviation *Button* would be used.

Result: "Temperature_Parameters_Settings_Group1_Active_On_Button" is the ID of the described button on the content `Temperature_Parameter_Settings.content` with the previously described scheme.

Overview of controls with possible abbreviations

Control type	Abbreviation
TcHmi.Controls.Beckhoff.TcHmiButton	Button
TcHmi.Controls.Beckhoff.TcHmiCheckbox	Checkbox
TcHmi.Controls.Beckhoff.TcHmiCombobox	Combobox
TcHmi.Controls.Beckhoff.TcHmiDatagrid	Datagrid
TcHmi.Controls.Beckhoff.TcHmiEllipse	Ellipse
TcHmi.Controls.Beckhoff.TcHmiEventGrid	EventGrid
TcHmi.Controls.Beckhoff.TcHmiEventLine	EventLine
TcHmi.Controls.Beckhoff.TcHmiLine	Line
TcHmi.Controls.Beckhoff.TcHmiLinearGauge	LinearGauge
TcHmi.Controls.Beckhoff.TcHmiLocalizationSelect	LocalizationSelect
TcHmi.Controls.Beckhoff.TcHmiRadialGauge	RadialGauge
TcHmi.Controls.Beckhoff.TcHmiRadioButton	RadioButton
TcHmi.Controls.Beckhoff.TcHmiRectangle	Rectangle
TcHmi.Controls.Beckhoff.TcHmiStateImage	StateImage
TcHmi.Controls.Beckhoff.TcHmiTextblock	Text block
TcHmi.Controls.Beckhoff.TcHmiTextbox	Text box
TcHmi.Controls.Beckhoff.TcHmiToggleButton	ToggleButton
TcHmi.Controls.Beckhoff.TcHmiToggleSwitch	ToggleSwitch
TcHmi.Controls.Beckhoff.TcHmiTrendLineChart	TrendLineChart
TcHmi.Controls.Beckhoff.TcHmiUserManagement	UserManagement
TcHmi.Controls.FavoriteBarControl.FavoriteBarControl	FavoriteBar / FavoriteBarControl
TcHmi.Controls.Plastic.ArrowMotionGraphControl	ArrowMotionGraph / ArrowMotionGraphControl
TcHmi.Controls.Plastic.BlowPressureChart	BlowPressureChart
TcHmi.Controls.Plastic.CamControl	Cam / CamControl
TcHmi.Controls.Plastic.Configurator	Configurator
TcHmi.Controls.Plastic.CurveEditorControl	CurveEditor / CurveEditorControl
TcHmi.Controls.Plastic.InputBox	InputBox
TcHmi.Controls.Plastic.ManualOperation	ManualOperation
TcHmi.Controls.Plastic.MeasurementUnitSelector	MeasurementUnitSelector
TcHmi.Controls.Plastic.MonitoringControl	Monitoring / MonitoringControl
TcHmi.Controls.Plastic.PfwSingleTempControl	PfwSingleTempControl
TcHmi.Controls.Plastic.PfwTempParameters	PfwTempParameters
TcHmi.Controls.Plastic.ProcessScheduler	ProcessScheduler
TcHmi.Controls.Plastic.StateIndicator	StateIndicator
TcHmi.Controls.Plastic.Table	Table
TcHmi.Controls.Plastic.TimerControl	Timer / TimerControl
TcHmi.Controls.Plastic.ZoneConfiguration	ZoneConfiguration
TcHmi.Controls.Plastic.ZoneGrouping	ZoneGrouping
TcHmi.Controls.Plastic.ZoneImageLayout	ZoneImageLayout
TcHmi.Controls.ResponsiveNavigation.TcHmiBreadcrumb	Breadcrumb
TcHmi.Controls.ResponsiveNavigation.TcHmiNavigationBar	NavigationBar
TcHmi.Controls.ResponsiveNavigation.TcHmiNavigationContent	NavigationContent
TcHmi.Controls.System.TcHmiContainer	Container
TcHmi.Controls.System.TcHmiContent	Content
TcHmi.Controls.System.TcHmiGrid	Grid
TcHmi.Controls.System.TcHmiRegion	Region

Control type	Abbreviation
TcHmi.Controls.System.TcHmiView	View

Standard Content structure

Most contents are built using grids. The grids make it easier to align the other controls that are in that grid to allow for the tile design that is consistent throughout the project. The contents or controls added to the grid are also arranged in smaller grids to make it easier to align the controls contained within a tile. The grid, which is to represent a tile, can be assigned the CSS class from the `Beckhoff.TwinCAT.HMI.Plastic.Themes` NuGet package with the name `Tiles` via the `ClassNames` property of the `Common` category to color the background color white, round the corners and add a slight shadow. Using the CSS class `TilesHeadline` from the same NuGet package, you can quickly customize the appearance of the text block for the tile heading. The class colors the background of the text block in an appropriate shade of gray, the text color becomes white and the corners are adjusted according to the tile. A height of 45 px is usually used for the heading text block.

Use of grids

When using grids in grids, pay attention to the Z-index of the controls. If a control (e.g. a TcHmiGrid) is located within a TcHmiGrid control and its size exceeds the specified range of the corresponding location in the grid, this may mean that some contained controls (such as TcHmiButton controls) can no longer be operated. Thus, the `Zindex` property of the `Layout` category of the control that is inside a grid should be given a higher value. For the outer grid, for example, the `Zindex` property can be set to 0.

Special case: Creation of multiple pages in one content

The content intended for the slider area can be larger than the actual available space. Thus, for example, the width of the content can be twice as wide as the actual available space of the TcHmiRegion control if a second page is to be created. By changing the `Left` properties from the `Layout` category, the included grid that fills the entire content will be shifted accordingly to display the specific content that is desired. This can be done, as in the Plastic Application HMI project, using controls of type `TcHmiRadioButton`.

Example: Manual functions [▶ 152]

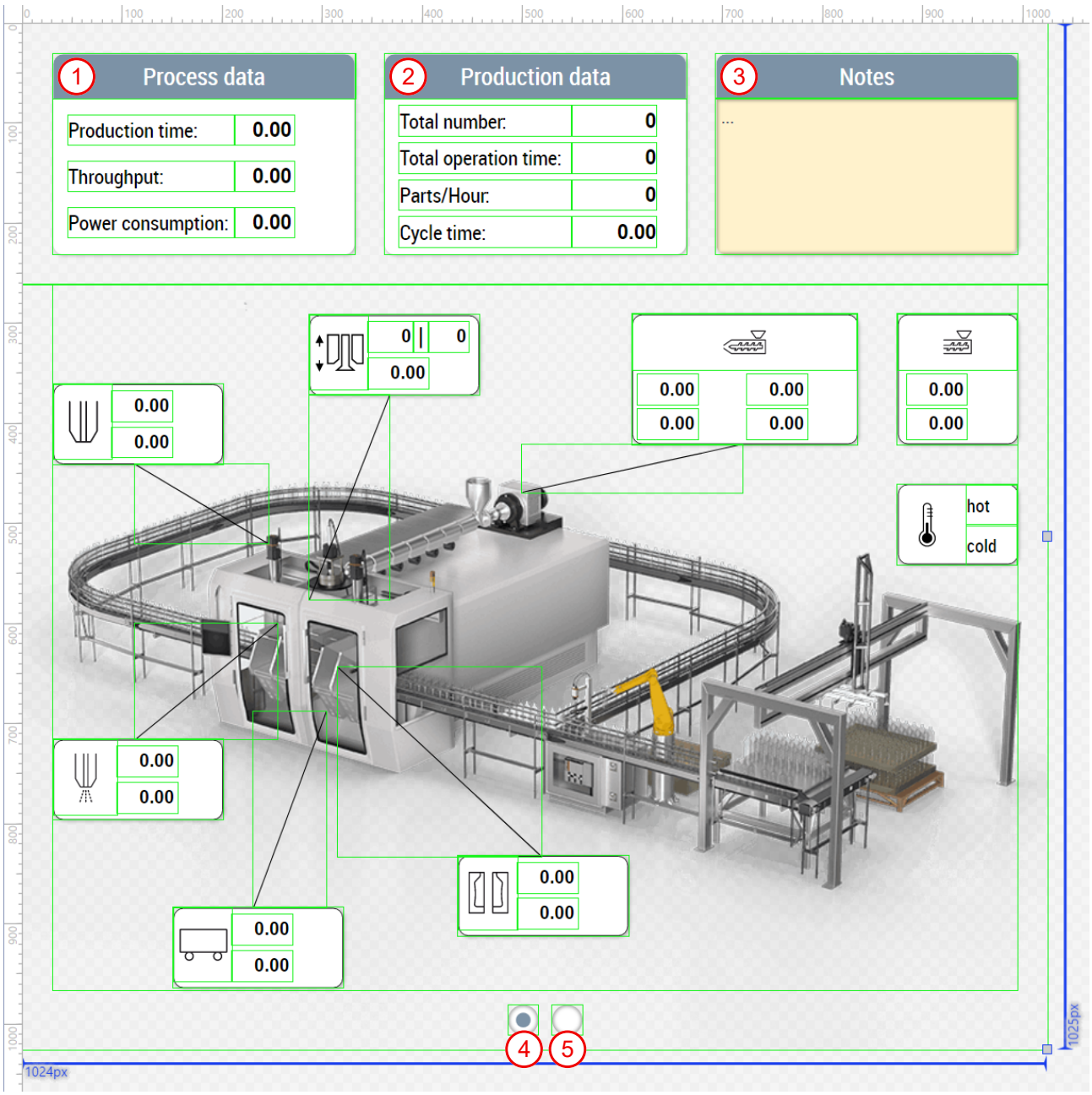
Available from version 12.6.0

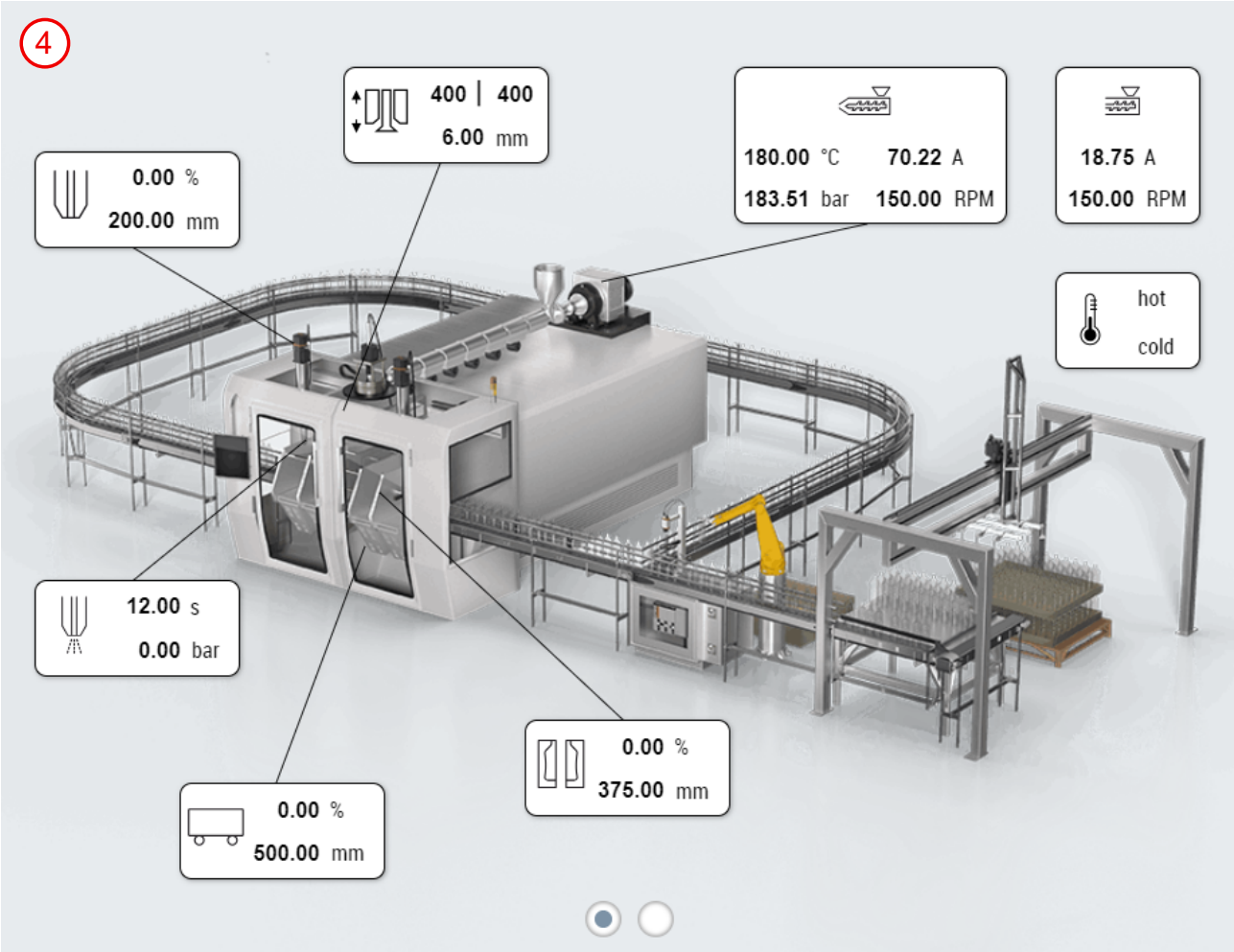
7.3.1 Start page

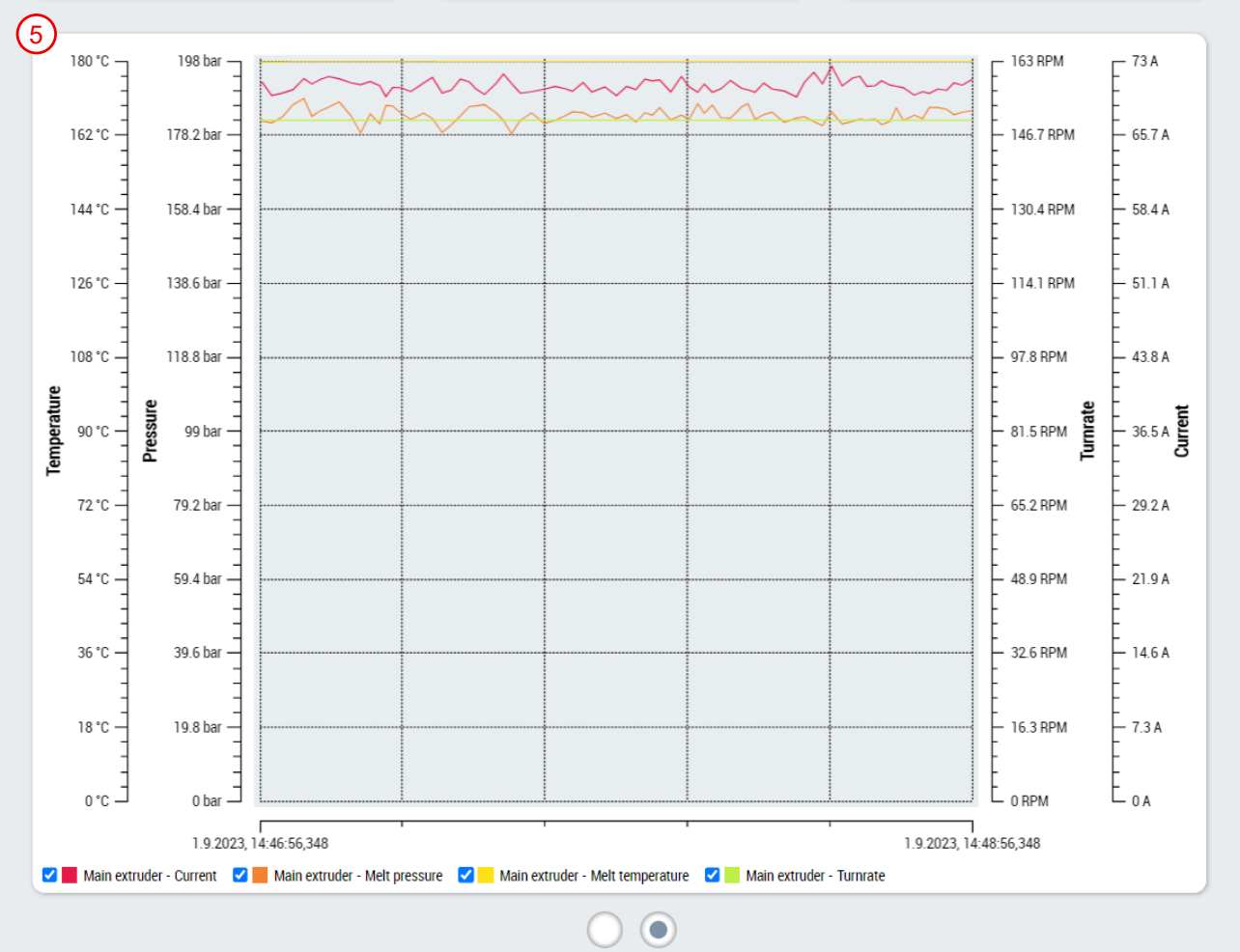
Home.content

The start page of the user interface consists of a collection of important data and information. The page contains the following components:

1. Display of the most important process data
2. Display of the most important production data
3. Field for entering notes
4. Switching using the radio button to display the machine graphic with the most important actual values
5. Switching using the radio button to display the trend values of the main extruder


1
Process data
Production time:
0.00
Throughput:
0.00
Power consumption:
0.00
2
Production data
Total number:
0
Total operation time:
0
Parts/Hour:
0
Cycle time:
0.00
3
Notes
...
0
0
0.00
0.00
0.00
0.00
0.00
0.00
0.00
0.00
0.00
0.00
0.00
0.00
hot
cold
0.00
0.00
0.00
0.00
4
5
1024px
1025px


4
400 | 400
6.00 mm
0.00 %
200.00 mm
180.00 °C
70.22 A
183.51 bar
150.00 RPM
18.75 A
150.00 RPM
hot
cold
12.00 s
0.00 bar
0.00 %
375.00 mm
0.00 %
500.00 mm



Available from version 12.6.0

7.3.2 Navigation

Navigation.content

The navigation page is called in the slider area via the 1st tab. It consists of controls from the NuGet package `Beckhoff.TwinCAT.HMI.ResponsiveNavigation`. The TcHmiNavigationBar control is used to define the pages that appear in the navigation and to display the top categories. In addition, the following controls must be set:

- **TcHmiNavigationContent:** Display the available pages in the navigation depending on the selected category in the TcHmiNavigationBar control (also located on the Navigation.content)
- **TcHmiRegion:** Area for displaying the selected content (main content of View [▶ 190])
- **TcHmiBreadcrumb:** Display of the breadcrumb (located directly on the View [▶ 190])

The `MenuData` property of the `Common` category of the TcHmiNavigationBar control is used to determine the navigation content. This property can be connected to the `NavBarMenuData` property of the Configurator control. This means that all settings related to navigation can be made in the `NavigationConfig` property of the Configurator control and do not have to be set multiple times.

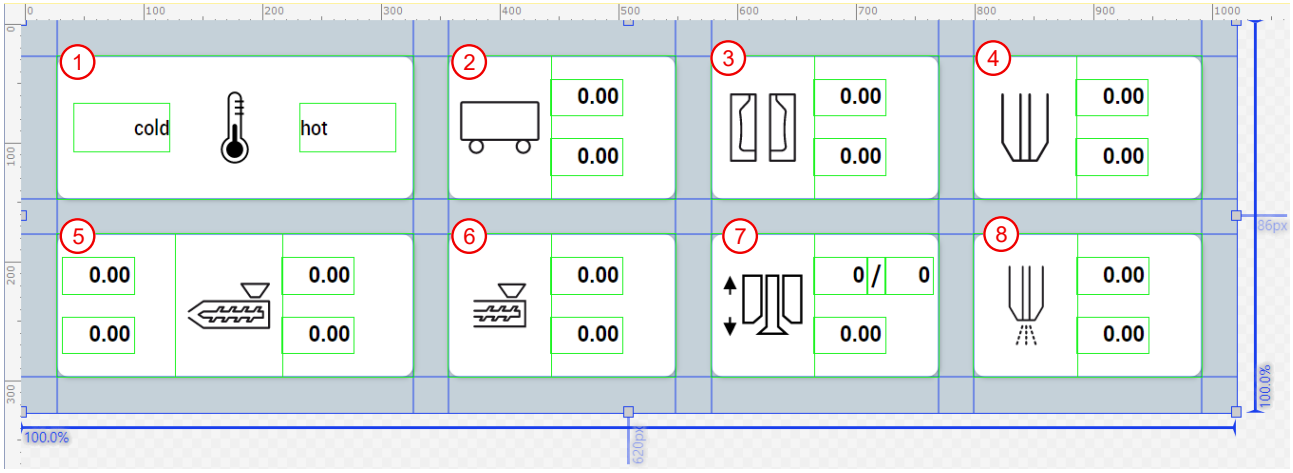
Available from version 12.6.0

7.3.3 Info

Info.content

The info page contains the most important machine data. This page is displayed in the slider area with the info tab (2nd tab). The page must be adapted to the specific machine. The variables are assigned to the corresponding machine components with the help of icons. The page contains the following components:

1. One of the temperatures is outside of the set tolerance
2. Torque and position of the carriage
3. Torque and position of the clamping unit
4. Torque and position of the blow pin
5. Melt temperature, melt pressure, current and turnrate of the main extruder
6. Co-extruder current and turnrate
7. Index and position of the WTC
8. Blowing time and blowing pressure of the blowing process



Available from version 12.5.1

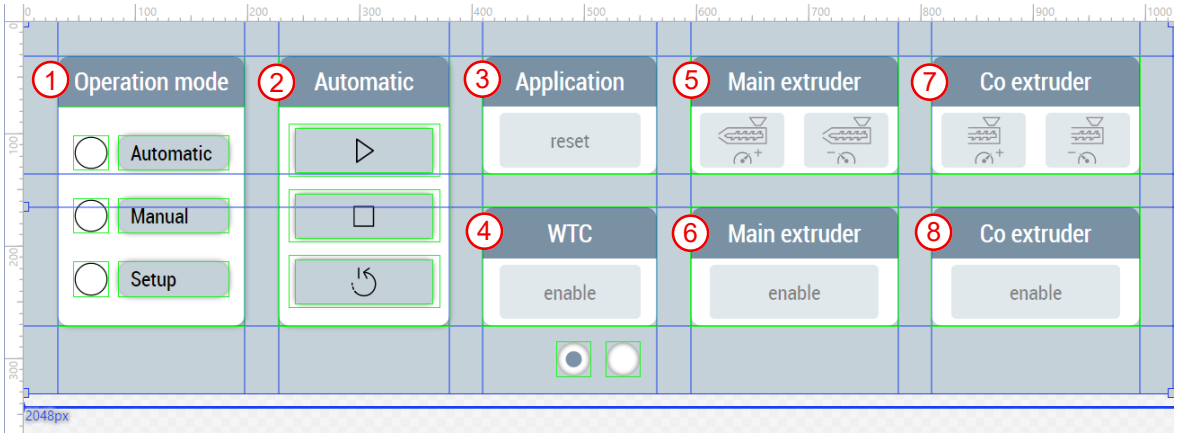
7.3.4 Manual functions

ManualFunctions.content

The manual function page contains all manual functions and must be adapted to the specific machine. This page is displayed in the slider area with the Manual tab (3rd tab). With the help of the ManualOperation controls and the instances of the type FB_ManualFunctionHmi [▶ 45], controls for manual functions can easily be created. The content is divided into two pages.

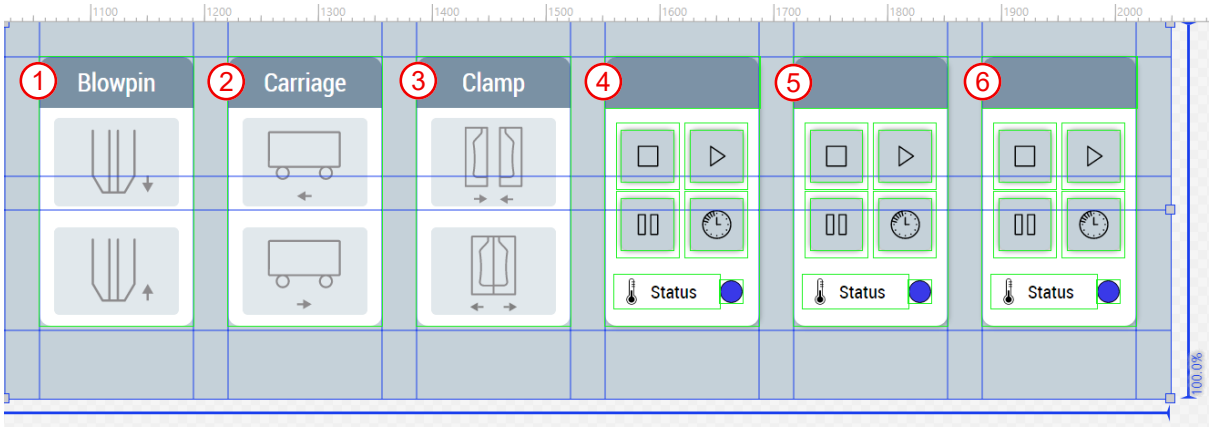
The first page contains the following components:

1. Change of the operation mode
2. Switching the automatic mode on and off and moving the axes to the base position
3. Resetting the entire application
4. Enabling the WTC
5. Increasing and decreasing the turnrate of the main extruder
6. Switching the main extruder on
7. Increasing and decreasing the turnrate of the co-extruder
8. Switching the co-extruder on



The second page contains the following components:

1. Movement of the blow pin
2. Movement of the carriage
3. Movement of the clamping unit
4. Select temperature mode of group 1
5. Select temperature mode of group 2
6. Select temperature mode of group 3



Available from version 12.6.0

7.3.5 Axes

The folders named *Axes* (*Contents/Navigation/Axes* and *Contents/Slider/Axes*) contain all contents needed for the axes of the machine. This includes a subfolder for each of the three axes included in the project (blow pin [▶ 153], carriage [▶ 157] and clamping unit [▶ 157]) as well as a common oscilloscope page [▶ 158] for all axes.

7.3.5.1 Blow pin

The folders named *Blowpin* (*Contents/Navigation/Axes/Blowpin* and *Contents/Slider/Axes/Blowpin*) contain all contents needed for the blow pin.

Content	Description
Blowpin.content [▶ 153]	The main page of the blow pin.
Blowpin_Settings.content [▶ 153]	Displayed in the slider area under the 4th tab when `Blowpin.content` is in the main area.
Blowpin_Homing.content [▶ 155]	The homing page of the blow pin.
Blowpin_Homing_Settings.content [▶ 155]	Displayed in the slider area under the 4th tab when `Blowpin_Homing.content` is in the main area.

7.3.5.1.1 Movement

Blowpin.content

The main page of the blow pin consists on the one hand of the use of the ArrowMotionGraph control and is used to set and display the blow pin movement in segments. The control allows the adjustment of velocity and position, acceleration and deceleration, as well as in both directions of movement with up to 5 segments each.

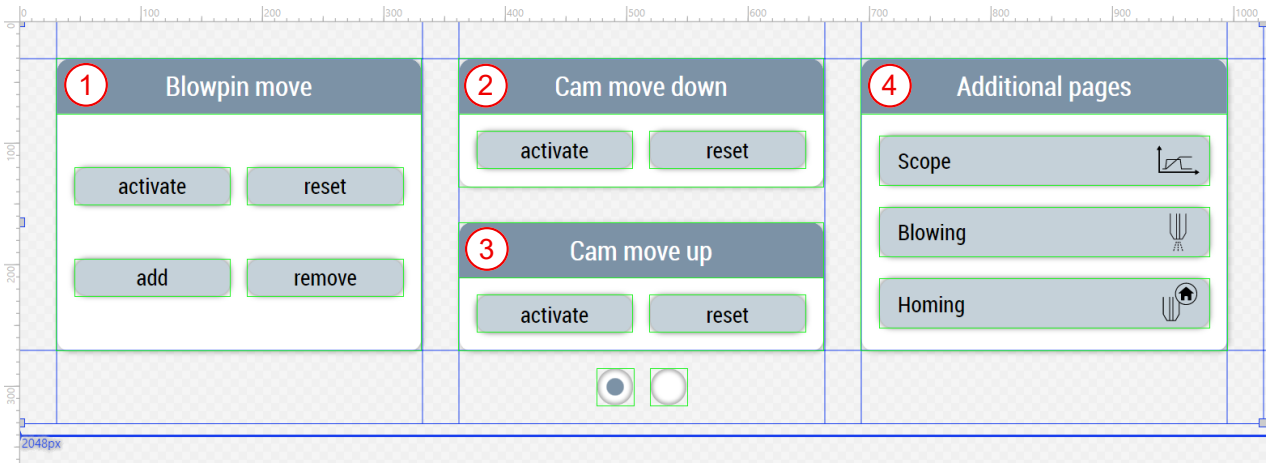
On the other hand, the blow pin main page contains the cam control twice, once for each direction of movement of the axis. The two controls are used to set and display the cams by entering the length and position per segment. Up to 5 segments can be added per control.

Blowpin_Settings.content

In addition to the main page of the blow pin there is a supplementary page for the slider area. The content is divided into two pages.

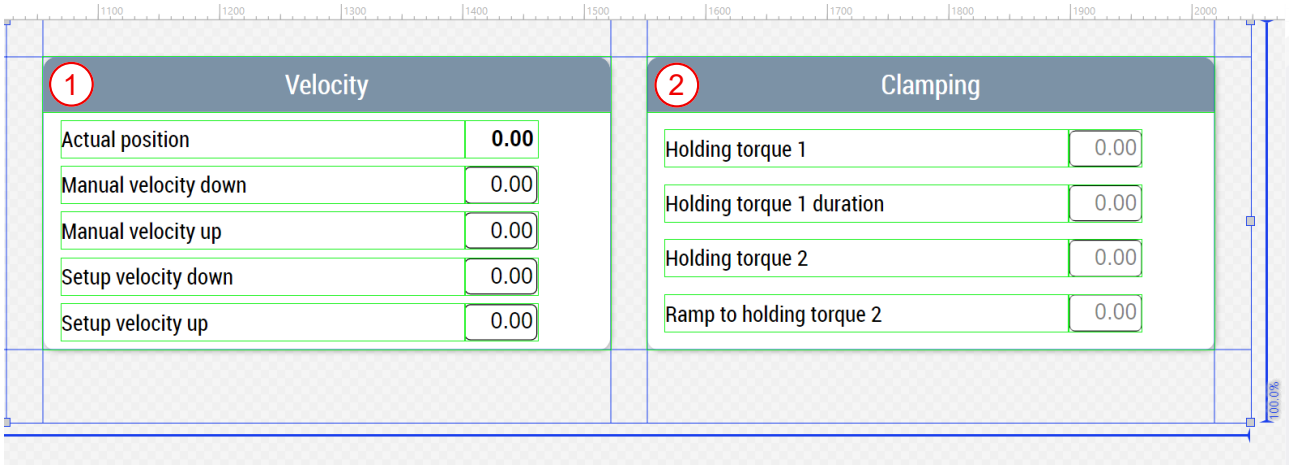
The first page contains the following components:

1. Activating and resetting the settings and add and remove segments of the blow pin movement
2. Activating and resetting the settings of the cams in the downward movement of the blow pin
3. Activating and resetting the settings of the cams in the upward movement of the blow pin
4. Area for navigation to other related pages



The second page contains the following components:

1. Setting velocity values
2. Setting torques for clamping



Available from version 12.6.0

7.3.5.1.2 Homing

Blowpin_Homing.content

The homing page for the blow pin consists of the following components:

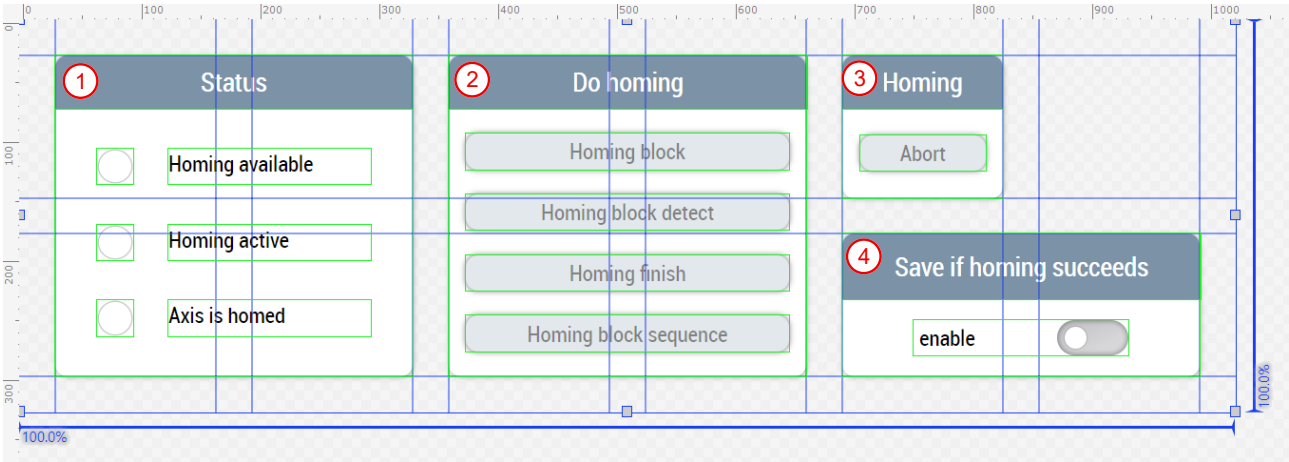
1. Homing block
2. Homing block detection
3. Homing finish
4. Encoder zero offset shift

Using the `UpdateAxesHomingUnits` function from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package, certain units of the values are converted to angle unit groups if it is a transformed axis.

Blowpin_Homing_Settings.content

Supplementary to the homing page of the blow pin, there is page for the slider area. This page contains the following components:

1. Homing status display
2. Homing functions
3. Aborting the homing
4. Activation of the storage option



Available from version 12.6.0

7.3.5.2 Carriage

The folders named *Carriage* (*Contents/Navigation/Axes/Carriage* and *Contents/Slider/Axes/Carriage*) contain all contents needed for the carriage.

Content	Description
Carriage.content	The main page of the carriage.
Carriage_Settings.content	Displayed in the slider area under the 4th tab when `Carriage.content` is in the main area.
Carriage_Homing.content	The homing page of the carriage.
Carriage_Homing_Settings.content	Displayed in the slider area under the 4th tab when `Carriage_Homing.content` is in the main area.

The contents have almost the same structure as the contents of the Blow pin [▶ 153] axis.

7.3.5.3 Clamping unit

The folders named *Clamp* (*Contents/Navigation/Axes/Clamp* and *Contents/Slider/Axes/Clamp*) contain all the contents required for the clamping unit.

Content	Description
Clamp.content	The main page of the clamping unit.
Clamp_Settings.content	Displayed in the slider area under the 4th tab when `Clamp.content` is in the main area.
Clamp_Homing.content	The homing page of the clamping unit.
Clamp_Homing_Settings.content	Displayed in the slider area under the 4th tab when `Clamp_Homing.content` is in the main area.

The contents have almost the same structure as the contents of the blow pin [▶ 153] axis.

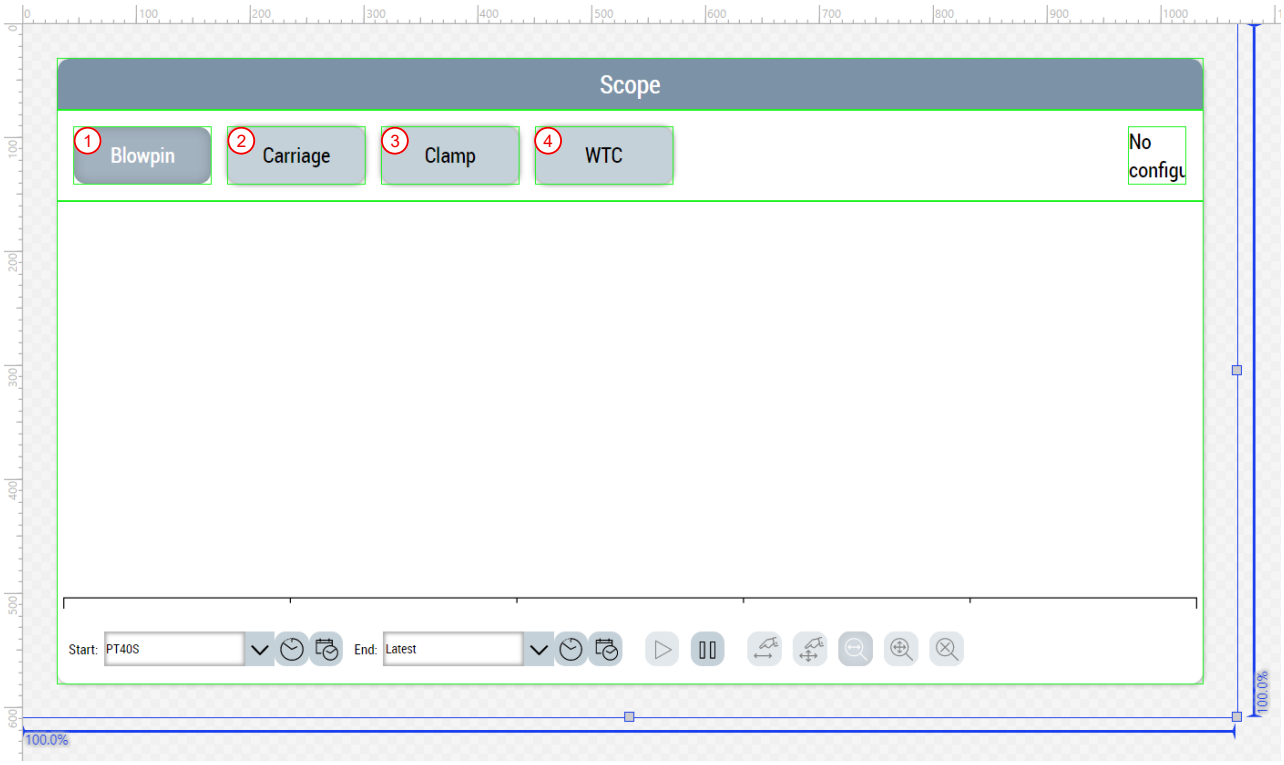
7.3.5.4 Oscilloscope

Scope.content

The oscilloscope page is used to display the following historized symbol values per axis:

- Actual value of the position
- Actual value of the velocity
- Actual value of the torque

The `UpdateScope` function is triggered from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package with the TcHmiToggleButton controls. The function ensures that the TcHmiTrendLineChart control adapts to the selected axis: blow pin (1), carriage (2), clamp (3), WTC (4). In addition, an icon for the respective axis can be displayed via the TcHmiStateImage control.



Available from version 12.6.0

7.3.6 Extruder

The folders named *Extruder* (*Contents/Navigation/Extruder* and *Contents/Slider/Extruder*) contain all the contents needed for the extruders.

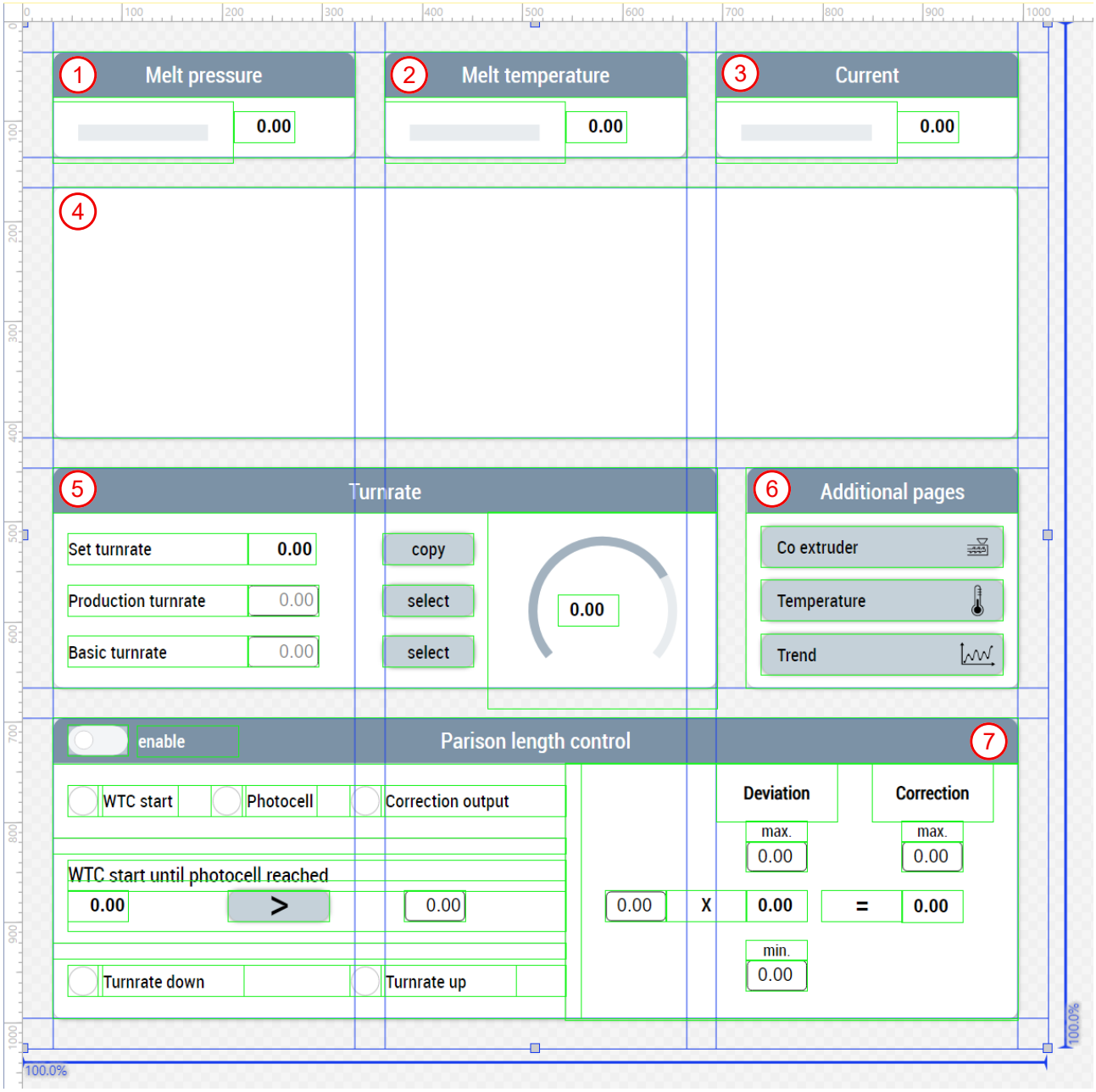
Content	Description
MainExtruder.content [▶ 159]	The page for the main extruder.
CoExtruder.content [▶ 160]	The page for the co-extruder.
Trend.content [▶ 161]	The trend page for extruders and temperature zones.
Trend_Settings.content [▶ 161]	Displayed in the slider area under the 4th tab when `Trend.content` is in the main area.

7.3.6.1 Main extruder

MainExtruder.content

The main extruder page contains the following components:

1. Display of the melt pressure as a value and in the graph
2. Display of the melt temperature as a value and in the graph
3. Display of the current as a value and in the graph
4. Illustration of the main extruder with the associated temperature zones for displaying the actual values and setting option for the setpoints (is configured via the Layout [▶ 183] page of the temperatures)
5. Area for displaying and setting the turnrate
   - Display of the set turnrate
   - Button for copying the currently set turnrate to the production turnrate
   - Display and setting of the production turnrate
   - Button for selecting the production turnrate
   - Display and setting of the basic turnrate
   - Button for selecting the basic turnrate
   - Display of actual value as numerical value and in graph with reference to basic and production turnrate
6. Area for navigation to other related pages
7. Area for setting the parison length control



Available from version 12.6.0

7.3.6.2 Co-extruder

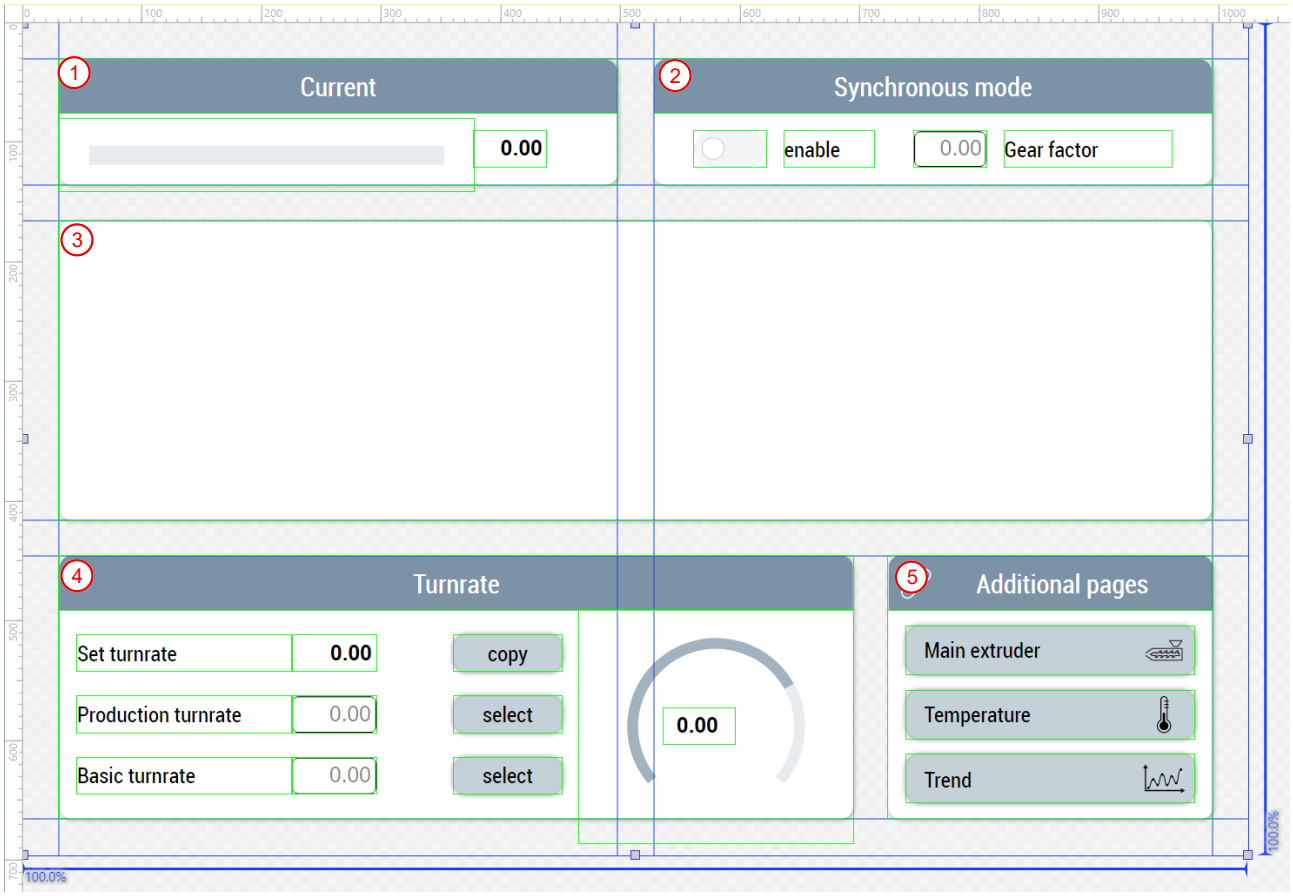
CoExtruder.content

The co-extruder page contains the following components:

1. Display of the current as a value and in the graph
2. Area for switching on the synchronous mode and setting the gear factor
3. Illustration of the co-extruder with the associated temperature zones for displaying the actual values and setting option for the setpoints (is configured via the Layout [▶ 183] page of the temperatures)
4. Area for displaying and setting the turnrate
   - Display of the set turnrate
   - Button for copying the currently set turnrate to the production turnrate

- Display and setting of the production turnrate
- Button for selecting the production turnrate
- Display and setting of the basic turnrate
- Button for selecting the basic turnrate
- Display of actual value as numerical value and in graph with reference to basic and production turnrate

5. Area for navigation to other related pages

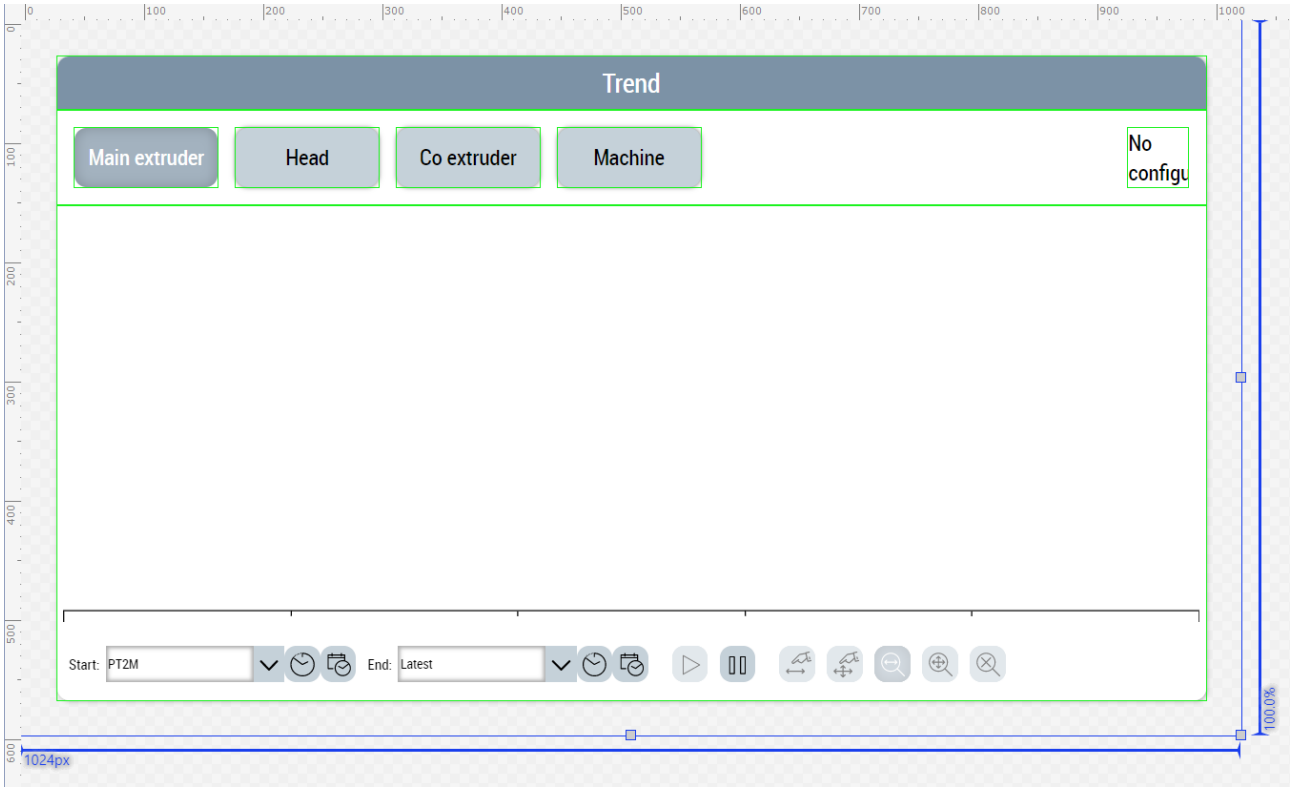


Available from version 12.6.0

7.3.6.3 Trend

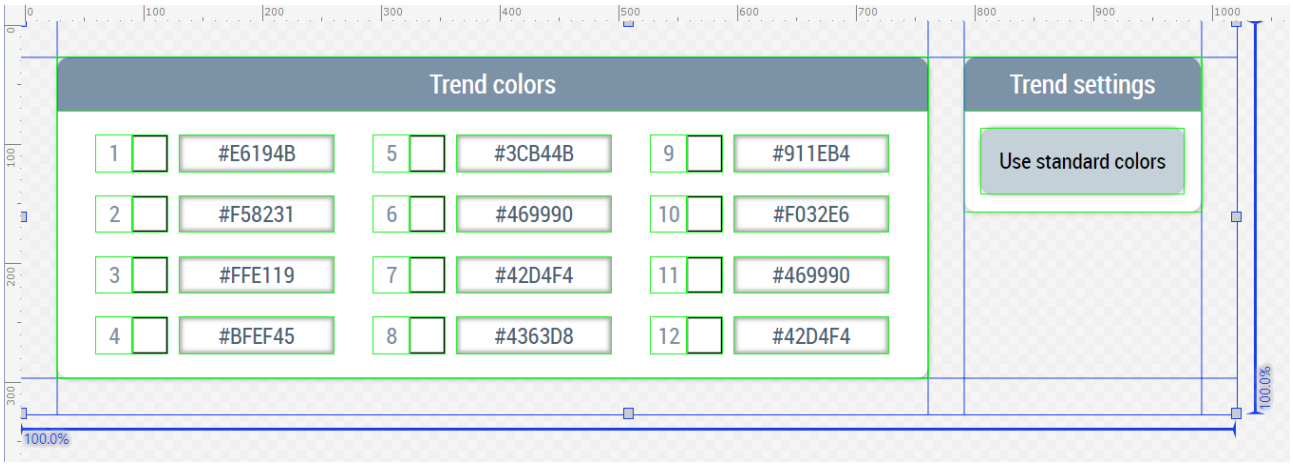
Trend.content

The trend page is used to display curves of extruder and temperature values in a trend graph (Historized Symbols). Using the `UpdateTrend` function from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package and the FB_TrendHmi [▶ 105], the TcHmiTrendLineChart control is adjusted depending on the TcHmiToggleButton control clicked. In addition, an icon for the selection can be displayed via the TcHmiStateImage control.



Trend_Settings.content

This content can be used to adjust the colors of the curves. The hexadecimal color code is entered via TcHmiInput controls. The TcHmiButton control triggers the function `ResetTrendAxisColors` from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package to be able to overwrite the changes with standard colors again.



Available from version 12.5.1

7.3.7 Parameter

The folders named *Parameters* (*Contents/Navigation/Parameters* and *Contents/Slider/Parameters*) contain all contents needed for the configuration of the machine data.

Content	Description
Parameters_Blowpin.content	The configuration page of the machine data for the blow pin.
Parameters_Carriage.content	The configuration page of the machine data for the carriage.
Parameters_Clamp.content	The configuration page of the machine data for the clamping unit.
Parameters_CoExtruder.content	The configuration page of the machine data for the co-extruder.
Parameters_MainExtruder.content	The configuration page of the machine data for the main extruder.
Parameters_Monitoring.content	The configuration page of the machine data for the monitoring values.
Parameters_Setpoints.content	The configuration page of the machine data for the setpoints.
Parameters_Temperature.content	The configuration page of the machine data for the temperature zones and the temperature supply.
Parameters_Wtc.content	The configuration page of the machine data for the WTC.
Parameters_Settings.content	Displayed in the slider area under the 4th tab when one of the configuration pages is in the main area.

A configuration page is built using the Table control. The control offers the possibility to create various variables in different data types with units and to distribute them to different subpages. With the help of the function `FillParamTable` from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package, the table control can be automatically filled with the appropriate values based on the parameters passed. Changes in the PLC are automatically applied to the table. Additional rows can be added by adjusting the `FirstTable` attribute of the table control.

Parameters_Blowpin.content

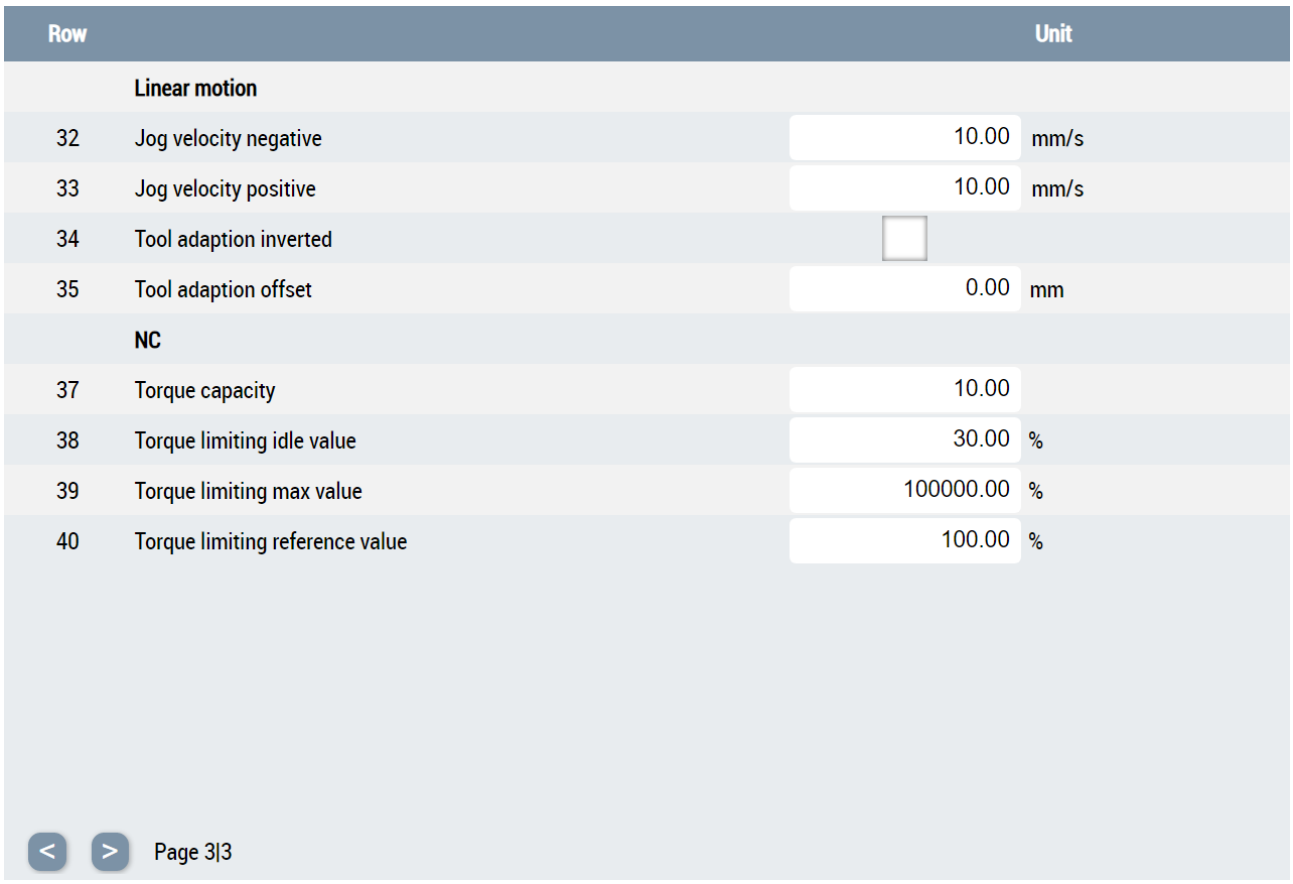
The configuration page of the machine data for the blow pin consists of three sub-pages. The variables are categorized by subheadings. The `FillParamTable` function automatically fills and updates the table when there are changes in the PLC.

Row			Unit
	Axis		
2	Drive reversed	✓	
3	Encoder interpolation	1048576.00	
4	Encoder reversed	✓	
5	Encoder weighting	5.00	
6	Encoder zero shift	0.00	
7	Lag Kp	1.00	
8	Lag filter	0.02	
9	Lag limit	5.00	mm
10	Lag monitored	✓	
11	Max acceleration	276.00	mm/s²
12	Max deceleration	276.00	mm/s²
13	Max jerk	830.00	mm/s³
14	Max application velocity	184.00	mm/s
15	Velo Max System	203.00	mm/s

< > Page 1|3

Row			Unit
16	Software end max	200.00	mm
17	Enable software end max	✓	
18	Software end min	0.00	mm
19	Enable software end min	✓	
	Blowpin		
21	Clamping distance	0.50	mm
22	Max settable clamping duration	30.00	s
23	Max settable clamping ramp	5.00	s
24	Clamping velocity	50.00	mm/s
	Homing		
26	Detection velocity max settable time	10.00	s
27	Max settable distance	800.00	mm
28	Max settable Position	500.00	mm
29	Min settable Position	0.00	mm
30	Max settable time limit	30.00	s

< > Page 2|3



Parameters_Temperature.content

The machine data configuration page for temperature consists of two table controls that are toggled using the TcHmiToggleSwitch control:

1. Parameterization of the temperature channels
2. Parameterization of the temperature supply

Using the functions `FillParamTable` and `UpdateTemperatureParametersTableNumberEx` from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package, the tables are automatically filled and the number of temperature zones is determined from the PLC in order to adjust the table control accordingly.

Row	Last tuning: 2023-09-01T12:38:11Z	(1) Channels / Temperature supply	Unit
	Temperature channel		
2	Absolute high	300.00	%
3	Absolute low	100.00	%
4	Actual current	**0.00**	A
5	Actual temperature gain	1.00	
6	Actual temperature offset	0.00	%
7	Actual temperature	**18.00**	%
8	Cold junction compensation mode	Disabled	
9	Cold junction compensation zone	0	
10	Cooler swap index	0	
11	Disable auto step	☐	
12	Disable terminal communication	☐	
13	Enable	☐	
14	Enable error heating	☐	
15	Error current tolerance	0.00	

< > Page 1|6 Zone 1|20 < >

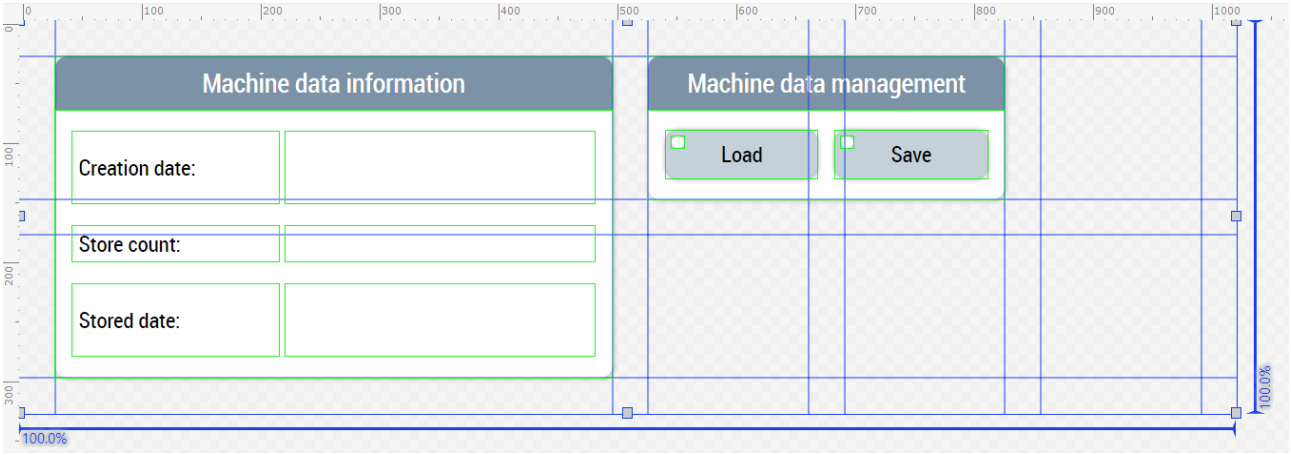
Row	Phase 1	Channels / Temperature supply (2)	Unit
	Temperature supply		
2	PWM cycle time	1.00	
3	PWM factor cooling	10	
4	PWM max on C	0.00	
5	PWM max on time	1.00	s
6	PWM max ramp load	1.00	
7	PWM min on time	0.10	s
8	Use supply PWM parameters	☐	

Temperature supply 1|4 < >

Parameters_Settings.content

The content specially made for the slider area serves as a supplement. It can be selected via the 4th tab of the slider area when the main area displays one of the configuration pages. Included is information about the saved file and functions for loading and saving this data. The displayed texts and functions of the buttons are

automatically adapted depending on the displayed configuration page in the main area of the user interface using the `UpdateParametersSettingsControls` function from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package.



Available from version 12.6.0

7.3.8 Process

The folders named *Process* (*Contents/Navigation/Process* and *Contents/Slider/Process*) contain all contents needed for the process of the machines.

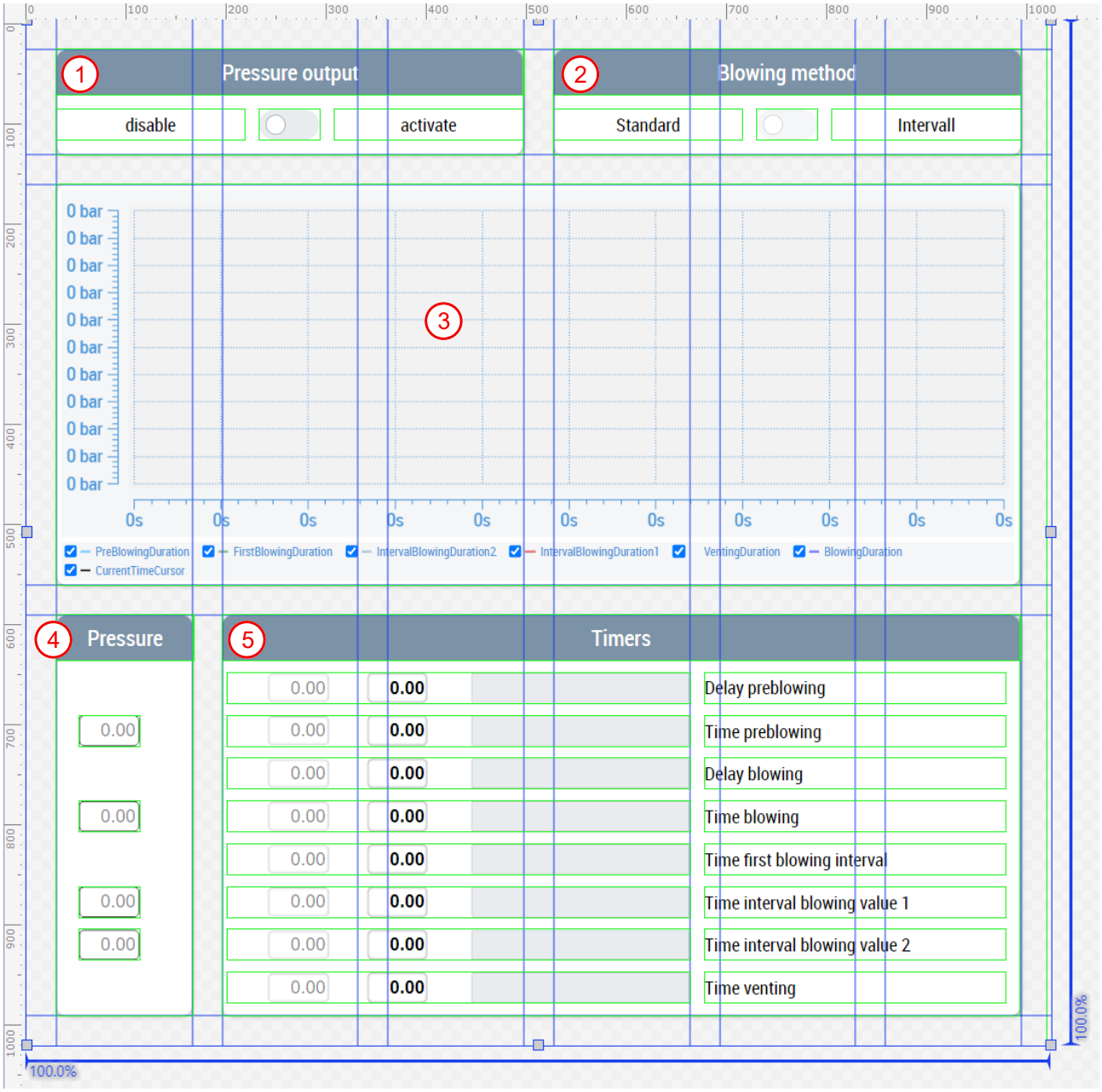
Content	Description
Blowing.content [▶ 167]	The page for the blowing process.
Monitoring.content [▶ 168]	The page for monitoring values.
Setpoints.content [▶ 170]	The page for setpoints.
Timer.content [▶ 171]	The page for time values.

7.3.8.1 Blowing

Blowing.content

The blowing page consists of the following components:

1. Enable or disable the pressure output
2. Choice of blowing method between standard and interval blowing
3. Blow curve mapped in BlowPressureChart control
4. Pressure settings at the specific times
5. Setting delays and times using timer controls



Available from version 12.5.1

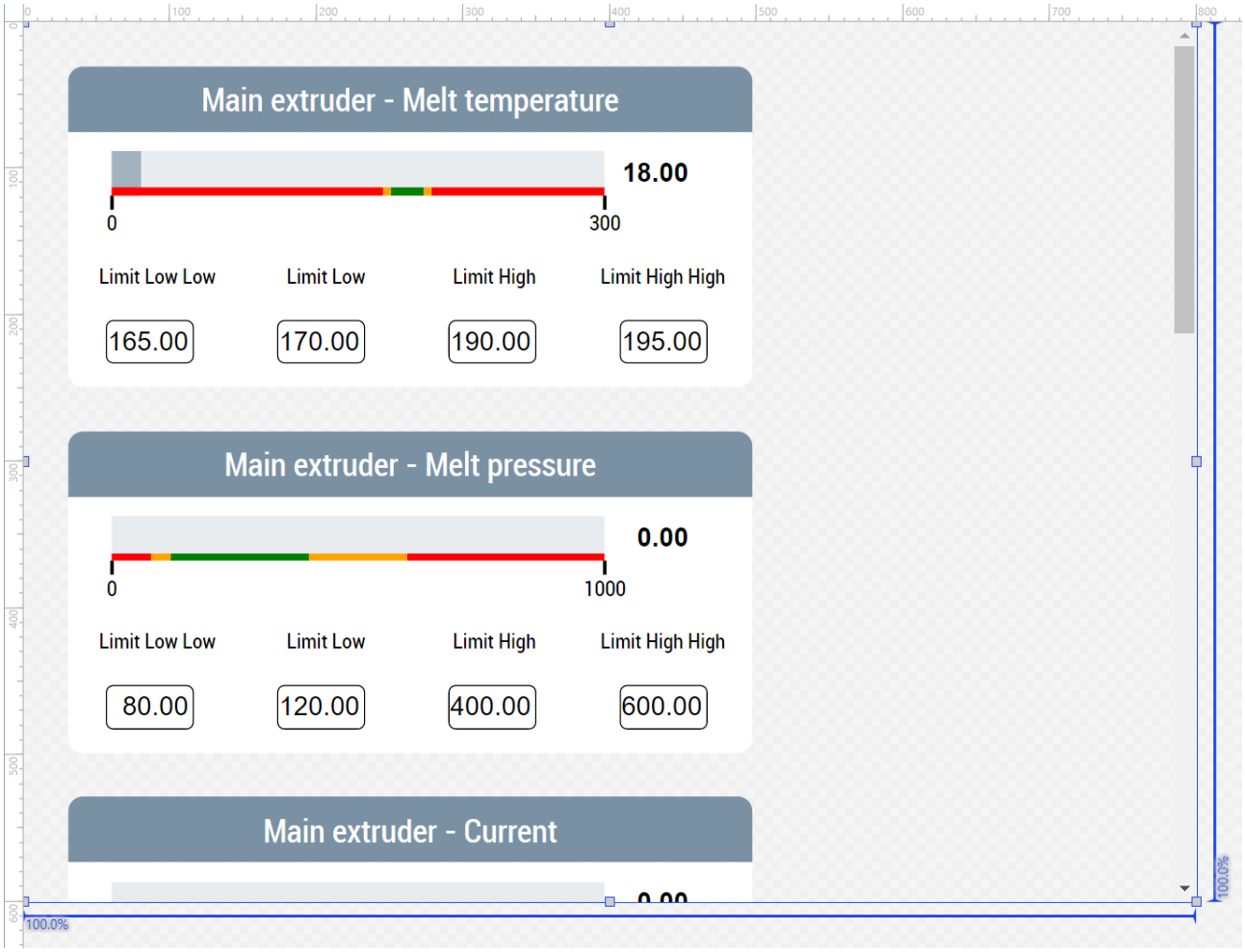
7.3.8.2 Monitoring

Monitoring.content

The monitoring page is built using the Monitoring control from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package and instances of type FB_MonitoringHmi. It contains one area per value for displaying the current value and for setting the threshold values. In addition, the set and measured values are illustrated in a graph. The values currently included are the following:

1. Melt temperature of the main extruder
2. Melt pressure of the main extruder
3. Current of the main extruder
4. Current of the co-extruder

5. Control cabinet temperature
6. CPU temperature
7. Hydraulic system pressure

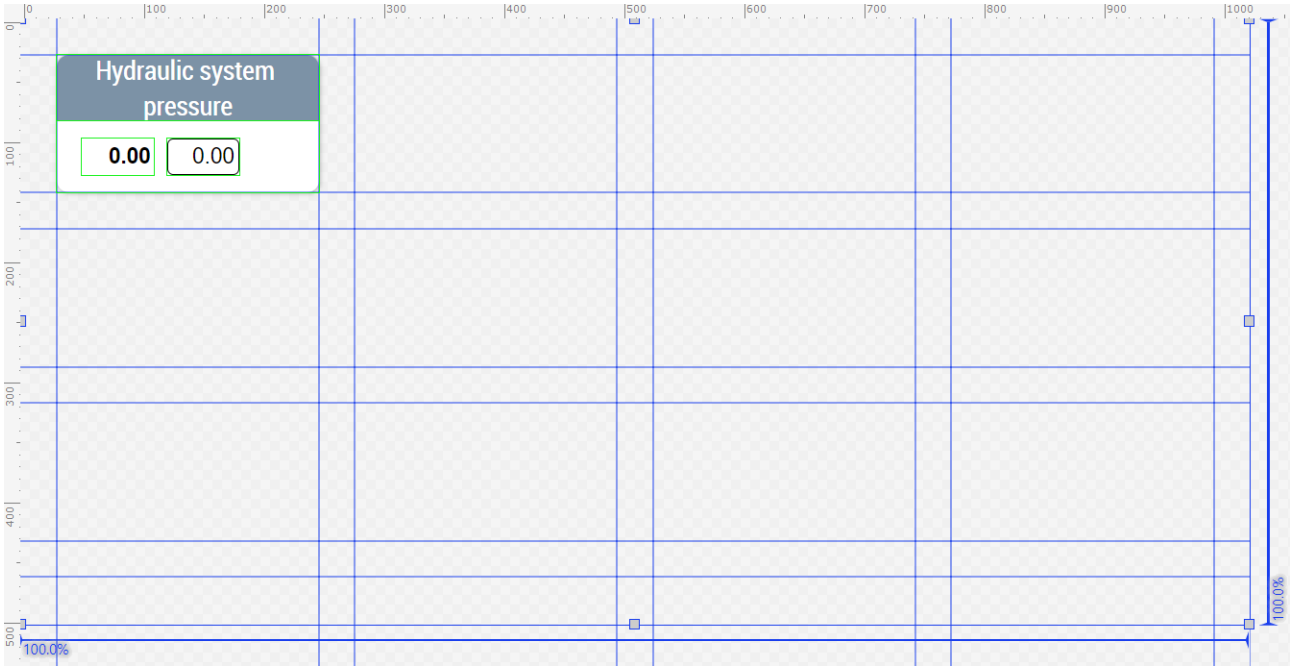


Available from version 12.6.0

7.3.8.3 Setpoints

Setpoints.content

The setpoint page offers the possibility of setting setpoints. Currently, the setpoint of the hydraulic system pressure is included.

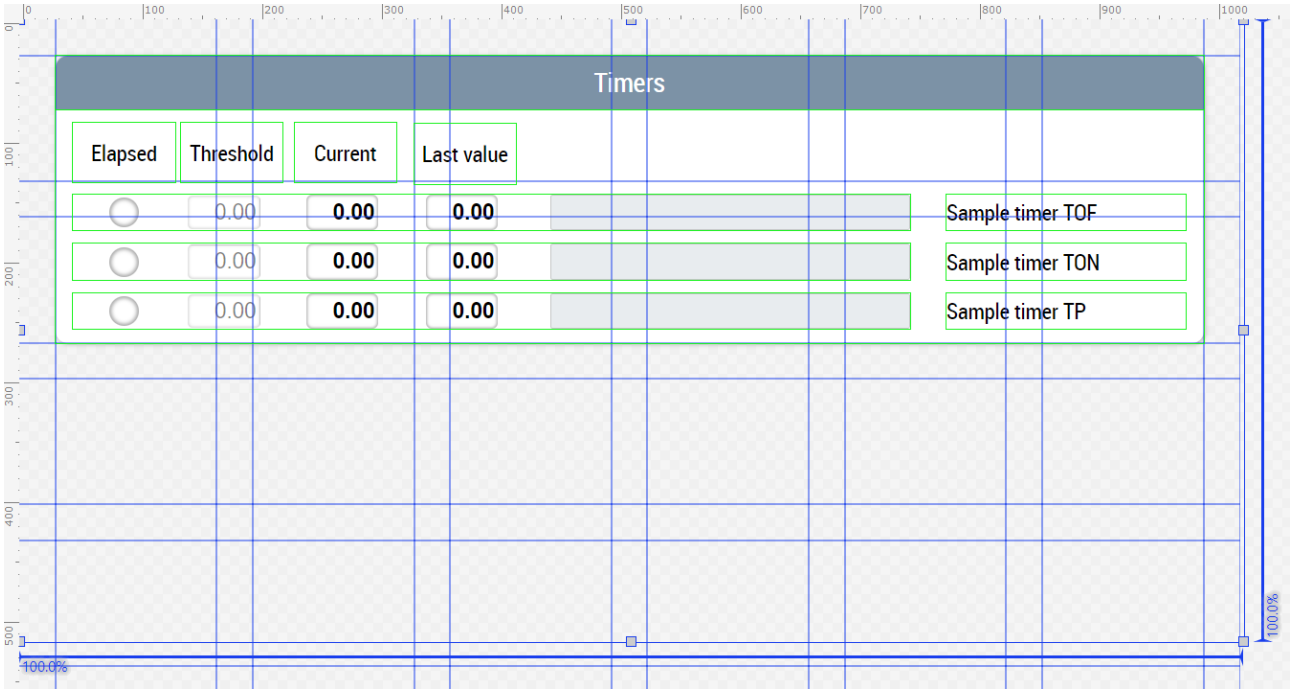


Available from version 12.5.1

7.3.8.4 Times

Timer.content

The timer page is used for setting and monitoring times. Timer controls and the FB_TimerHmi [▶ 84] can be used to create controls for times.



Available from version 12.5.1

7.3.9 System

The folders named *System* (*Contents/Navigation/System* and *Contents/Slider/System*) contain all system relevant contents of the machine.

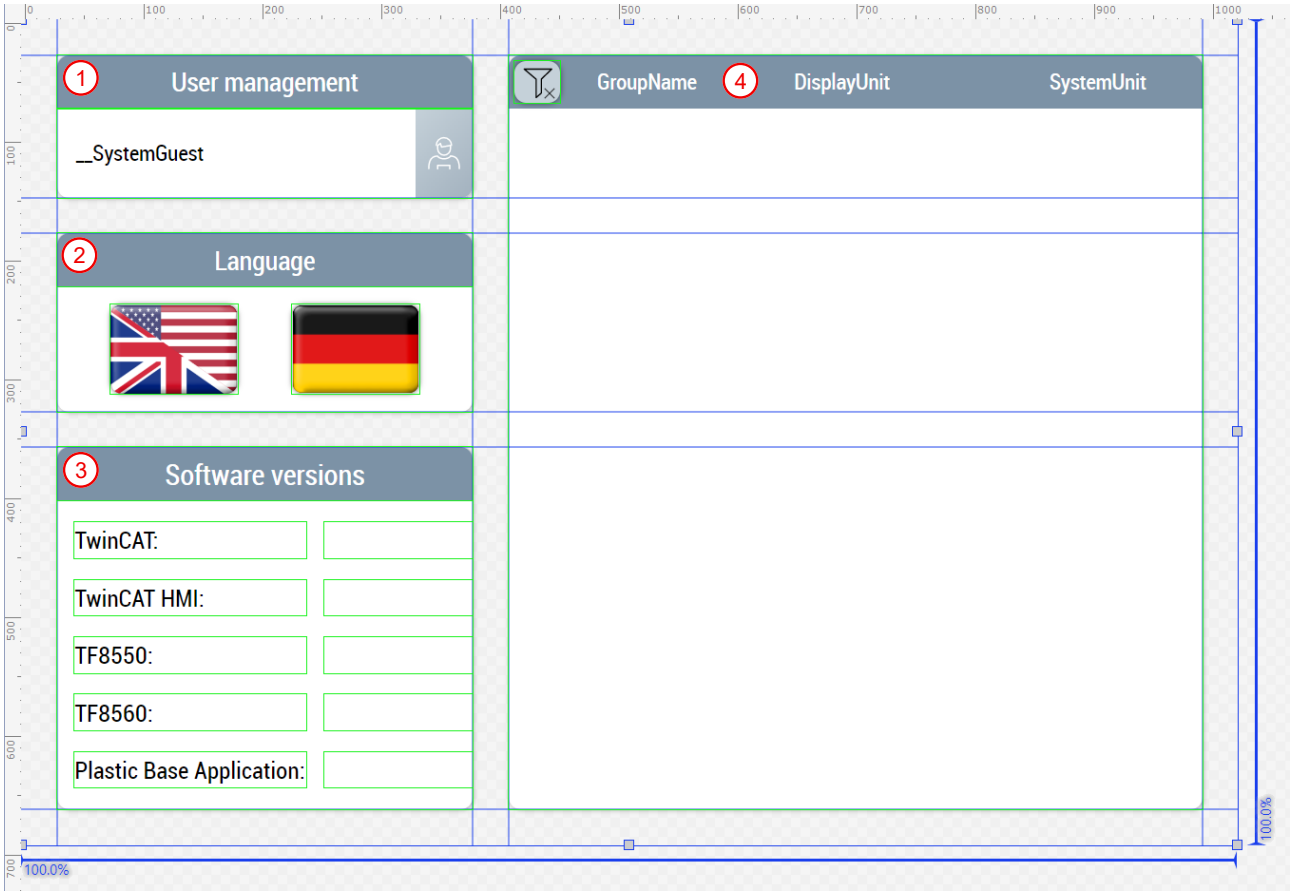
Content	Description
Administration.content [▶ 172]	The page for the administration.
Alarms.content [▶ 173]	The page for events and alarms.
RecipeManagement.content [▶ 173]	The recipe management page.
RecipeManagement_Settings.content [▶ 173]	Displayed in the slider area under the 4th tab when `RecipeManagement.content` is in the main area.

7.3.9.1 Administration

Administration.content

The administration page is used to manage and set the user interface and contains the following components:

1. TcHmiUserManagement control to logout, change users, edit user properties and manage users
2. Area for changing the language with one TcHmiToggleButton control per language and flag respectively
3. Area for displaying the software versions
4. MeasurementUnitSelector control for toggling the units in the Plastic Application HMI project



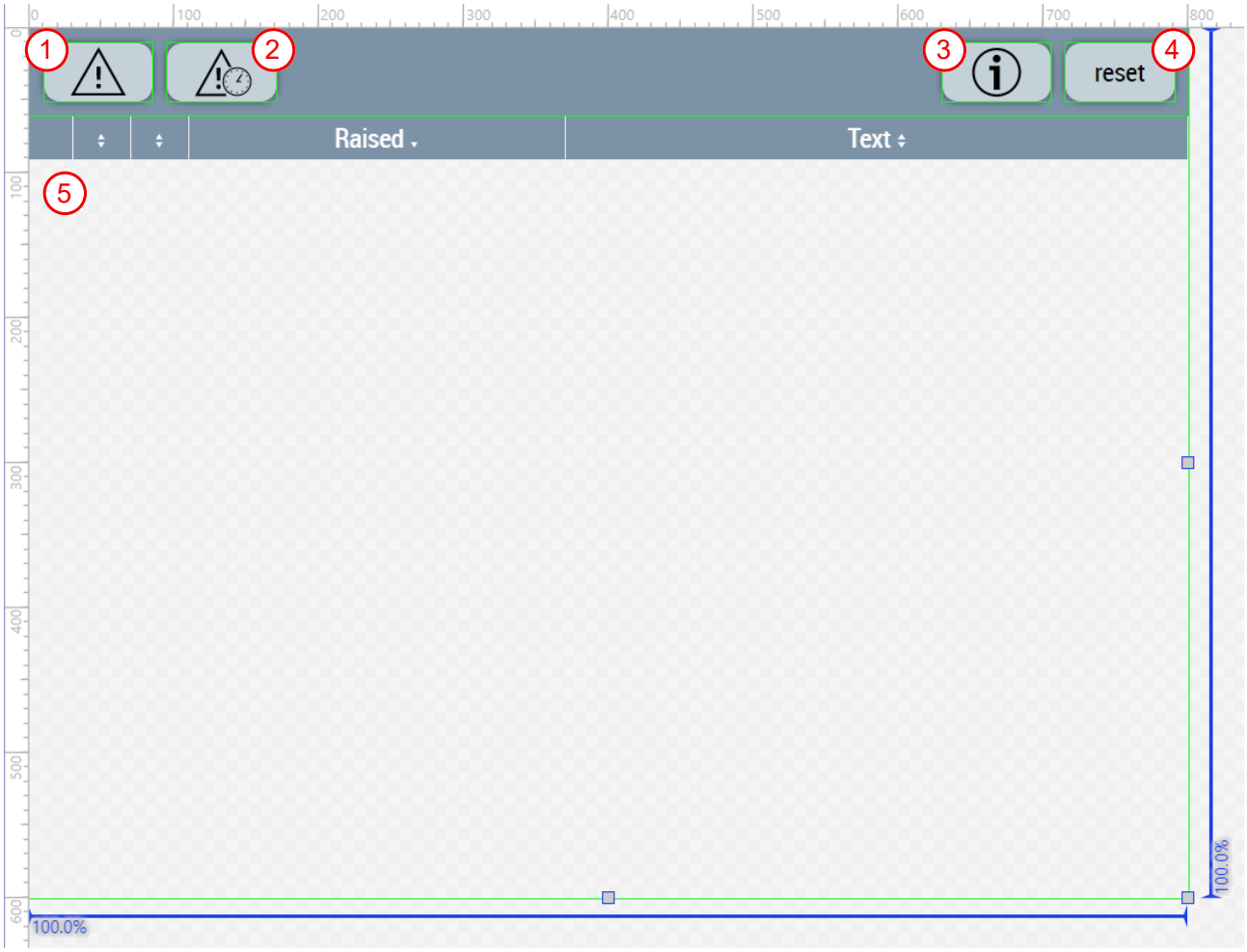
Available from version 12.5.1

7.3.9.2 Events / Alarms

Alarms.content

The alarm and event page contains a TcHmiEventGrid control that displays all events and alarms transmitted by the PLC by means of an appropriate configuration. The control offers further filter settings to limit the display of events. The events can be viewed and acknowledged in more detail. Additionally implemented TcHmiButton controls simplify the TcHmiEventGrid control with predefined filters. The page contains the following components:

1. Button to display the pending alarms
2. Button to display all alarms so far
3. Button to display the messages
4. Button to reset the pending alarms
5. Area for listing the alarms and messages using the TcHmiEventGrid control



Available from version 12.6.0

7.3.9.3 Recipe management

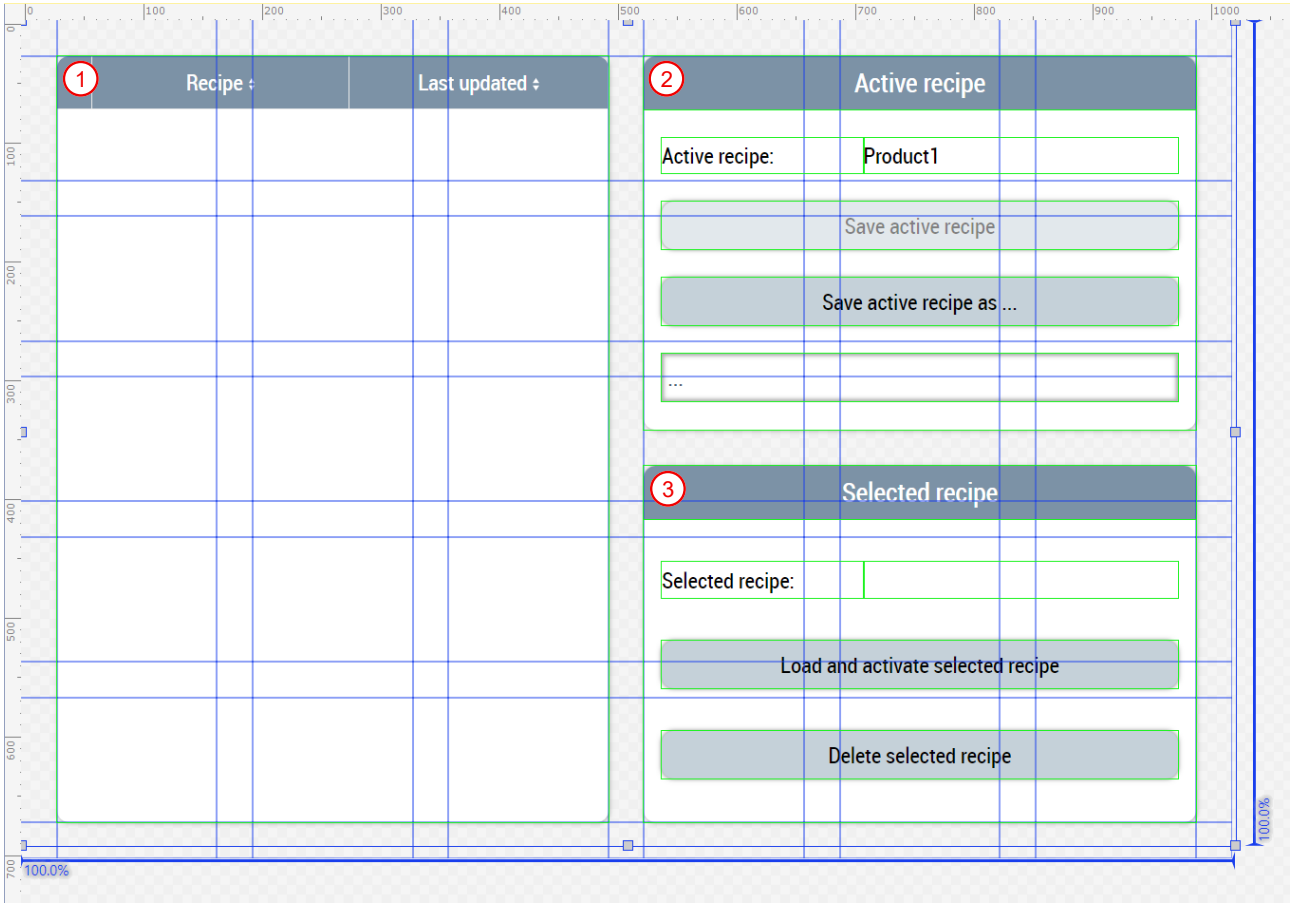
RecipeManagement.content

The recipe management page consists of the following elements:

1. TcHmiDatagrid control as a table to display all recipes with the date of the last update
2. Area for the active recipe:

- Display of the name of the active recipe
- TcHmiButton control for saving the changes to the active recipe
- TcHmiButton control for saving the current values into a new recipe
- TcHmiTextbox control for entering the new recipe name

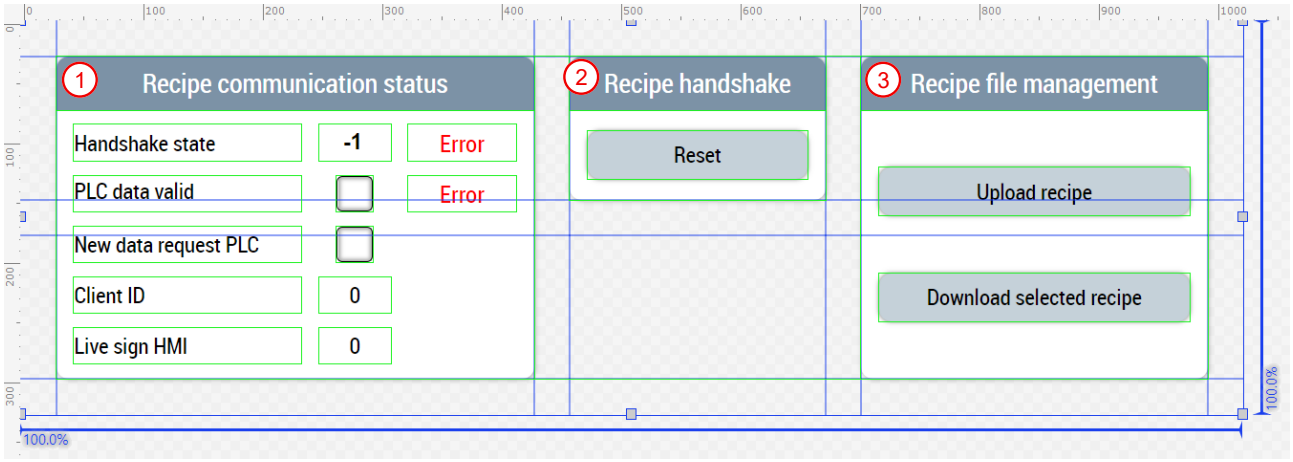
3. Area for the selected recipe:
   - Display of the name of the selected recipe in the table
   - TcHmiButton control for loading and activating the selected recipe
   - TcHmiButton control for deleting the selected recipe



RecipeManagement_Settings.content

This content is displayed in the slider area and serves as a supplement. This consists of the following elements:

1. Display of the recipe communication status
2. TcHmiButton control for resetting the recipe communication
3. Recipe file management area:
   - TcHmiButton control for uploading recipes
   - TcHmiButton control to download the selected recipe



Available from version 12.5.1

7.3.10 Temperature

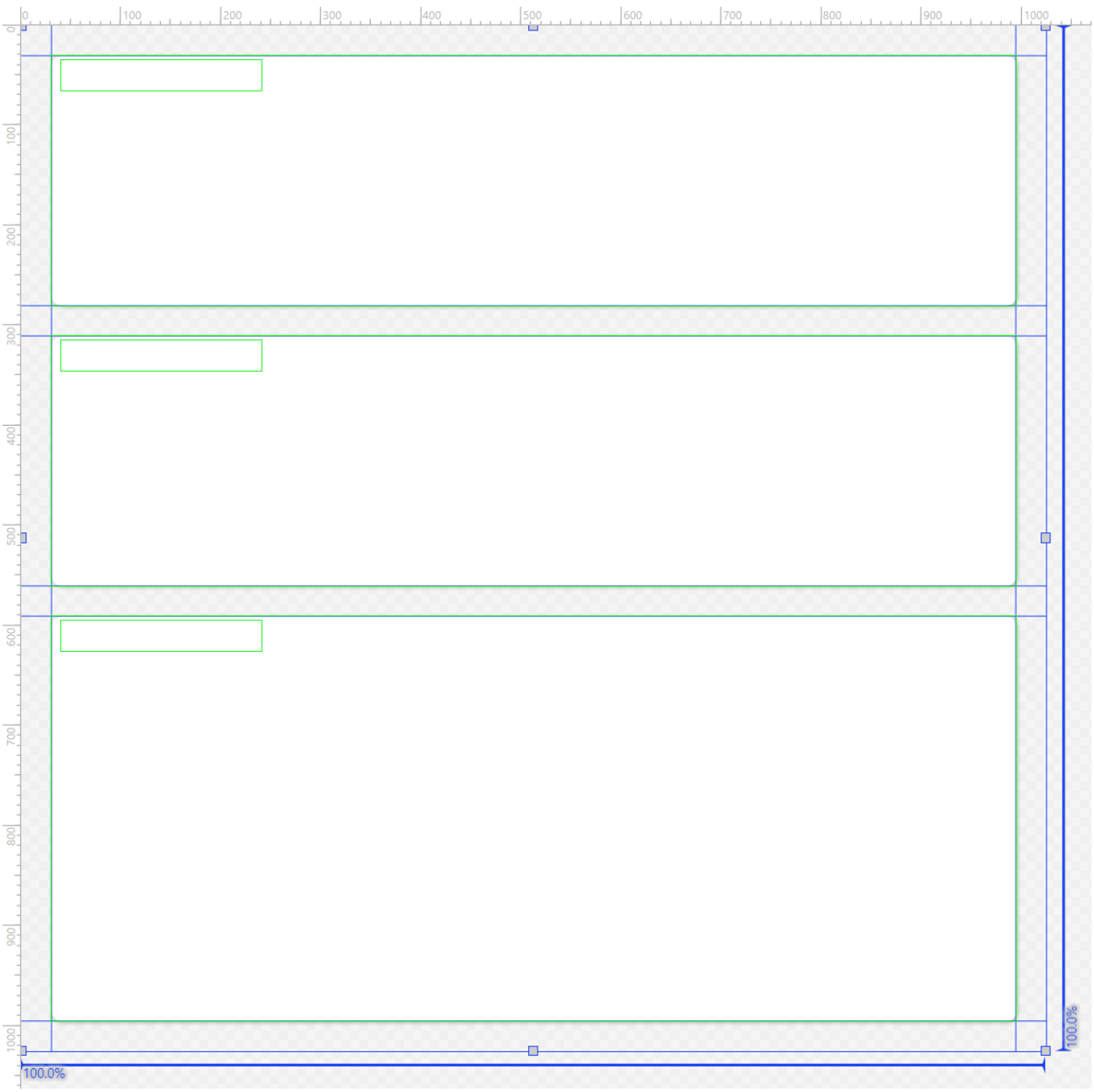
The folders named *Temperature* (*Contents/Navigation/Temperature* and *Contents/Slider/Temperature*) contain all contents needed for the temperature control on the machine.

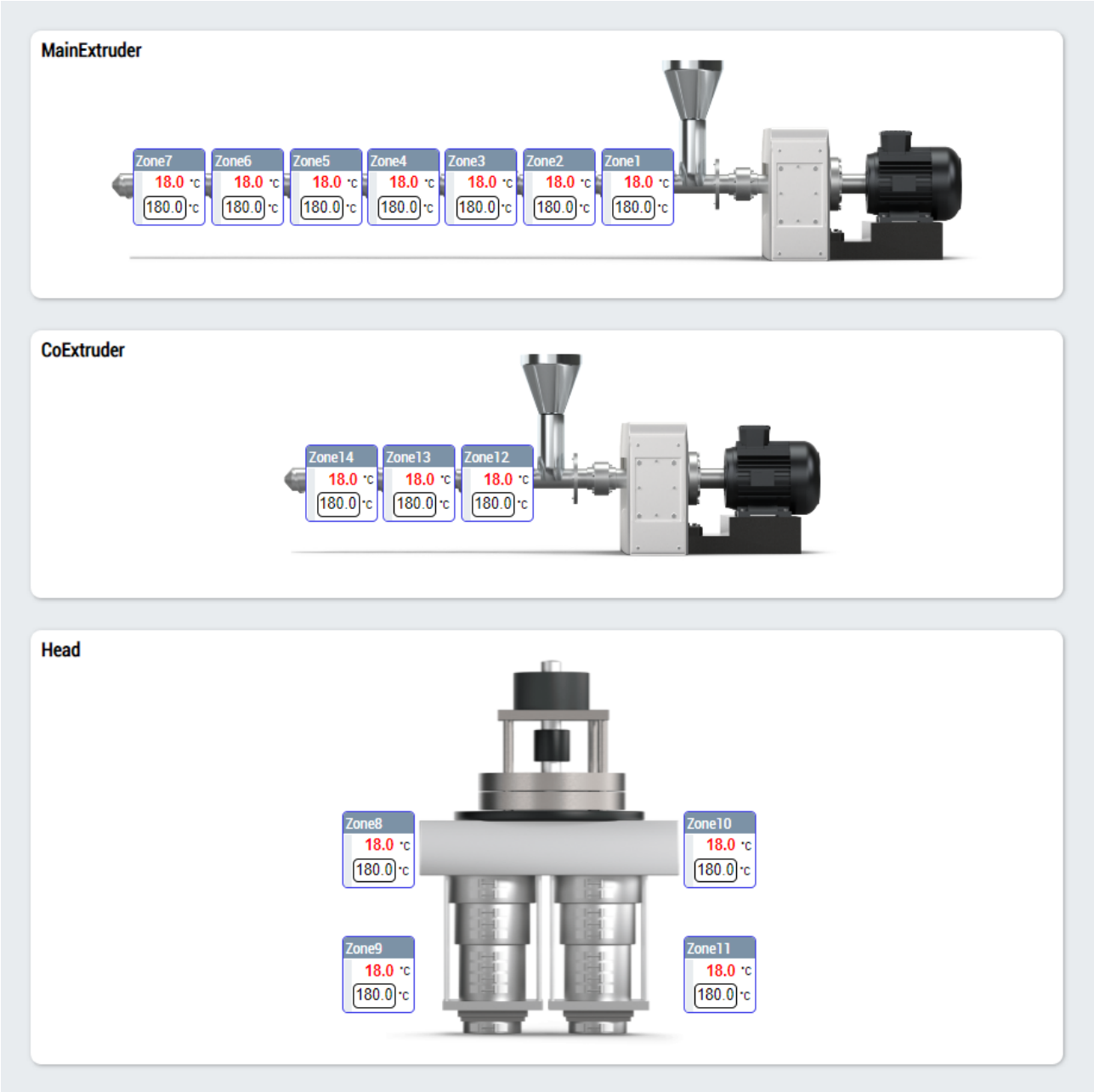
Content	Description
Temperature_Overview.content [▶ 175]	The page for single zone monitoring of temperatures.
Temperature_Overview_Settings.content [▶ 175]	Displayed in the slider area under the 4th tab when `Temperature_Overview.content` is in the main area.
Temperature_Configuration.content [▶ 178]	The page for setting the temperature zone parameters.
Temperature_Configuration_Settings.content [▶ 178]	Displayed in the slider area under the 4th tab when `Temperature_Configuration.content` is in the main area.
Temperature_TimeScheduling.content [▶ 180]	The page for time control of the temperature zones and temperature groups.
Temperature_TimeScheduling_Settings.content [▶ 180]	Displayed in the slider area under the 4th tab when `Temperature_TimeScheduling.content` is in the main area.
Temperature_Grouping.content [▶ 181]	The page for grouping the temperature channels into temperature groups.
Temperature_Layout.content [▶ 183]	The page for creating layouts with temperature zones.
Temperature_Layout_Settings.content [▶ 183]	Displayed in the slider area under the 4th tab when `Temperature_Layout.content` is in the main area.

7.3.10.1 Overview

Temperature_Overview.content

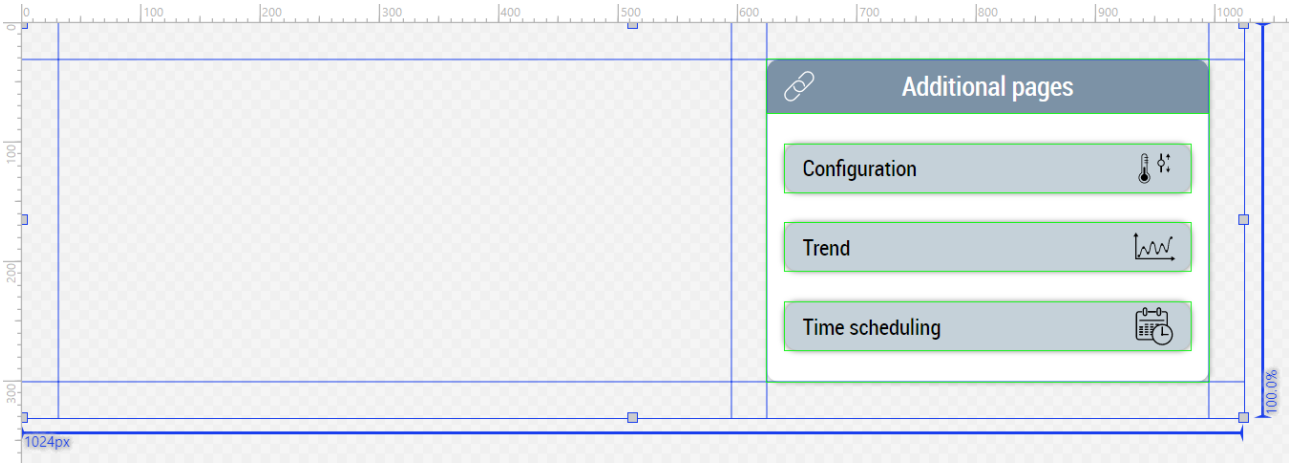
This page is used for the temperature zone overview on the machine. Using the ZoneImageLayout control in display mode, the configurations from the Layout [▶ 183] page can be displayed on this page and the temperatures can be set. Clicking on the temperature group name in the top left corner of the two extruders (MainExtruder [▶ 159] and CoExtruder [▶ 160]) navigates to the associated overview page.


0
100
200
300
400
500
600
700
800
900
1000
100.0%
100.0%



Temperature_Overview_Settings.content

In addition to the temperature overview page, there is a supplementary page in the slider area that is used to navigate to other related pages.

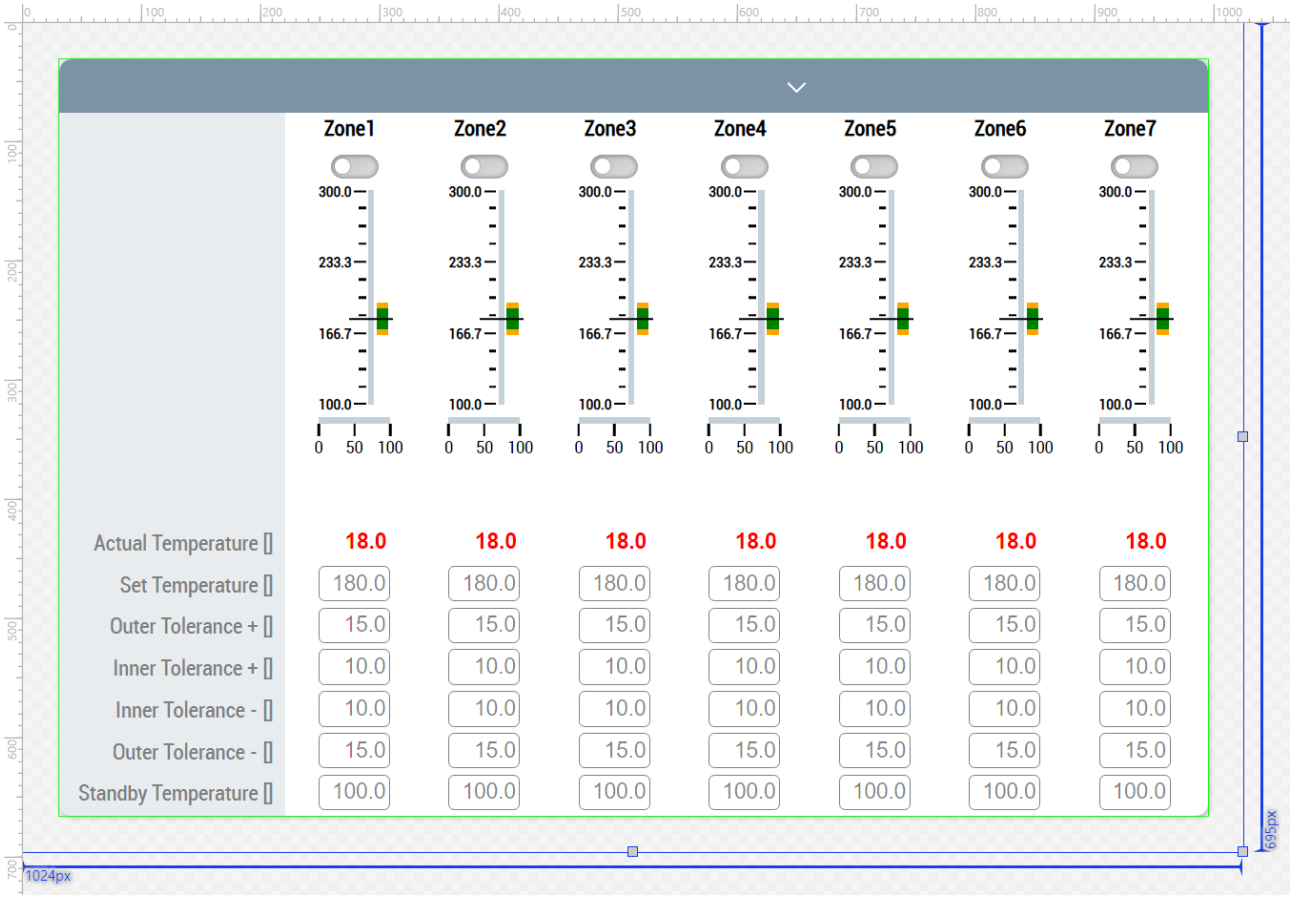


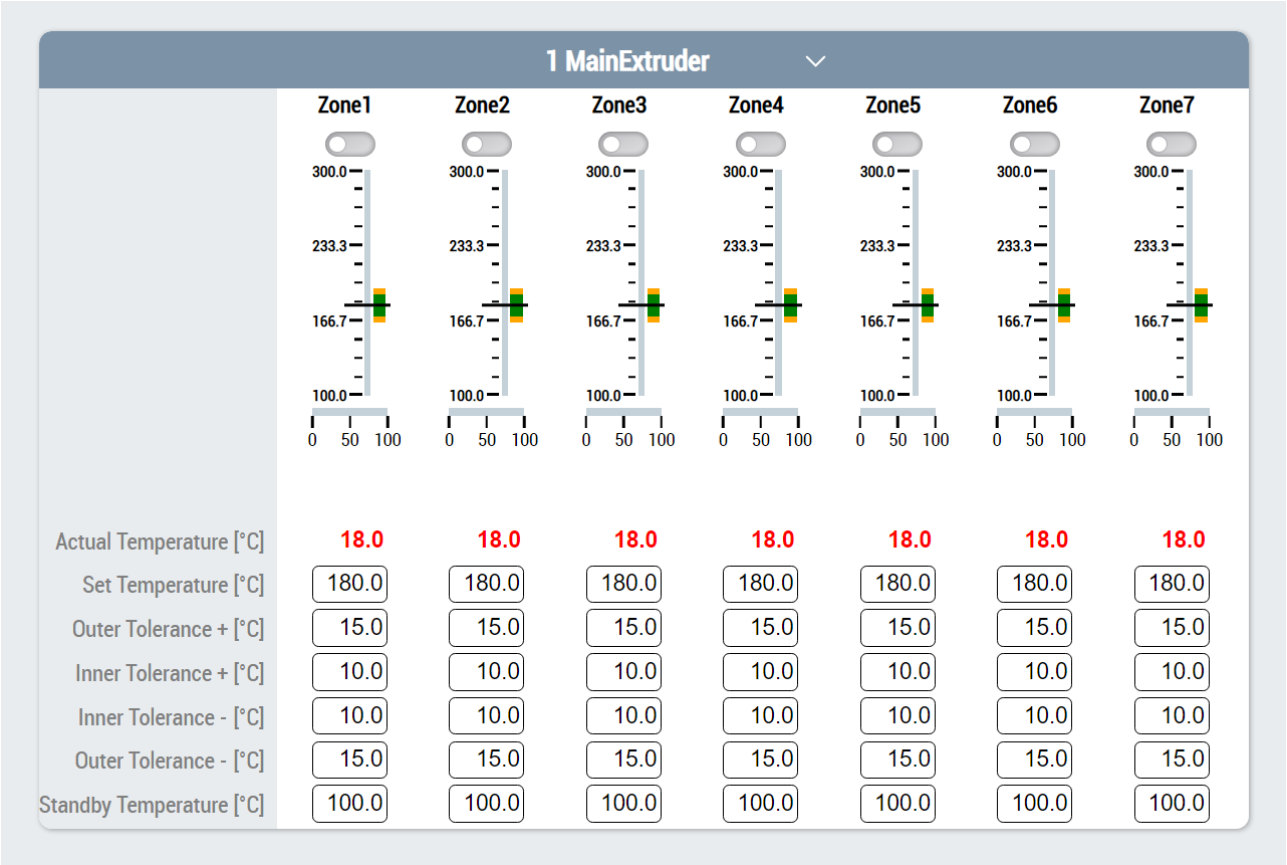
Available from version 12.6.0

7.3.10.2 Configuration

Temperature_Configuration.content

The configuration page of the temperatures contains the ZoneConfiguration control and is used for grouped display and setting of the temperature zones. An instance of FB_TemperatureHmi [▶ 55] is required to use the control.

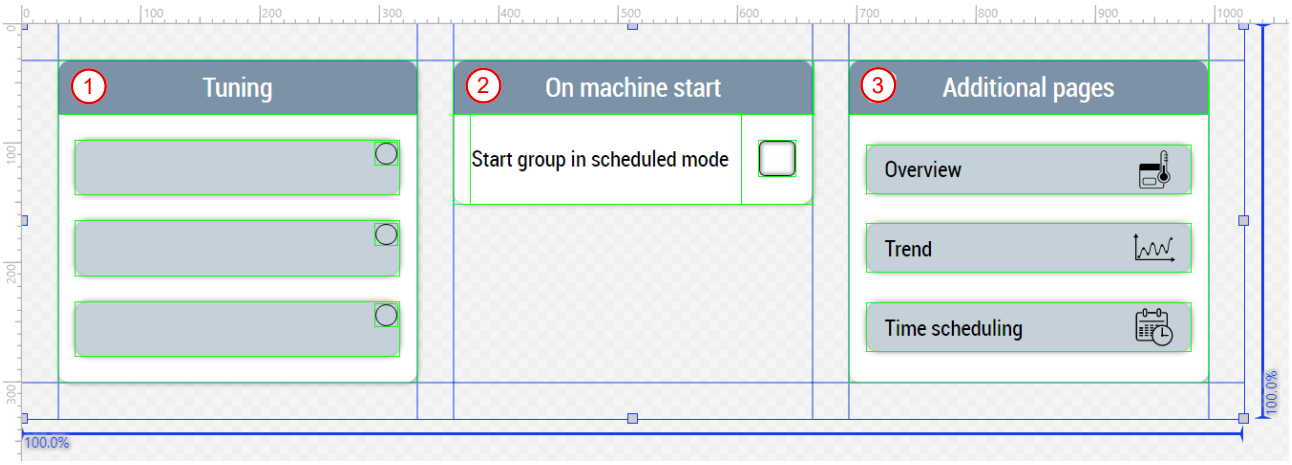


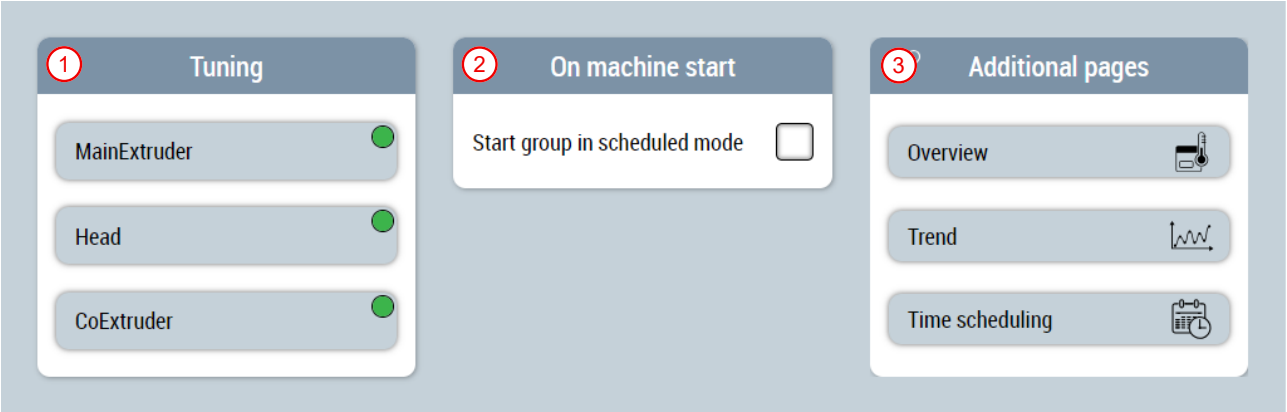


Temperature_Configuration_Settings.content

This page serves as a supplement in the slider area and contains the following components:

1. Tuning of the temperature zone group and status display of tuning
2. Settings that should take effect at machine startup
3. Area for navigation to other related pages



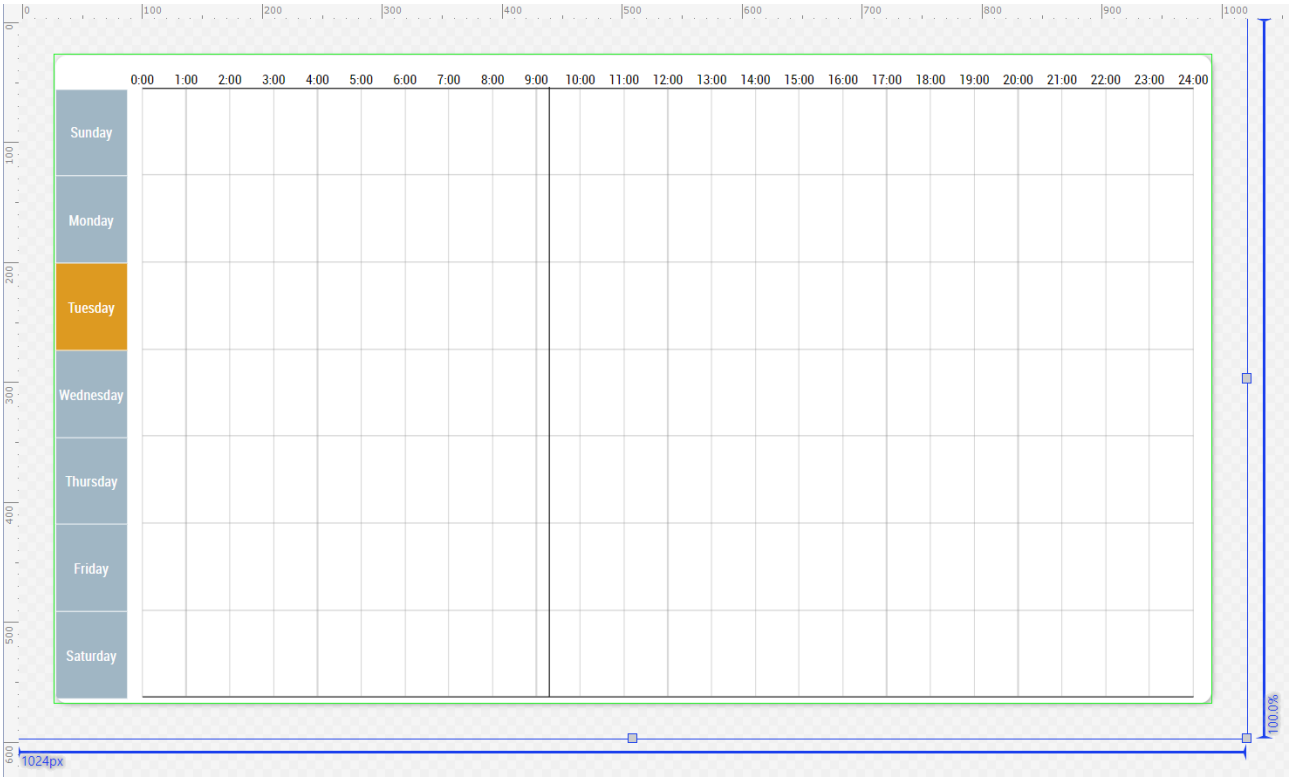


Available from version 12.6.0

7.3.10.3 Scheduling

Temperature_TimeScheduling.content

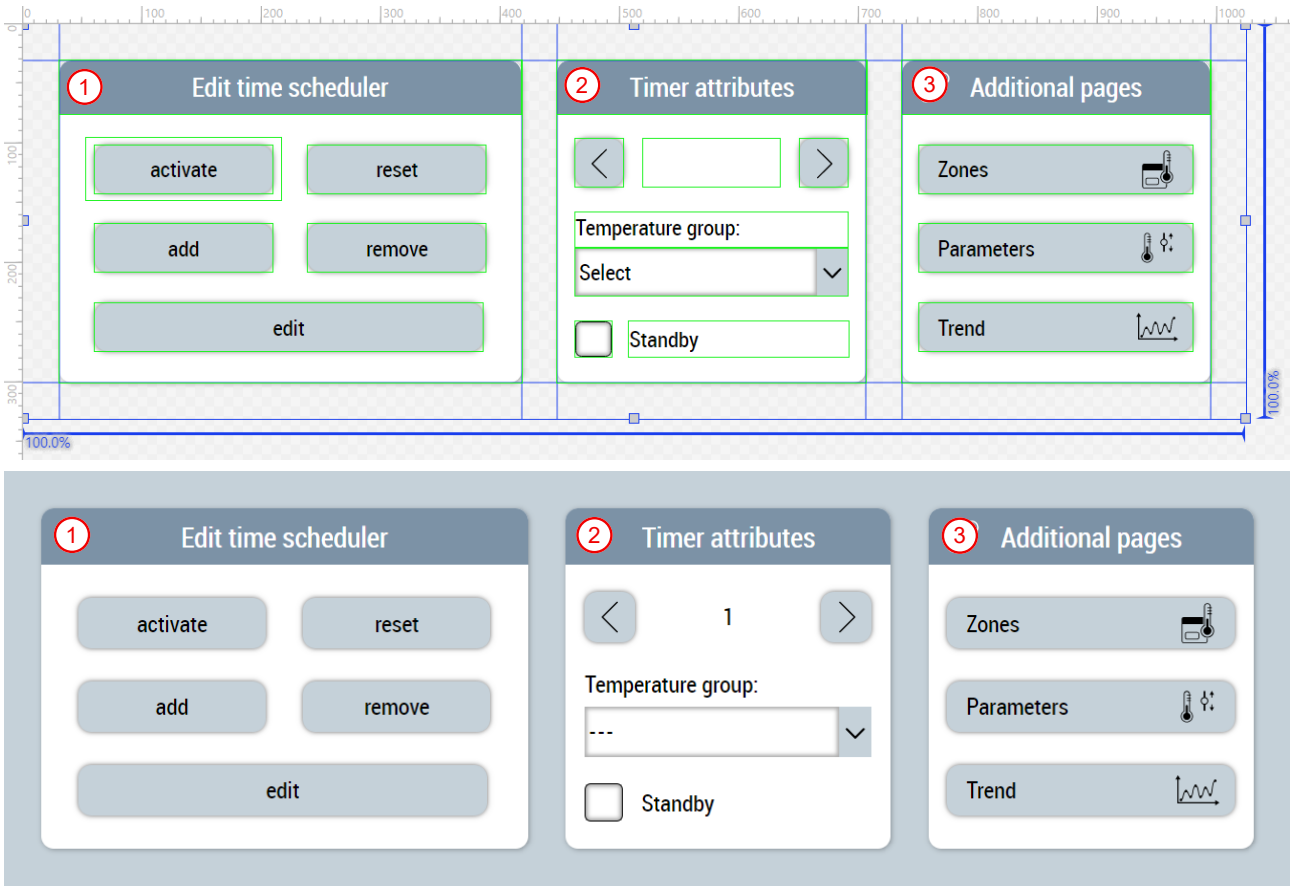
The page for scheduling the temperature zones and temperature groups contains a ProcessScheduler control. The control requires an instance of the FB_TempScheduleHmi [▶ 68] for the `TimerList` property of the `Scheduler` category.



Temperature_TimeScheduling_Settings.content

The page serves as a supplement in the slider area and includes the following functionalities:

1. Activate and reset the changes, add new elements, remove or edit the selected element
2. Selection of the timer via arrow keys, setting of the temperature group via the combo box and selection of the standby mode via the checkbox
3. Area for navigation to other related pages



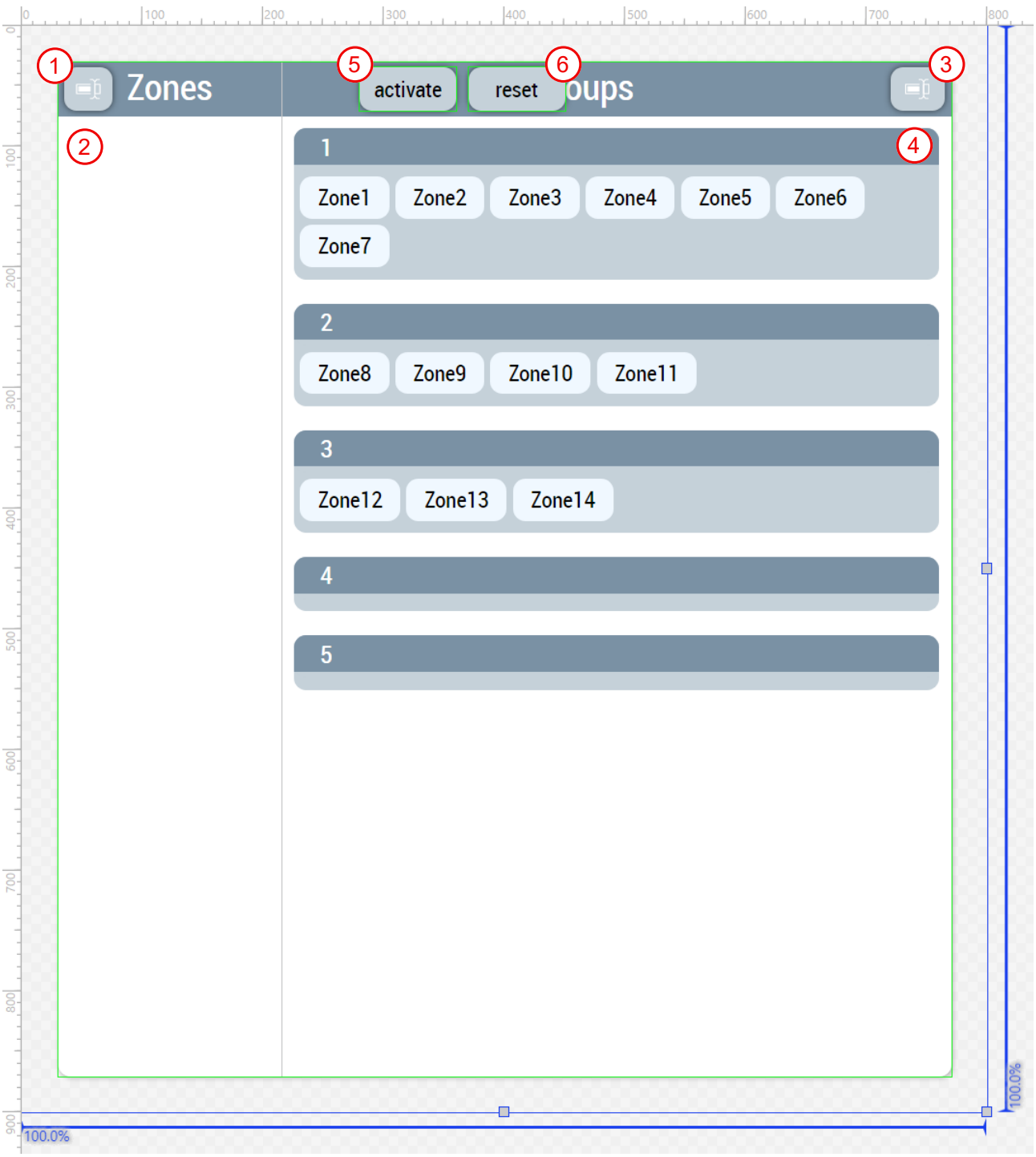
Available from version 12.6.0

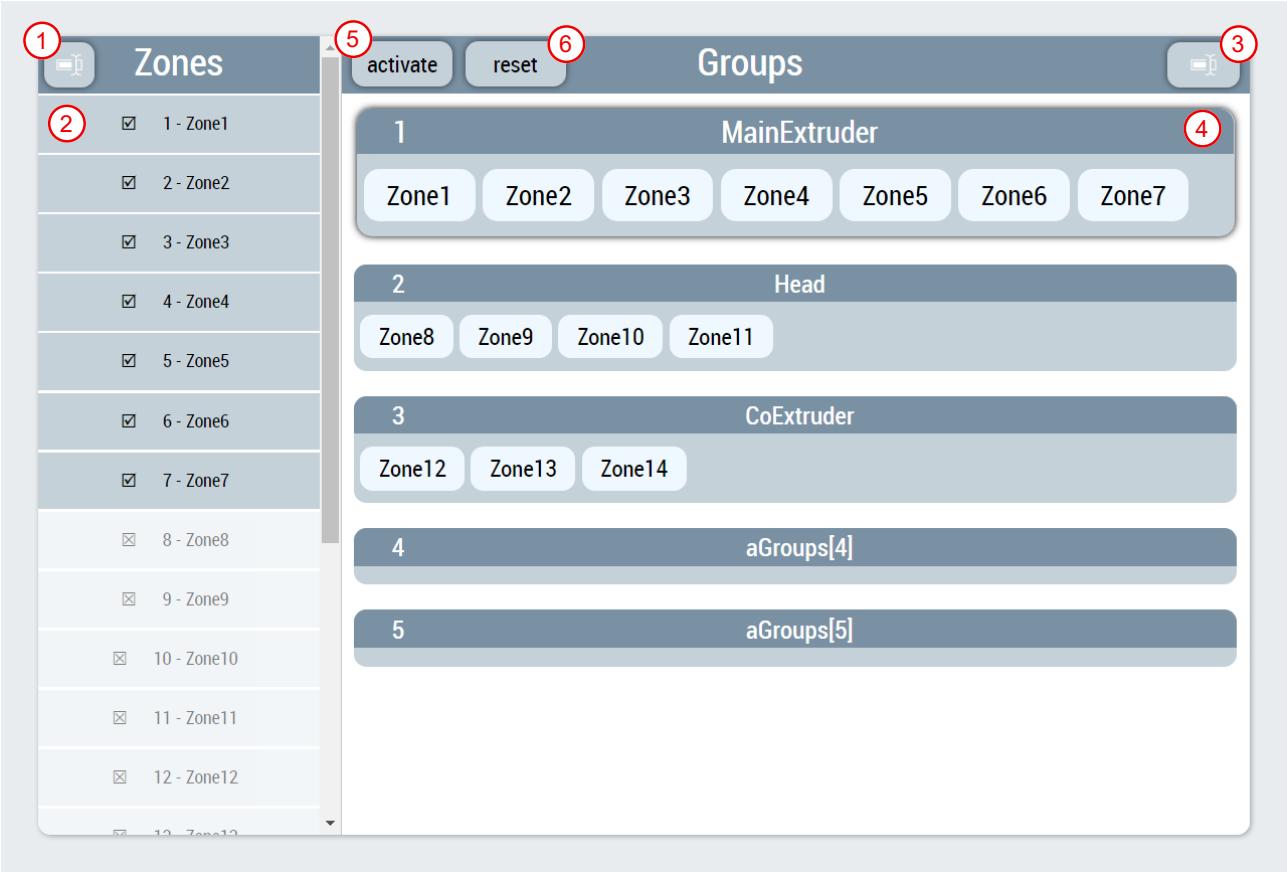
7.3.10.4 Grouping

Temperature_Grouping.content

This page is used to group the temperature channels. Only the temperature channels that are enabled via the temperature parameter page [▶ 162] are made available in the list. The page consists of the following functions:

1. Adjustment of the temperature zone names
2. List of available temperature channels and display of grouped temperature channels of the selected group (4)
3. Customization of the temperature group names
4. Display of the temperature groups with the grouped temperature zones and selection of the temperature group by clicking for adjustment via the temperature zone list (2)
5. Activation of the change
6. Resetting the changes


1
Zones
5
activate
6
reset
oups
3
2
1
4
Zone1
Zone2
Zone3
Zone4
Zone5
Zone6
Zone7
2
Zone8
Zone9
Zone10
Zone11
3
Zone12
Zone13
Zone14
4
5
100.0%
100.0%



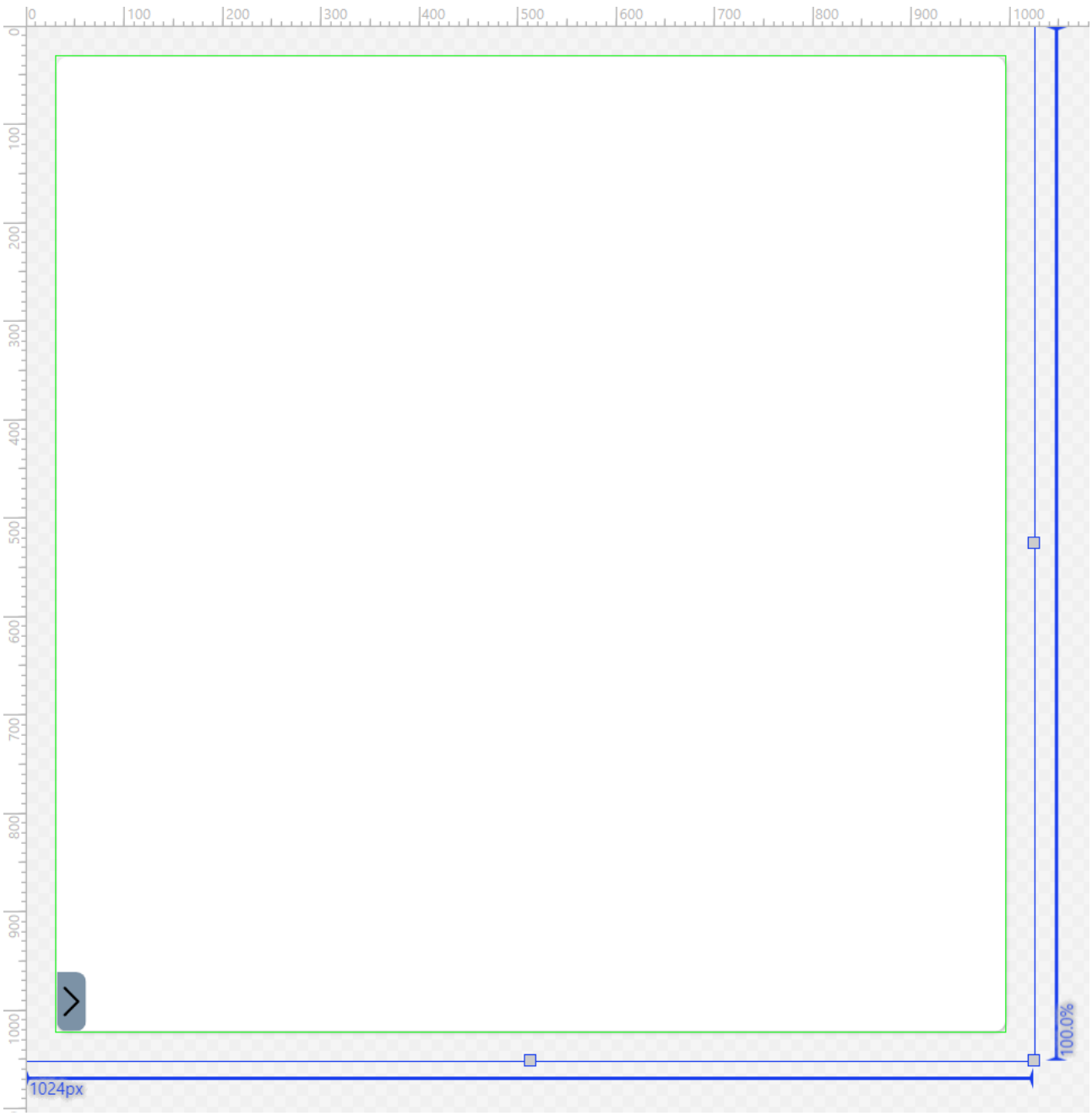
Available from version 12.6.0

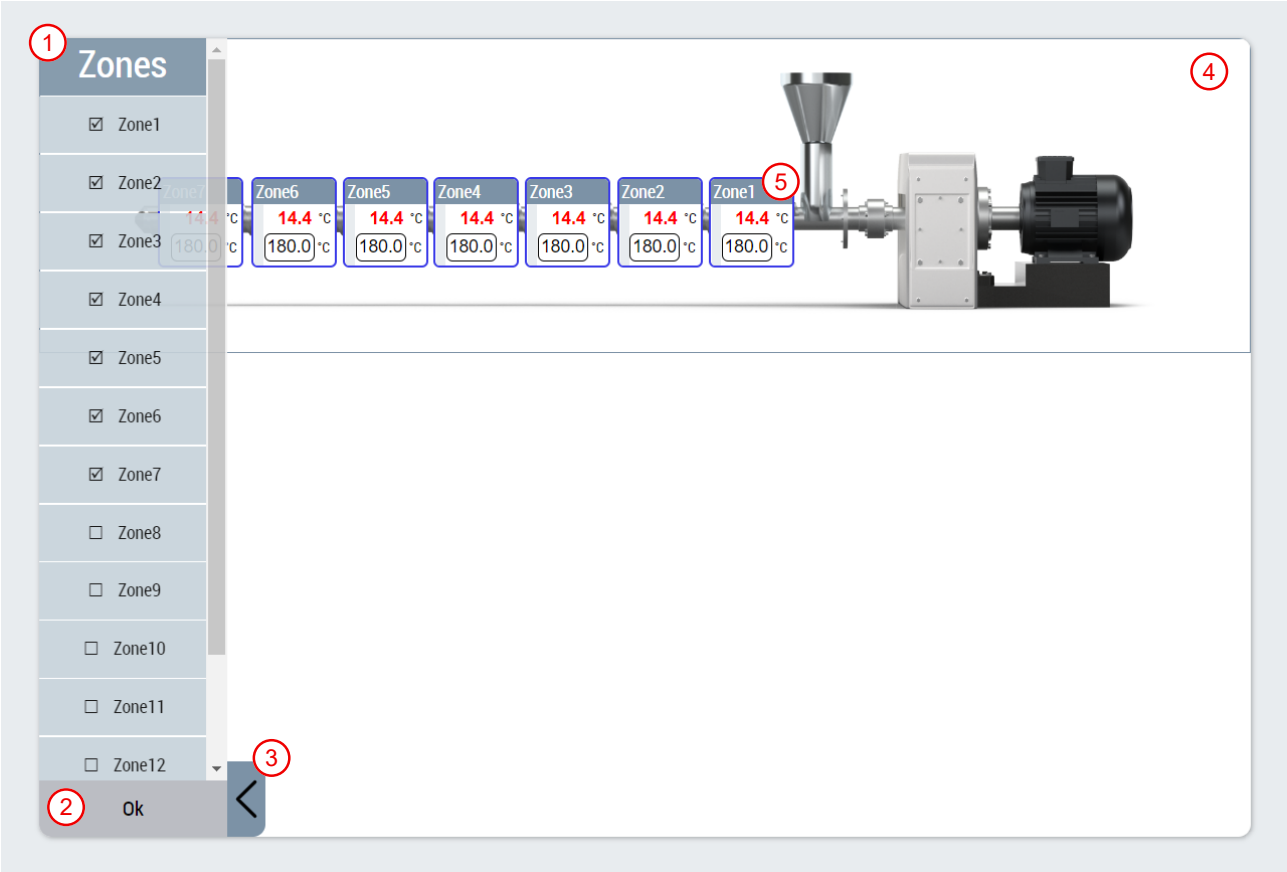
7.3.10.5 Layout

Temperature_Layout.content

This page is used to create layouts that can be reused throughout the HMI. A background image can be selected and customized for each layout added, and temperature zones can be selected and positioned on the image as desired. Only the temperature zones that are assigned to a group via the temperature grouping page [▶ 181] are made available in the list. The page consists of the following functions:

1. List of available grouped temperature zones
2. Activate the changes in the list with the button
3. Closing the temperature zone list by the arrow key
4. Display of the layout selected via the combo box in the slider area
5. Positioned display of selected temperature zones

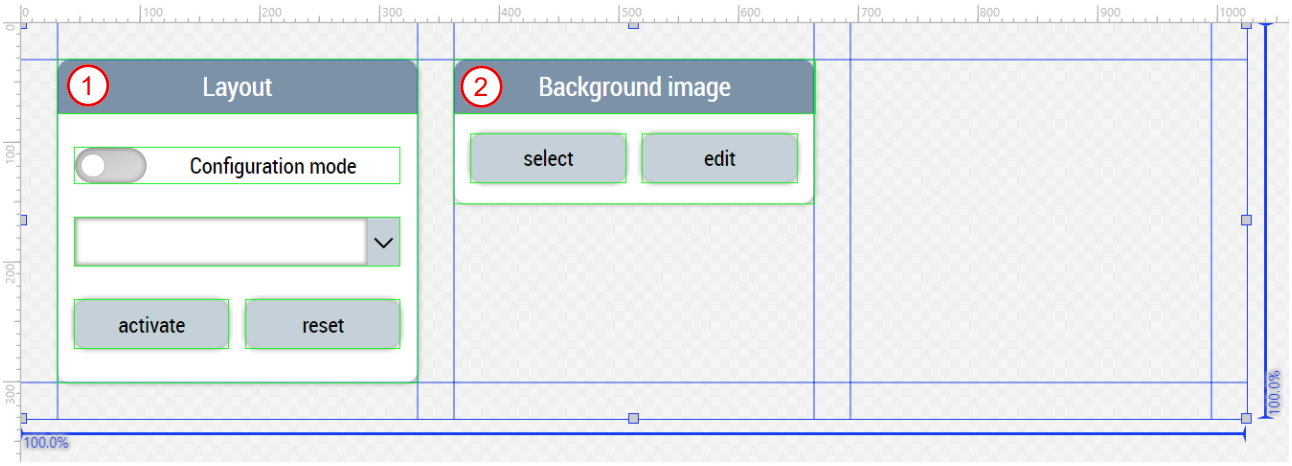

0
100
200
300
400
500
600
700
800
900
1000
100.0%
1024px

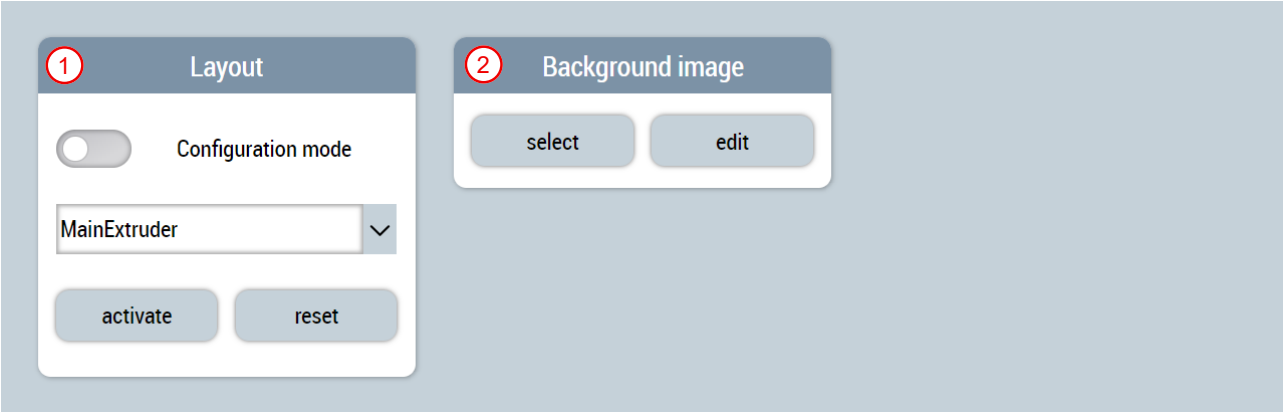


Temperature_Layout_Settings.content

The page serves as a supplement in the slider area and includes the following functionalities:

1. Switching on the configuration mode, selecting the layout via a combo box, activating and resetting the changes
2. Select background image and edit background image properties via separate pop-up windows





Available from version 12.6.0

7.3.11 WTC

The folders named *Wtc* (*Contents/Navigation/Wtc* and *Contents/Slider/Wtc*) contain all contents needed for the wall thickness control (WTC) of the machine.

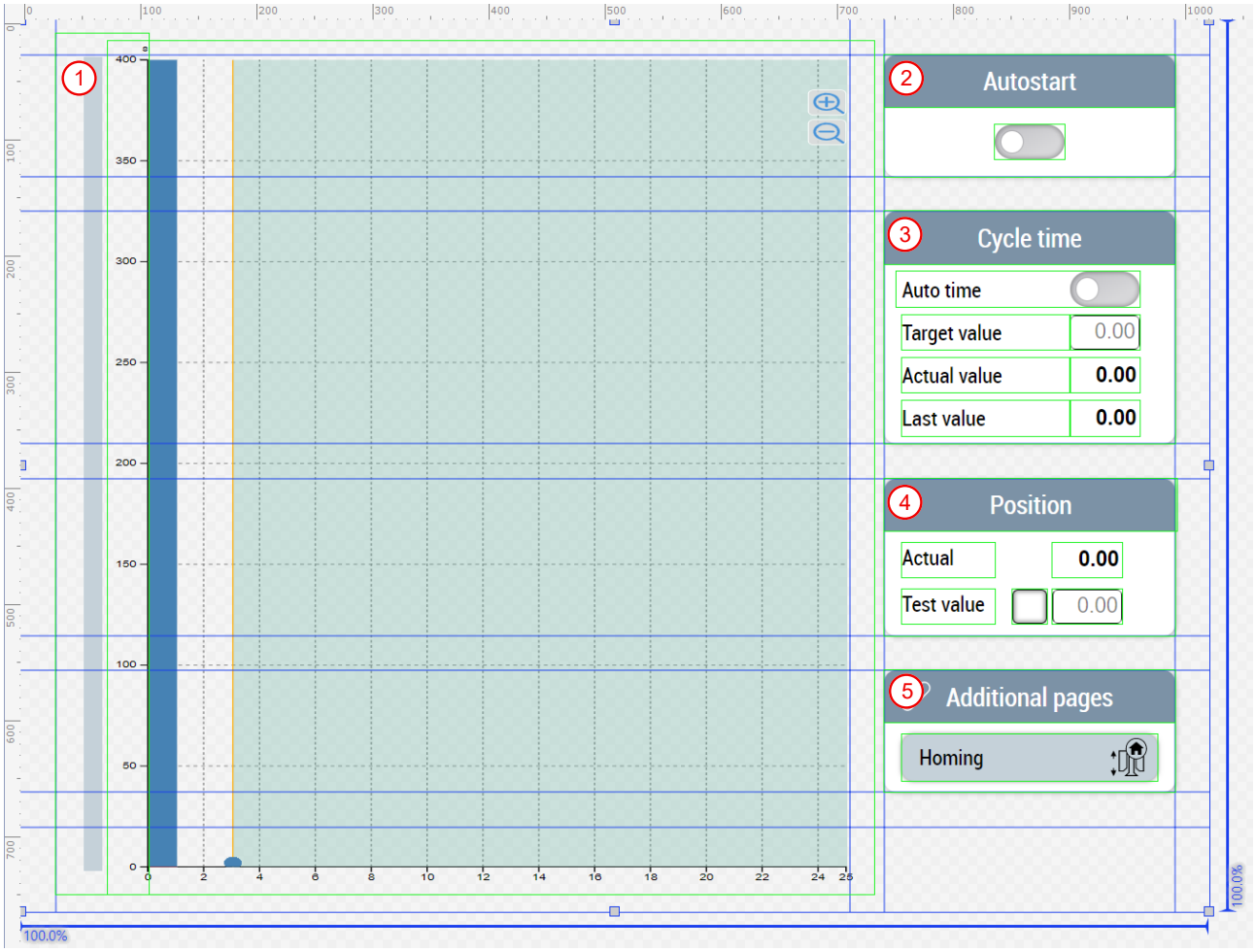
Content	Description
Wtc.content [▶ 186]	The main page of the WTC.
Wtc_Settings.content [▶ 186]	Displayed in the slider area under the 4th tab when `Wtc.content` is in the main area.
Wtc_Homing.content [▶ 188]	The homing page of the WTC.
Wtc_Homing_Settings.content [▶ 188]	Displayed in the slider area under the 4th tab when `Wtc_Homing.content` is in the main area.

7.3.11.1 Curve

Wtc.content

The main content of the WTC consists of the following components:

1. WTC curve using the CurveEditor control
2. Function for Autostart
3. Area for settings and displays related to the cycle time
4. Area for displaying the current position and setting a test position
5. Area for navigation to a related page

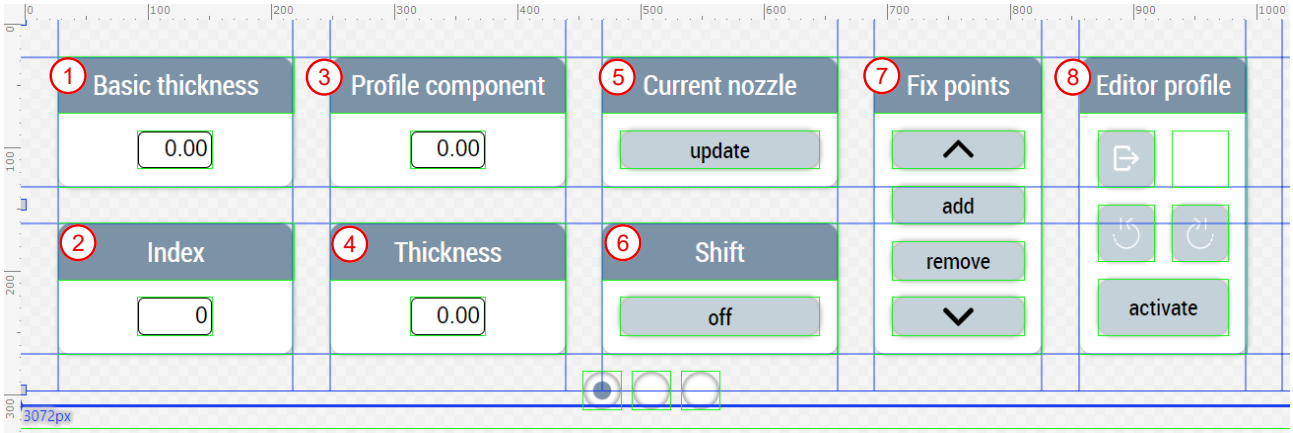


Wtc_Settings.content

There is a supplementary page for the slider area in addition to the main page of the WTC. This content is split into three pages.

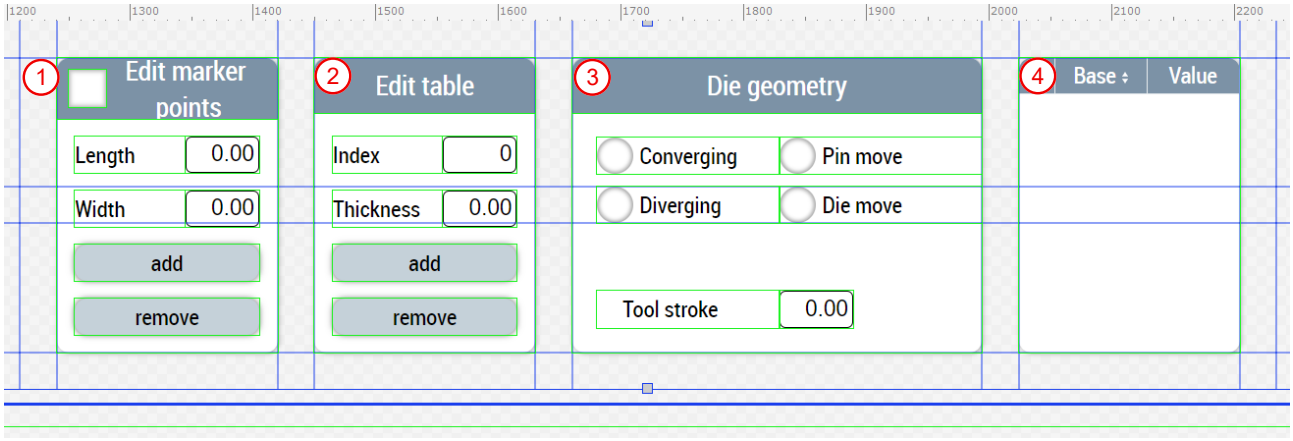
The first page contains the following components:

1. Setting or displaying the basic thickness of the entire profile
2. Setting or displaying the index of the selected point
3. Setting or displaying the profile component
4. Setting or displaying the thickness of the selected point
5. Possibility to update the setting according to the currently selected nozzle
6. Switching the shift function on and off
7. Navigate between existing fix points and add and remove fix points
8. Undo changes, navigate between activated profiles and activation of the displayed profile



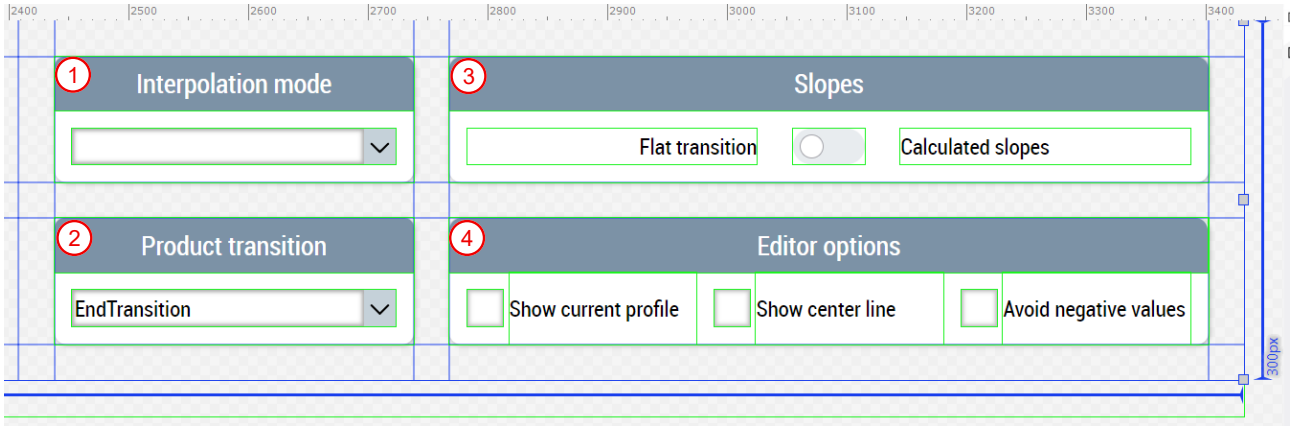
The second page contains the following components:

1. Add, remove, and edit marker points, and choose how to use them
2. Add, remove and edit fix points
3. Adjustment of die geometry by switching between Converging and Diverging, switching between Pin move and Die move, and adjustment of tool stroke
4. Table for displaying the fix points



The third page contains the following components:

1. Setting or displaying the interpolation mode
2. Setting or displaying the product transition
3. Adjustment of the slopes
4. Adjustment of the editor options



Available from version 12.5.1

7.3.11.2 Homing

Wtc_Homing.content

The homing page for the WTC consists of the following components:

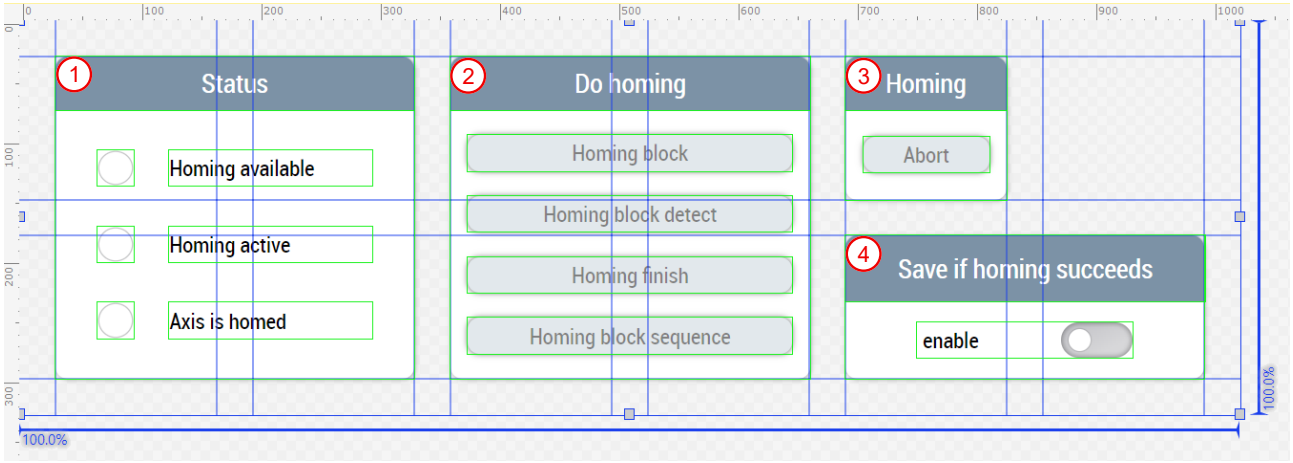
1. Homing block
2. Homing block detection
3. Homing finish
4. Encoder zero offset shift

Using the `UpdateAxesHomingUnits` function from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package, certain units of the values are converted to angle unit groups if it is a transformed axis.

Wtc_Homing_Settings.content

There is also a supplementary page for the slider area in addition to the homing page of the WTC. This page contains the following components:

1. Homing status display
2. Homing functions
3. Aborting the homing
4. Activation of the storage option



Available from version 12.6.0

7.4 Localization

The folder *Localization* manages the different languages that should be available in the project. New languages can be added. In addition, new *Localized Symbols* can be created, for which the translation can then be entered in all required languages.

Installing the `Beckhoff.TwinCAT.HMI.Plastic.Localizations` NuGet package provides the Plastic Application HMI project with language keys that can be used in the project.

Update capability of the Plastic Application HMI

Editing the Plastic Application HMI can lead to an impairment of the update capability. This may affect the support provided by Beckhoff Automation.

Available from version 12.5.1

7.5 View

A View is a file with the extension `.view` and contains the overall structure of the user interface. A project can contain several Views. At the beginning it has to be defined in the engineering which View should be used as Start View to build the HMI. The structure of the user interface is defined by the View selected as Start View.

Update capability of the Plastic Application HMI

Editing the Plastic Application HMI can lead to an impairment of the update capability. This may affect the support provided by Beckhoff Automation.

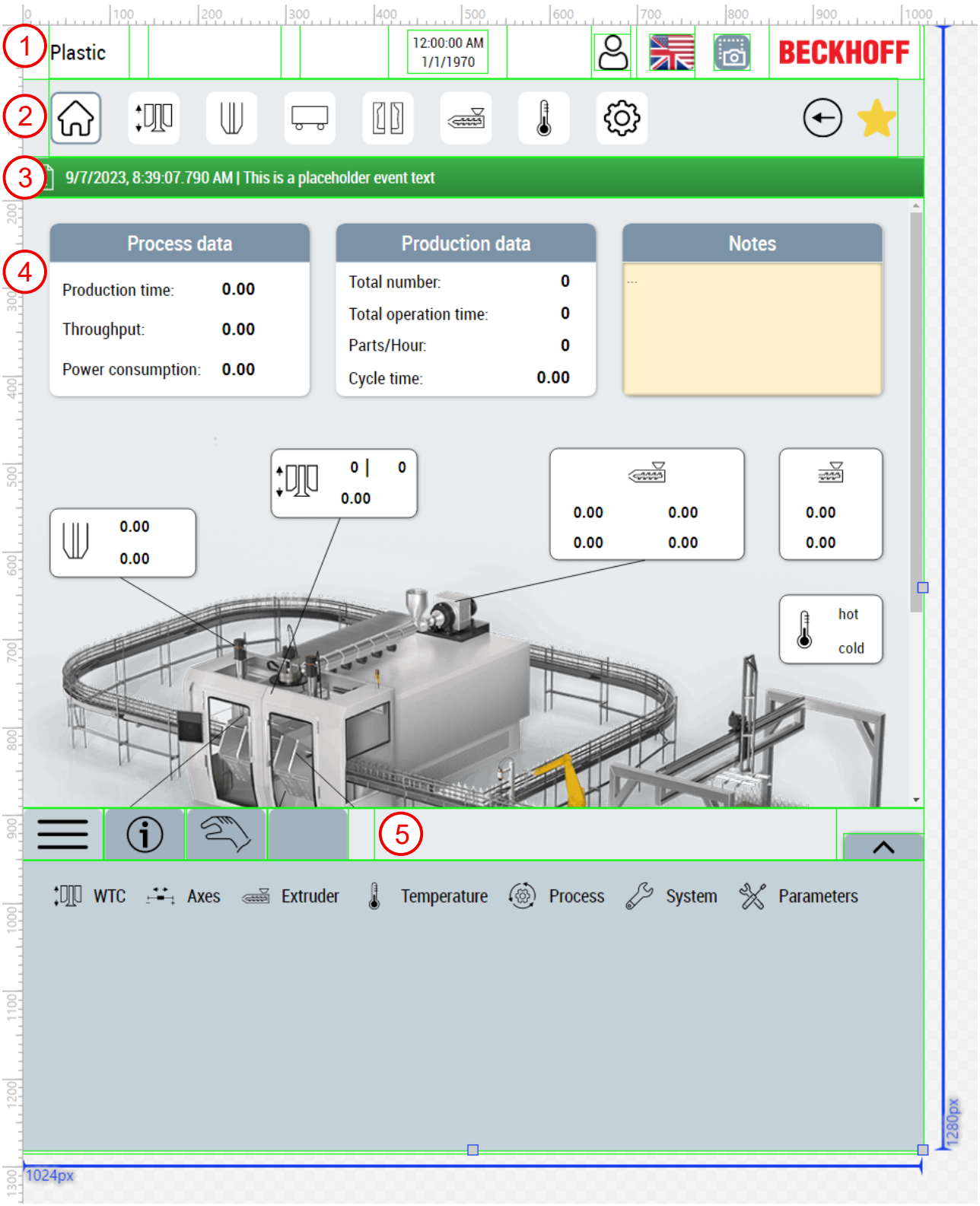
Portrait.view

Unless the `Portrait.view` file is modified, this user interface has a portrait aspect ratio (4:5) with a width of 1024 px and a height of 1280 px. It consists of the following components:

1. **Header:**
   - Application name (click on it to display `Home.content` in the main area)
   - Currently loaded recipe (click on it to display `RecipeManagement.content` in the main area)
   - Current operation mode
   - Date and time

- Username (click on it to open submenu)
- Flag to display the selected language (click on it to open submenu)
- Screenshot function (enabled using the function `TakeScreenShot` from the Beckhoff.TwinCAT.HMI.Plastic.Functions NuGet package)
- Logo

2. **Favorites area:** Contains user-specific favorite buttons for quick navigation of frequently used pages.
3. **Event line:** Display of the latest upcoming alarm and a click on it shows `Alarms.content` in the main area.
4. **Main area:** Display of the selected content.
5. **Slider area:** With the help of the ToggleSliderArea, ShowSliderArea and HideSliderArea functions from the Beckhoff.TwinCAT.HMI.Plastic.Functions NuGet package, the slider area can be shown and hidden via the arrow keys, shown by clicking on a tab or hidden for desired contents in the main area (e.g. `Scope.content`). Clicking on one of the following tabs will display the corresponding content in the slider area:
   - **1st tab:** `Navigation.content` contains all available contents of the project in comparison to the favorites area.
   - **2nd tab:** `Info.content` displays the most important machine data.
   - **3rd tab:** `ManualFunctions.content` contains the manual functions of the machine.
   - **4th tab:** Using the `UpdateSliderContentRegion` function from the `Beckhoff.TwinCAT.HMI.Plastic.Functions` NuGet package, the 4th tab can be optionally displayed and the icon and the content displayed in the slider area will be updated depending on the content displayed in the main area. Possible contents can be recognized by the name extension `_Settings.content`.



Configurator control

The Configurator control serves as a general control for setting control-wide configurations. The instance of the control only needs to exist and does not require visibility. The following attributes are required for further setting of the HMI:

- **UnitConfig:** Path to the JSON file with the corresponding unit switching scheme. The file is supplied by default with the Beckhoff.TwinCAT.HMI.Plastic.Controls NuGet package, but it can also be copied and modified so that a different path must be set.
- **NavigationConfig:** Setting of the navigation structure with additional information like icons, slider content etc.

Available from version 12.6.0

8 Appendix

8.1 Commissioning of the temperature control

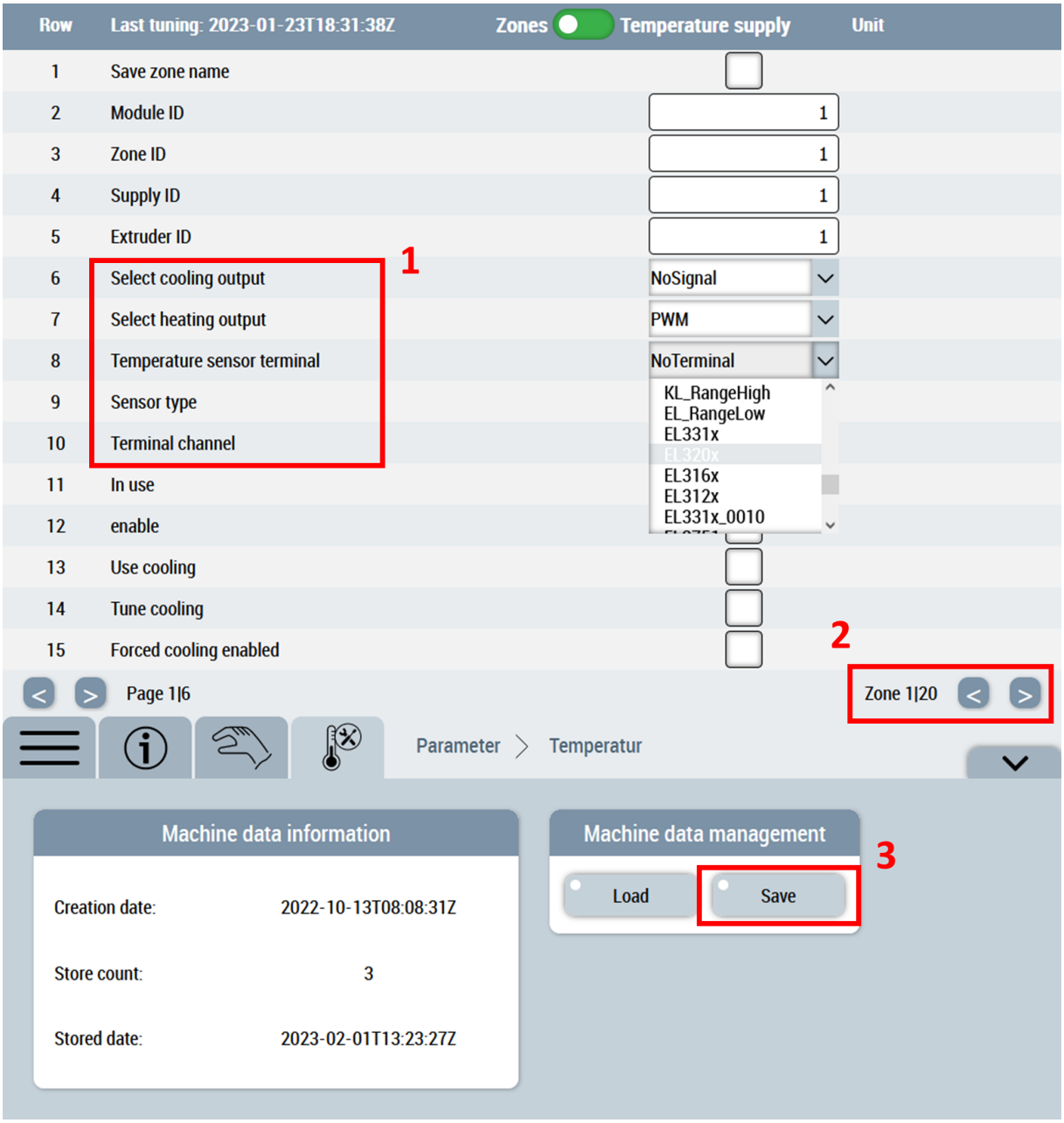
The commissioning of the temperature control includes both a TwinCAT Engineering part and a part to be performed at runtime. In this sample the individual steps that are performed at runtime are summarized.

Before the subsequent steps, the preparation of the TwinCAT project has to be done according to the example from section Mapping and configuration of temperature zones [▶ 135].

Set the appropriate input and output signal types and devices for all their temperature zones

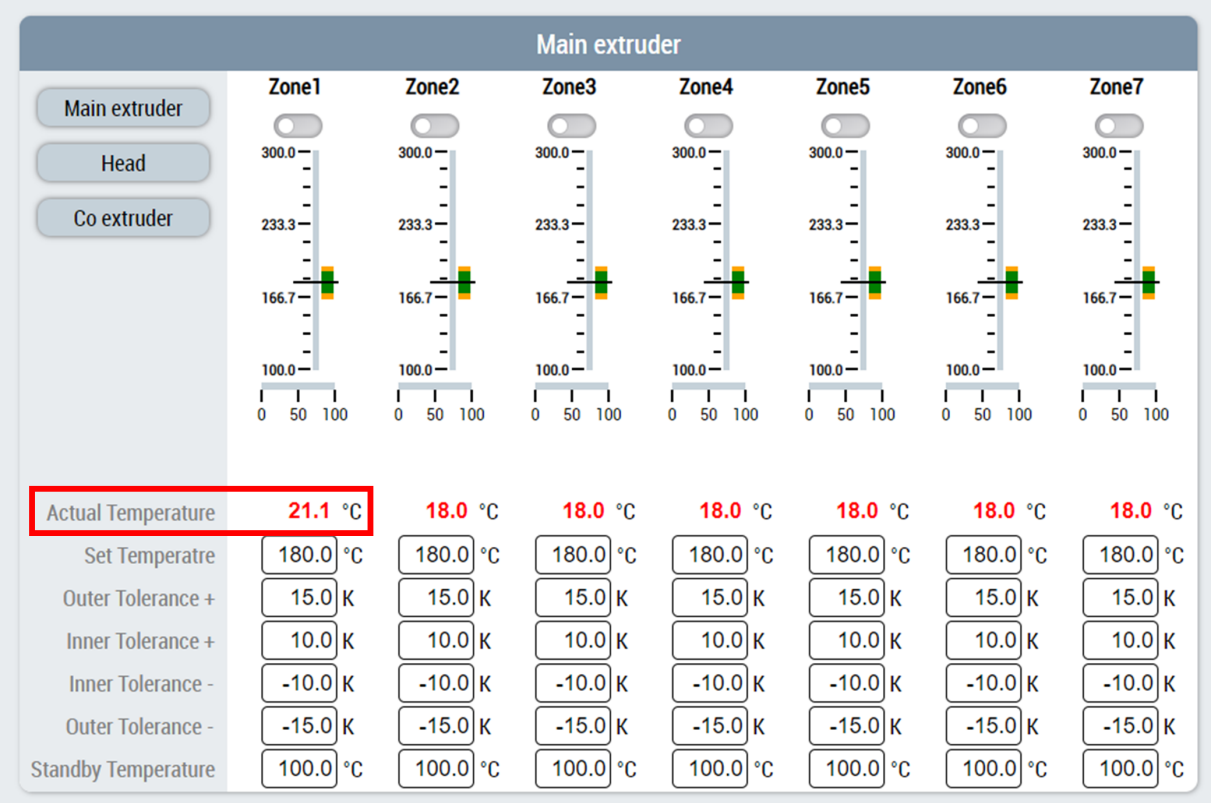
> **Hardware parameterization without grouping**
>
> This step refers to the linear mapping of the TF8540 library. Already configured groupings are ignored in this step.

1. Log in with administrator rights (default user = 4th administrator, password = 4).
2. Navigate via the navigation (≡-symbol) to *Parameter* > *Temperature.*
3. Set your hardware configuration for the first mapped zone at the parameters marked with (1).
4. Check the checkbox of the parameter **In Use** to validate the use of the zone.
5. Repeat step 3 and 4 for all zones to be used via the button marked with (2).
6. Save your settings via the button marked with (3).



Check the reaction of the hardware inputs on the machine

1. Navigate via the navigation to *Temperature* > *Parameters*.
2. Heat the sensor of a zone via an external heat source.
3. Observe if the temperature change occurs in the expected zone.
4. Repeat steps 2 and 3 for each zone.



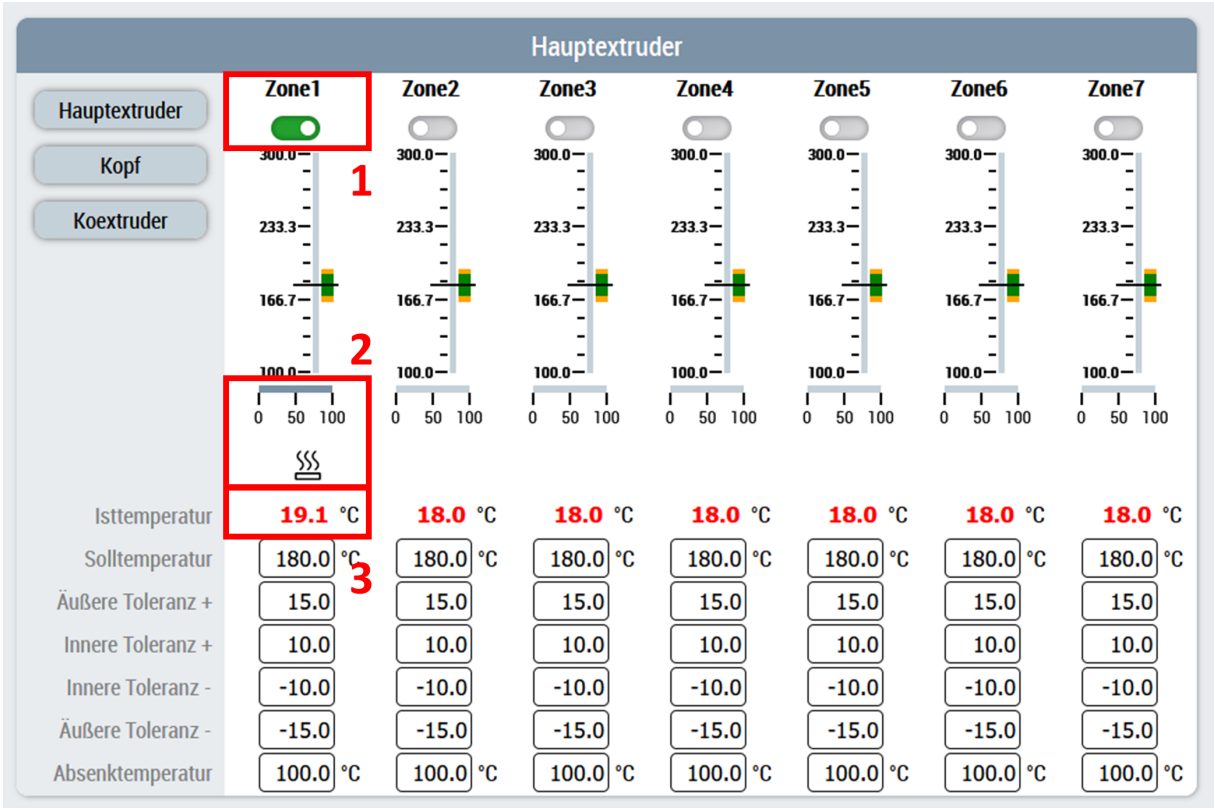
Check the response of the hardware outputs on the machine

Switching on a zone does not generate a power level

Make sure that at the time of this step the temperature control has been enabled by the PLC!

- To enable all zones execute the method FB_TempCtrl.EnableAll(...) [▶ 51]
- To enable exactly one group execute the method FB_TempCtrl.Groups[...].Enable(...) [▶ 56]

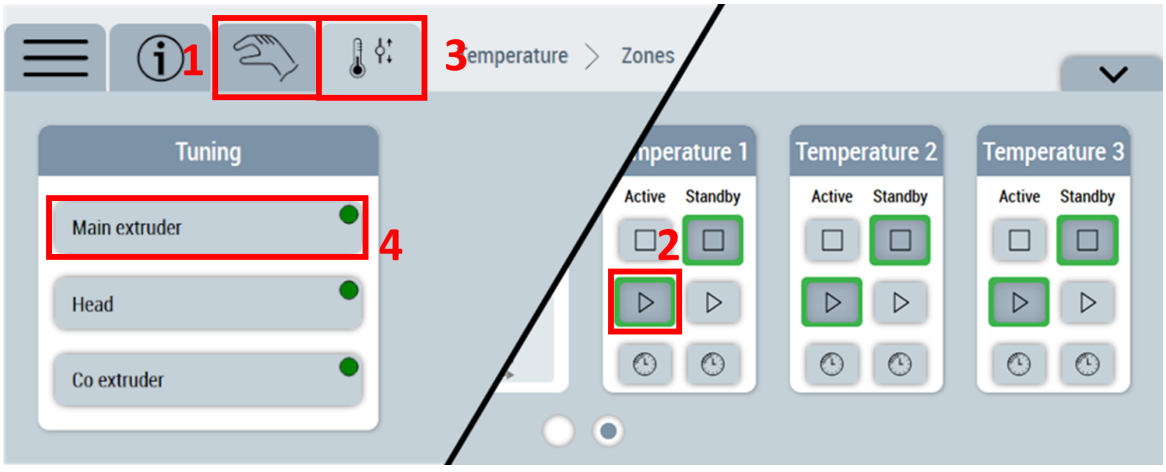
1. Switch on an individual temperature zone via the slider (1).
2. Check if the power level (2) increases and the actual value (3) of the activated zone changes.
3. Switch the zone off again as soon as possible to keep the temperature rise to a minimum.
4. Repeat steps 1 to 3 for each zone.



Start the automatic tuning of the control parameters

You should still be on the *Temperature* > *Parameters* page

1. Select the third tab (1) in the slider area.
2. Start the temperature control of a group (2) to be tuned.
3. Switch to the fourth tab (3) and start tuning via the corresponding button (4) of the group.
4. Carry out steps 1 to 3 for all groups to be commissioned.



Monitor the automatic tuning until it completes successfully

1. As soon as the display element on the button is no longer yellow, the tuning is finished.
2. If the display element on the button is green, the tuning has been completed successfully.

You have successfully commissioned your temperature control.

8.2 Creating and using the ZoneImageLayoutConfig server symbol

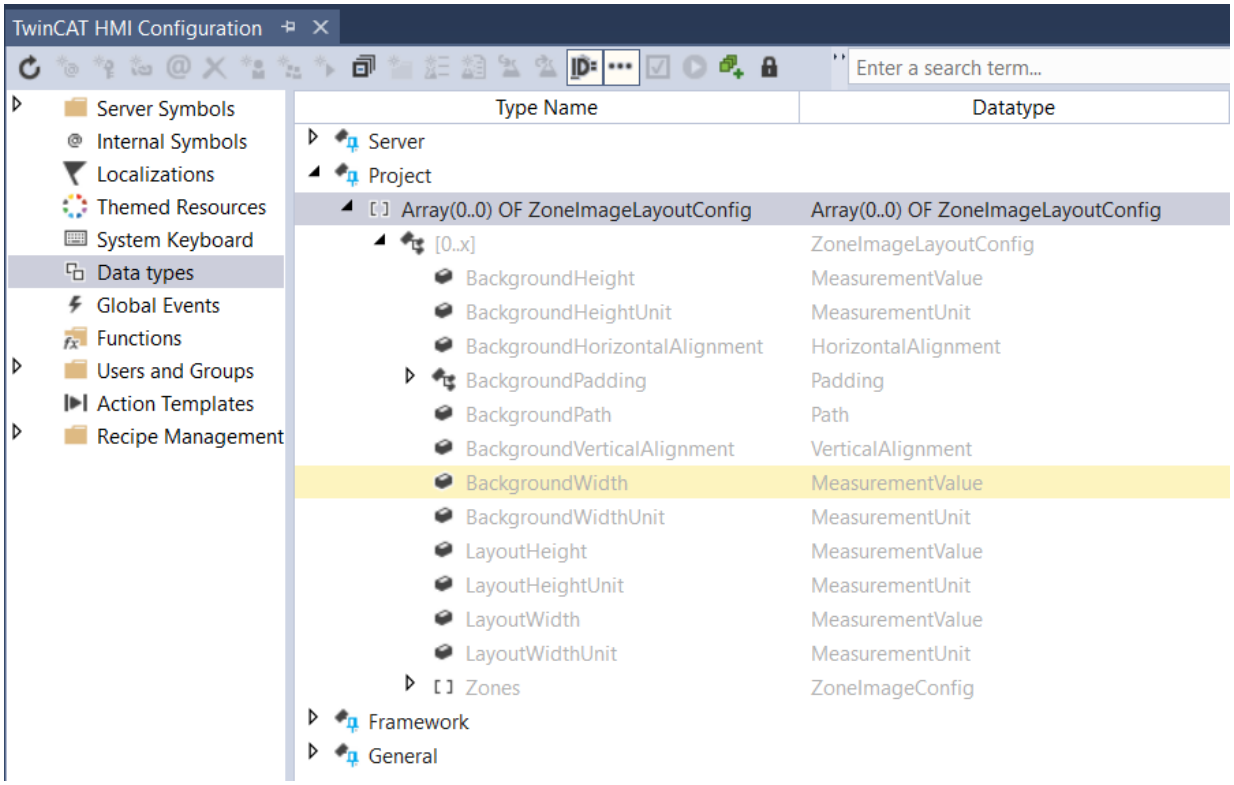
For storage and reuse [▶ 175] of layouts created on the Layout [▶ 183] page of the temperatures, one instance of the ZoneImageLayoutConfig server symbol is required per layout.

Creation of the server symbol

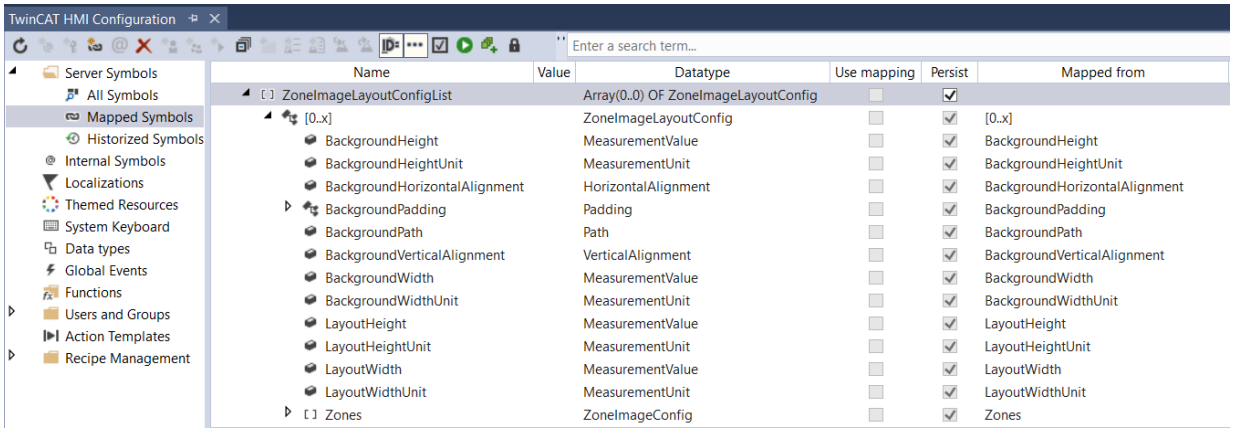
For each required layout a single position in the `ZoneImageLayoutConfigList` array is needed

For this purpose it is recommended to create a dynamic array of this type. This is done in the TwinCAT HMI configuration window:

1. Creation of the data type



2. Creation of a server symbol under the category `TcHmiSrv` of the corresponding data type

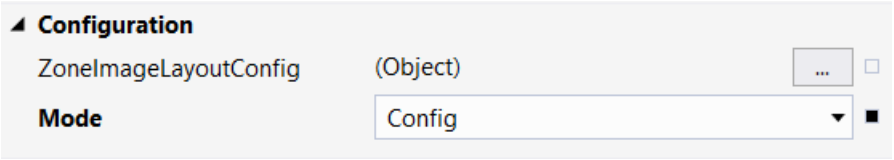


3. Setting the server symbol as persistent using the checkbox (`Persist`).

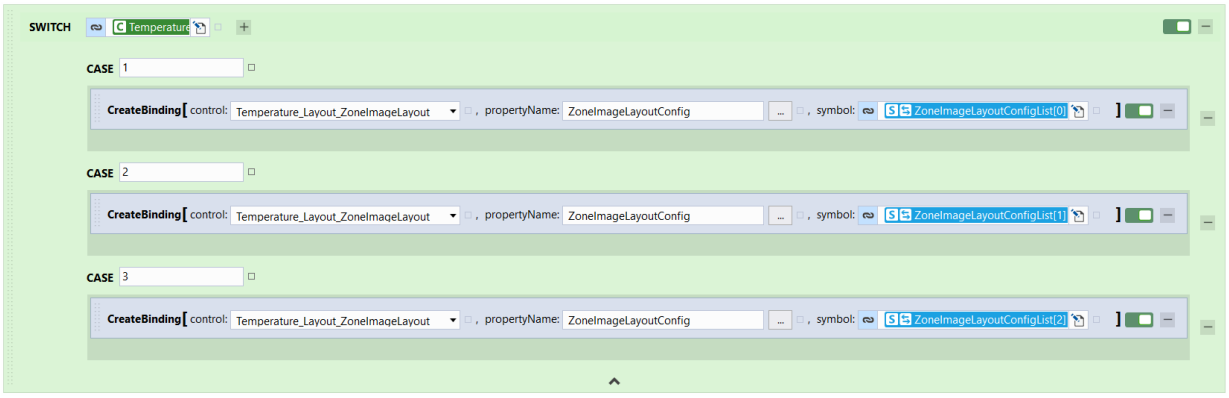
⇨ Server symbol is created.

Using the server symbol in the configuration mode of the ZoneImageLayout control

1. Select ZoneImageLayout control in the Toolbox and drag it to a content.
2. Set `Mode` parameter under the category `Configuration` to `Config`.



3. For example, using a TcHmiCombobox control to configure the number of layouts available in LiveView via the `.onSelectionChanged` event.

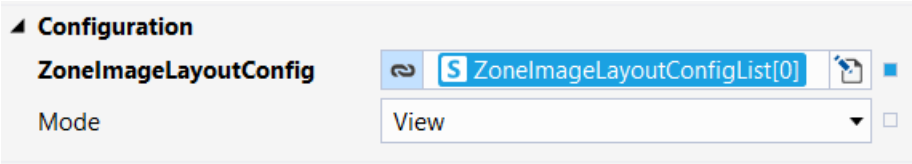


4. For this purpose, a new `CASE` can be created with the value 4 for the `SelectedId` parameter of the combo box.
5. Copy and paste the `CreateBinding` function into the new `CASE` and select a different array location such as `ZoneImageLayoutConfigList[3]` for the `symbol` transfer parameter.

⇨ Server symbol is used correctly in configuration mode.

Using the server symbol in display mode

1. Select ZoneImageLayout control in the Toolbox and drag it to a content.
2. Under the `Configuration` category, associate the `ZoneImageLayoutConfig` parameter with one of the array locations of the `ZoneImageLayoutConfigList` server symbol and set the `Mode` parameter to `View`.



3. The set width and height of the ZoneImageLayout control must be set in the server symbol (`LayoutHeight`, `LayoutHeightUnit`, `LayoutWidth` and `LayoutWidthUnit`). To do this, right-click on the server symbol in the TwinCAT HMI Configuration window and adjust the default value.

⇨ Server symbol is used correctly in display mode.

8.3 PLC-API (obsolete)

This chapter lists the obsolete PLC elements. These are still available for compatibility purposes, a change to the replacing elements is strongly recommended!

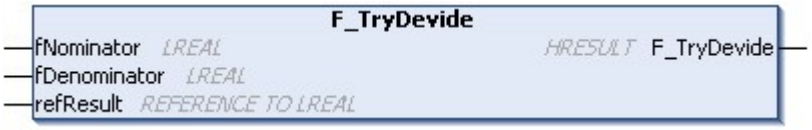
8.3.1 F_TryDevide()

Function is 'obsolete'

This function is marked as `obsolete` and should not be used anymore!

Alternative: F_TryDivide [▶ 114]

Original version:



Divides two values without throwing an exception.

Return value mathematically invalid

The function defines the mathematically invalid case x / 0 as 0. This is a mathematically invalid result, but is sufficient for many use cases. Check for your use case whether this definition does not lead to unforeseen misbehavior.

Syntax:

```
FUNCTION F_TryDevide : HRESULT
VAR_INPUT
    fNominator:      LREAL;
    fDenominator:    LREAL;
    refResult:       REFERENCE TO LREAL;
END_VAR
```

Inputs

Name	Type	Description
fNominator	LREAL	Value to be divided
fDenominator	LREAL	Value by which to divide
refResult	REFERENCE TO LREAL	Result of the division

Outputs

Name	Type	Description
F_TryDevide	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.2 FB_TrafoTableGenerator

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: TF8560 - Tc3_PlasticFunctions.FB_TrafoTableGenerator

Original version:

Represents the base class for generation algorithms of transformation cam plates.

Class is abstract

Since the class is defined as `ABSTRACT`, the class cannot be instantiated and must be implemented using inheritance.

The following pre-implemented geometries are included:

1. FB_TableGeneratorClampStandard [▶ 203] - Geometry of a typical clamping unit
2. FB_TableGeneratorCrankStandard [▶ 205] - Geometry of a crank mechanism
3. FB_TableGeneratorScotchYoke [▶ 206] - Geometry of a scotch yoke

Syntax:

```
FUNCTION_BLOCK ABSTRACT FB_TrafoTableGenerator
```

Properties

Name	Type	Access	Initial value	Description
DriveHighEnd	LREAL	Get, Set	0.0	Upper limit of the drive position
DriveLowEnd	LREAL	Get, Set	0.0	Lower limit of the drive position
LookUp	I_CammingLookUp	Get, Set	NULL	Interface to the TF8560 to be assigned `FB_CammingLookUp` Table
ParameterList	I_Parameter	Get	-	List of geometry-specific parameters
ParamValid	BOOL	Get	FALSE	The set parameters have valid values.
Scope [▶ 202]	I_TrafoScope	Get	-	Diagnostic values from the transformation table calculation

Methods

Name	Description
CalculateScope()	Calculates the diagnostic values of the Scope property.
DefineTable()	Starts the calculation of the transformation table.
ReadFromParamList()	Reads parameters from the list of special parameters.
WriteToParamList()	Writes local variables to the list of special parameters.

Methods are abstract

The methods are defined as ABSTRACT and must be implemented in inheriting classes.

Interfaces

Type	Description
I_TrafoTableGenerator	Standard interface on FB_TrafoTableGenerator

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.2.1 FB_TrafoScope

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: - (completely removed)

Original version:

Implements diagnostic values for the calculation of the transformation table

Syntax:

```
FUNCTION_BLOCK FB_TrafoScope
```

Properties

Name	Type	Access	Initial value	Description
HighLimitDefined	BOOL	Get, Set	FALSE	An upper limit is defined.
HighLimitDrive	LREAL	Get, Set	0.0	Upper limit of the drive
HighLimitLoad	LREAL	Get, Set	0.0	Upper limit of the load side
HighSideBlockpoint	BOOL	Get, Set	FALSE	An upper blocking of the mechanics exists.
HighSideTurnpoint	BOOL	Get, Set	FALSE	An upper turning point of the mechanics exists.
LowLimitDefined	BOOL	Get, Set	FALSE	A lower limit is defined.
LowLimitDrive	LREAL	Get, Set	0.0	Lower limit of the drive
LowLimitLoad	LREAL	Get, Set	0.0	Lower limit of the load side
LowSideBlockpoint	BOOL	Get, Set	FALSE	A lower blocking of the mechanics exists.
LowSideTurnpoint	BOOL	Get, Set	FALSE	A lower turning point of the mechanics exists.

Interfaces

Type	Description
I_TrafoScope	Standard interface on FB_TrafoScope

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.2.2 FB_TableGeneratorClampStandard_1

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

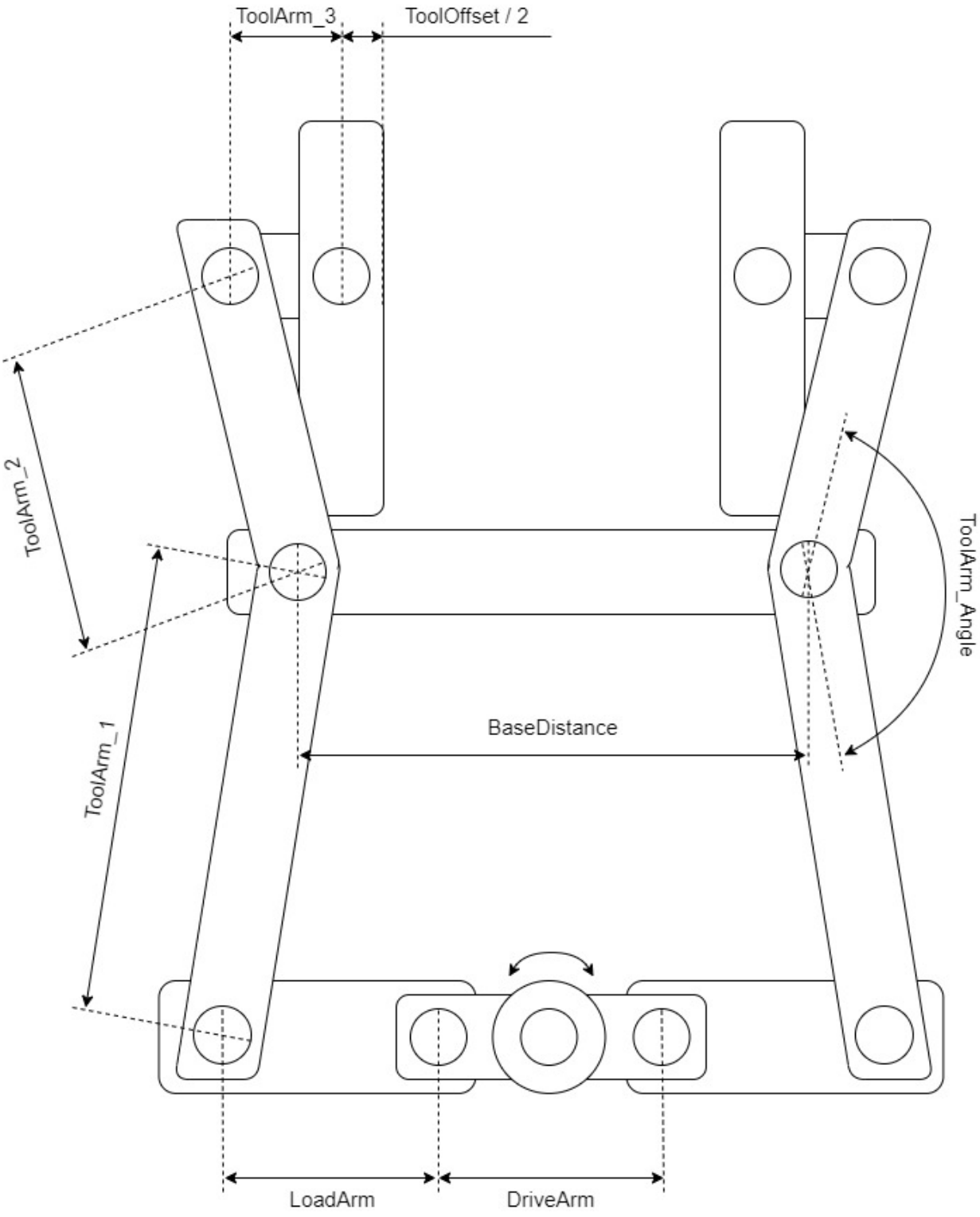
Alternative: TF8560 - Tc3_PlasticFunctions.FB_ClampTableGenerator

Original version:

Implements a calculation algorithm for a standard clamping unit mechanism.

Names of the dimension designations in the sketch

The dimension designations are the same as the names of the properties of the class to be configured.



Syntax:

```
FUNCTION_BLOCK FB_TableGeneratorClampStandard_1 EXTENDS FB_TrafoTableGenerator
```

Properties

Name	Type	Access	Initial value	Description
BaseDistance	LREAL	Get, Set	0.0	Distance between the two stationary bearing points in the center of the mechanism
DriveArm	LREAL	Get, Set	0.0	Lever arm attached to the drive
DriveStep	LREAL	Get,	0.0	Resulting resolution of the drive position in the transformation table
LoadArm	LREAL	Get, Set	0.0	Transfer arm to the tool arms
ToolArm_1	LREAL	Get, Set	0.0	Lower part of the lateral tool arm
ToolArm_2	LREAL	Get, Set	0.0	Upper part of the lateral tool arm
ToolArm_3	LREAL	Get, Set	0.0	Horizontal tool arm
ToolArm_Angle	LREAL	Get, Set	0.0	Angle between the two parts of the side tool arm
ToolOffset	LREAL	Get, Set	0.0	Total offset between the bearing and the tool clamping surface In the recommended ToolOffset design, the transformation results in the distance between the tool clamping surfaces. To use the distance of the tool opening, it is recommended to use the ToolAdaption. This is configurable in the machine data of each FB_PtpMotion based axis.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.2.3 FB_TableGeneratorCrankStandard

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: TF8560 - Tc3_PlasticFunctions.FB_CrankTableGenerator

Original version:

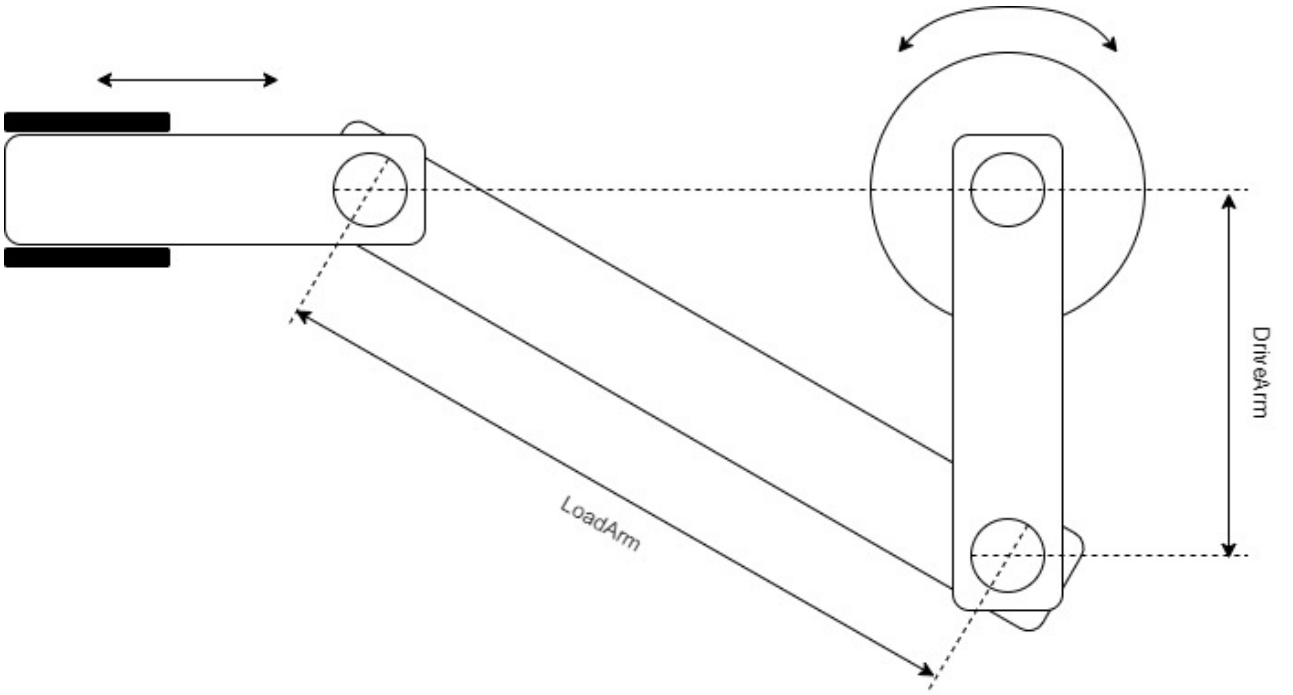
Implements a calculation algorithm for a crank mechanism.

Constructive assumption

The calculation assumes that the motion axis of the load side is in alignment with the center of rotation of the drive side. This is indicated in the sketch by the horizontal dashed line!

Names of the dimension designations in the sketch

The dimension designations are the same as the names of the properties of the class to be configured.



Syntax:

```
FUNCTION_BLOCK FB_TableGeneratorCrankStandard EXTENDS FB_TrafoTableGenerator
```

Properties

Name	Type	Access	Initial value	Description
DriveArm	LREAL	Get, Set	0.0	Lever arm attached to the drive
DriveStep	LREAL	Get	0.0	Resulting resolution of the drive position in the transformation table
LoadArm	LREAL	Get, Set	0.0	Transfer arm to the guided load side

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.2.4 FB_TableGeneratorScotchYoke

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

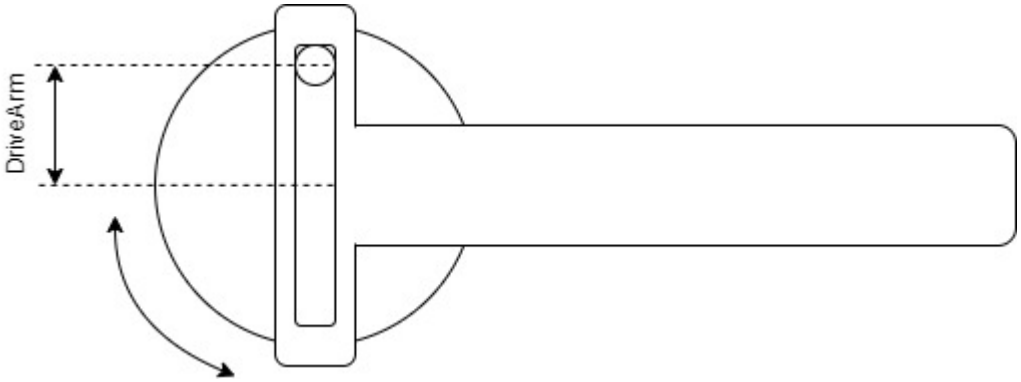
Alternative: TF8560 - Tc3_PlasticFunctions.FB_ScotchYokeTableGenerator

Original version:

Implements a calculation algorithm for a scotch yoke mechanism.

Names of the dimension designations in the sketch

The dimension designations are the same as the names of the properties of the class to be configured.



Syntax:

```
FUNCTION_BLOCK FB_TableGeneratorScotchYoke EXTENDS FB_TrafoTableGenerator
```

Properties

Name	Type	Access	Initial value	Description
DriveArm	LREAL	Get, Set	0.0	Radius of the eccentric rotation
DriveStep	LREAL	Get	0.0	Resulting resolution of the drive position in the transformation table

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.3 FB_MonitoringZone

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_MonitoringTemp [▶ 83]

Original version:

Implements monitoring of a temperature zone with direct connection to TF8540. The adjustable limits are synchronized with the temperature zone.

Syntax:

```
FUNCTION_BLOCK FB_MonitoringZone EXTENDS FB_Monitoring
```

Properties

Name	Type	Access	Initial value	Description
TempZoneHmi	I_TempZoneHmi	Get	NULL	[INTERNAL] Referencing to the values of the assigned zone

Methods

Name	Description
SetTempZone()	Sets the temperature zone to be monitored

Interfaces

Type	Description
I_MonitoringZone	Standard interface on FB_MonitoringZone

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4 FB_TempCtrl

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_Temperature [▶ 49]

Original version:

Main class of temperature control to manage all temperature zones and groups.

Syntax:

```
FUNCTION_BLOCK FB_TempCtrl EXTENDS FB_BaseMd
```

Properties

Name	Type	Access	Initial value	Description
DisableAlarms	BOOL	Get, Set	FALSE	Suppresses alarms triggered by errors in a temperature zone.
DisableMessages	BOOL	Get, Set	FALSE	Suppresses debug messages of the TF8540 library.
EnableCallback	BOOL	Get, Set	TRUE	Enables communication with the I/O components.
EnableConfig	BOOL	Get, Set	TRUE	Enables the configuration of all zones.
EnableLooptest	BOOL	Get, Set	FALSE	Enables current monitoring of all zones.
Groups	REFERENCE TO ARRAY[] OF FB_TempGroup	Get	-	Control of the individual groups.
LibScopeVars	REFERENCE TO FB_Scope_TempCtrlVariables	Get	-	Access to an overview of TF8540 live data.
RefMdTempSupply	REFERENCE TO ARRAY [] OF FB_MdTempSupply	Get	-	Access to the array of machine data containers of all supply units.
RefMdTempZone	REFERENCE TO ARRAY[] OF FB_MdTempZone	Get	-	Access to the array of machine data containers of all temperature zones.
Timer	I_TempSchedule	Get	NULL	Access to the connected scheduler.

Methods

Name	Description
CreateDefaultParams() [▶ 210]	Creates a default parameterization for all temperature zones.
EnableAll() [▶ 211]	Enables all temperature zones on the PLC side.
LinkGroup() [▶ 211]	Assigns a linear arrangement of zones to a group.
LinkSupply() [▶ 212]	Assigns a group to a supply unit.
LinkZone() [▶ 213]	Assigns a zone to a group.
SetOpMode() [▶ 213]	Configures the current operation mode.
SetScheduler()	Assigns a schedule to the temperature control.
StandbyAll() [▶ 214]	Sets all zones to standby.
UnlinkGroup()	Removes all links to a group.

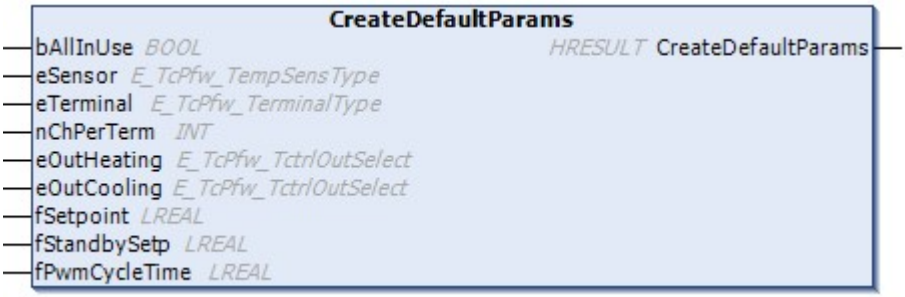
Interfaces

Type	Description
I_TempCtrl	Standard interface on FB_TempCtrl
I_TempCtrlMdRef	Interface for the transfer of zone machine data
I_TempTaskInterface	Runtime interface for a slow PLC task

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.1 CreateDefaultParams()



Creates a default parameterization for all temperature zones.

Syntax:

```
METHOD CreateDefaultParams : HRESULT
VAR_INPUT
    bAllInUse:              BOOL;
    eSensor:                E_TcPfw_TempSensType;
    eTerminal:              E_TcPfw_TerminalType;
    nChPerTerm:             INT;
    eOutHeating:            E_TcPfw_TctrlOutSelect;
    eOutCooling:            E_TcPfw_TctrlOutSelect;
    fSetpoint:              LREAL;
    fStandbySetp:           LREAL;
    fPwmCycleTime:          LREAL;
END_VAR
```

Inputs

Name	Type	Description	Recommended standard
bAllInUse	BOOL	All zones are initialized as "InUse".	TRUE
eSensor	E_TcPfw_TempSensType	Sensor type - NoSensor in simulation mode	eTcPfwTempSensT_NoSensor
eTerminal	E_TcPfwTerminalType	Terminal type - NoTerminal in simulation mode	eTcPfwTermT_NoTerminal
nChPerTerm	INT	Number of channels per terminal	8
eOutHeating	E_TcPfw_TctrlOutSelect	Output type of the heating output - NoSignal, to disable the heating function (e.g. for measuring zones)	eTcPfwTcOut_PWM
eOutCooling	E_TcPfw_TctrlOutSelect	Cooling output type	eTcPfwTcOut_NoSignal
fSetpoint	LREAL	Temperature setpoint for all zones	180.0
fStandbySetp	LREAL	Temperature setpoint for standby temperature of all zones	18.0
fPwmCycleTime	LREAL	PWM cycle time for all outputs (dutyCycle = fPwmCycleTime * 0.1)	1.0

Outputs

Name	Type	Description
CreateDefaultParams	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.2 EnableAll()

Enables all temperature zones on the PLC side.

Syntax:

```
METHOD EnableAll
VAR_INPUT
    bCmd:       BOOL;
    bGroups:    BOOL;
END_VAR
```

Inputs

Name	Type	Description
bCmd	BOOL	TRUE to grant the release, FALSE to withdraw the release.
bGroups	BOOL	The enable only takes into account zones that are assigned to a group.

Outputs

Name	Type	Description
EnableAll	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.3 LinkGroup()

Assigns a set of zones to a group.

Syntax:

```
METHOD LinkGroup : HRESULT
VAR_INPUT
    nStartIdx:          INT;
    nEndIdx:            INT;
    nGroupIdx:          INT;
    bOverwrite:         BOOL;
END_VAR
```

Inputs

Name	Type	Description
nStartIdx	INT	Index of the first zone to be assigned from the linear TF8540 library arrangement
nEndIdx	INT	Index of the last zone to be assigned from the linear TF8540 library arrangement
nGroupIdx	INT	Index of the group to which the zones are to be assigned
bOverwrite	BOOL	Zones are assigned even if the group contains already assigned zones.

Outputs

Name	Type	Description
LinkGroup	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.4 LinkSupply()

Assigns a supply unit to a group.

Syntax:

```
METHOD LinkSupply : HRESULT
VAR_INPUT
    nGroupIdx:          INT;
    nSupplyIdx:         INT;
END_VAR
```

Inputs

Name	Type	Description
nGroupIdx	INT	Index of the group to which a supply unit is to be assigned
nSupplyIdx	INT	Index of the supply unit to be assigned to the group

Outputs

Name	Type	Description
LinkSupply	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.5 LinkZone()

Assigns a single zone to a group.

Syntax:

```
METHOD LinkZone : HRESULT
VAR_INPUT
    nLinearIdx:         INT;
    nGroupIdx:          INT;
    nGroupMemberIdx:    INT;
    bOverwrite:         BOOL;
END_VAR
```

Inputs

Name	Type	Description
nLinearIdx	INT	Index of the zone from the linear TF8540 library arrangement that is to be assigned
nGroupIdx	INT	Index of the group to which the zone should be assigned
nGroupMemberIdx	INT	Index in the target group
bOverwrite	BOOL	Zone is assigned even if the index is already occupied in the target group.

Outputs

Name	Type	Description
LinkZone	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.6 SetOpMode()

Defines the current OpMode of the temperature control.

- OpMode None (0)
  - Temperature control does not accept commands
- OpMode Simple (1)

- Zones can be enabled and disabled via TmpCtrlHmi.Group[].Enable

- OpMode Standard (2)
  - Zones are controlled via TempCtrlHmi.Groups[].OpModeActive
  - Zones support standby via TempCtrlHmi.Groups[].OpModeStandby

Syntax:

```
METHOD SetOpMode : HRESULT
VAR_INPUT
    OpMode:             INT;
END_VAR
```

Inputs

Name	Type	Description
OpMode	INT	Selection parameters: 0 – None, 1 – Simple, 2 - Standard

Outputs

Name	Type	Description
SetOpMode	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.4.7 StandbyAll()

Sets all zones to standby temperature.

Syntax:

```
METHOD StandbyAll : HRESULT
VAR_INPUT
    bCmd:       BOOL;
    bGroups:    BOOL;
END_VAR
```

Inputs

Name	Type	Description
bCmd	BOOL	TRUE to enable the standby temperature, FALSE to disable.
bGroups	BOOL	The function only considers zones that are assigned to a group.

Outputs

Name	Type	Description
StandbyAll	HRESULT	Return value with feedback on the success of the execution

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

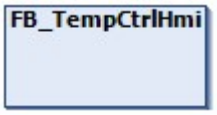
8.3.5 FB_TempCtrlHmi

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_TemperatureHmi [▶ 55]

Original version:



HMI parallel class to the `FB_TempCtrl`.

Syntax:

```
FUNCTION_BLOCK FB_TempCtrlHmi EXTENDS FB_BaseMdHmi
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	A zone (InUse = TRUE) has exceeded the absolute temperature maximum.
AlarmAbsoluteLow	BOOL	Get	FALSE	One zone (InUse = TRUE) has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	At least one zone with active control has exceeded the HighHigh tolerance.
AlarmHigh	BOOL	Get	FALSE	At least one zone with active control has exceeded the High tolerance.
AlarmLow	BOOL	Get	FALSE	At least one zone with active control has fallen below the low tolerance.
AlarmLowLow	BOOL	Get	FALSE	At least one zone with active control has fallen below the LowLow tolerance.
CountPfwZones	LREAL	Get	20.0	Number of available TF8540 temperature zones
Groups	REFERENCE TO ARRAY[] OF FB_TempGroupHmi	Get	-	Access to group-based information
ParamTempSupply	REFERENCE TO ARRAY[] OF FB_MdTempSupplyHmi	Get	-	Parameter interface for parameterization via the HMI
ParamTempZone	REFERENCE TO ARRAY[] OF FB_MdTempZoneHmi	Get	-	Parameter interface for parameterization via the HMI
TempAmbient	LREAL	Get, Set	18.0	Standard ambient temperature (for simulation)

Interfaces

Type	Description
I_TempCtrlHmi	Standard interface on FB_TempCtrlHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.6 FB_TempGroup

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_TemperatureGroup [▶ 56]

Original version:

Class for group control of temperature control.

Syntax:

```
FUNCTION_BLOCK FB_TempGroup EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	Group (at least one zone) has exceeded the absolute maximum temperature.
AlarmAbsoluteLow	BOOL	Get	FALSE	Group (at least one zone) has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	Group (at least one zone) has exceeded the HighHigh tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmHigh	BOOL	Get	FALSE	Group (at least one zone) has exceeded the High tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLow	BOOL	Get	FALSE	Group (at least one zone) has exceeded the Low tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmLowLow	BOOL	Get	FALSE	Group (at least one zone) has exceeded the LowLow tolerance. Alarms to the tolerance values are only active if the zones are actively controlled.
AlarmNoResponse	BOOL	Get	FALSE	Temperature value of the group (at least one zone) does not respond to the control.
EnableLimitAlarms	BOOL	Get, Set	FALSE	Exceeding a tolerance value/ limit triggers an alarm.
Fault	BOOL	Get	FALSE	Group (at least one zone) has an error.
IsActive	BOOL	Get	FALSE	Group (at least one zone) is actively controlled.
IsEnabled	BOOL	Get	FALSE	All zones of the group are enabled.
IsStandby	BOOL	Get	FALSE	All zones of the group are in standby mode.
Zones	REFERENCE TO ARRAY[] OF FB_TempZone	Get	-	Control of the individual zones

Methods

Name	Description
Enable()	Enables all zones of the group on the PLC side.
Force()	Forces all zones of the group to heating/cooling 100%.

Interfaces

Type	Description
I_TempGroup	Standard interface on FB_TempGroup

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.7 FB_TempGroupHmi

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_TemperatureGroupHmi [▶ 60]

Original version:

HMI parallel class to the FB_TempGroup class.

Syntax:

```
FUNCTION_BLOCK FB_TempGroupHmi EXTENDS FB_BaseMdHmi
```

Properties

Name	Type	Access	Initial value	Description
CountLinkedZones	INT	Get, Set	0	Number of assigned zones via the 'Link()' methods of the FB_TempCtrl
DoTune	BOOL	Get, Set	FALSE	Starts tuning of all active zones of the group.
Enable[1]	BOOL	Get, Set	FALSE	Releases the zones through the HMI.
GroupIndex	INT	Get	0	Index of the group in the FB_TempCtrlHmi array
GroupName	STRING	Get	‘‘	Temperature group name
OpModeActive[2]	REFERENCE TO FB_TempGroupOpModeHmi	Get	-	Interface for active switching of a temperature group
OpModeStandby[2]	REFERENCE TO FB_TempGroupOpModeHmi	Get	-	Interface for the standby circuit of the temperature group
TuningActive	BOOL	Get	FALSE	Tuning of at least one zone is active.
TuningDone	BOOL	Get	FALSE	The tuning of the group is completed.
Zones	REFERENCE TO ARRAY[] OF FB_TempZoneHmi	Get	-	Interface to the individual zones of a group

[1] Only in OpMode 'Simple'

[2] Only in OpMode 'Standard'

Methods

Name	Description
SetZoneData()	[INTERNAL] Connects the temperature zones of a group with the machine data.

Interfaces

Type	Description
I_TempGroupHmi	Standard interface on FB_TempGroupHmi

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.8 FB_TempGroupOpModeHmi

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: - (integrated in FB_TemperatureGroupHmi [▶ 60])

Original version:

Class for controlling the temperature operation mode (in FB_TempCtrl OpMode "Standard").

Syntax:

```
FUNCTION_BLOCK FB_TempGroupOpModeHmi
```

Properties

Name	Type	Access	Initial value	Description
Off	BOOL	Get, Set	TRUE	Switches the temperature group off.
On	BOOL	Get, Set	FALSE	Switches the temperature group on.
Timed	BOOL	Get, Set	FALSE	Switches the temperature group to a time-based switch-on.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.9 FB_TempZone

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_TempChannel [▶ 64]

Original version:

Class for single control of a temperature zone.

Syntax:

```
FUNCTION_BLOCK FB_TempZone EXTENDS FB_Base
```

Properties

Name	Type	Access	Initial value	Description
AlarmAbsoluteHigh	BOOL	Get	FALSE	Zone has exceeded the absolute maximum temperature.
AlarmAbsoluteLow	BOOL	Get	FALSE	Zone has fallen below the absolute temperature minimum.
AlarmHighHigh	BOOL	Get	FALSE	Zone has exceeded the HighHigh tolerance. Alarms to the tolerance values are only active when the zone is actively controlled.
AlarmHigh	BOOL	Get	FALSE	Zone has exceeded the high tolerance. Alarms to the tolerance values are only active when the zone is actively controlled.
AlarmLow	BOOL	Get	FALSE	Zone has exceeded the low tolerance. Alarms to the tolerance values are only active when the zone is actively controlled.
AlarmLowLow	BOOL	Get	FALSE	Zone has exceeded the LowLow tolerance. Alarms to the tolerance values are only active when the zone is actively controlled.
AlarmNoResponse	BOOL	Get	FALSE	Temperature value of the zone does not respond to the control.
EnableLimitAlarms	BOOL	Get, Set	FALSE	Exceeding a tolerance value/ limit triggers an alarm.
IsLinked	BOOL	Get	FALSE	Zone is connected to a group.

Interfaces

Type	Description
I_TempZone	Standard interface on FB_TempZone

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

8.3.10 FB_TempZoneHmi

Class is 'obsolete'

This class is marked as `obsolete` and should not be used anymore!

Alternative: FB_TempChannelBase [▶ 63]

Original version:

Access class for a single temperature zone via the HMI.

Syntax:

```
FUNCTION_BLOCK FB_TempZoneHmi
```

Properties

Name	Type	Access	Initial value	Description
Index	INT	Get, Set	0	References a PlasticApplication zone to a TF8540 zone.

Class contains significantly more properties than listed

The properties of the FB_TempZoneHmi class overlap with the TF8540 Global Variables aaaPfwTempToHmi, aaaPfwTempMparamFromHmi and aaaPfwTempPparamFromHmi. For more information on overlapping features, it is recommended to use the TF8540 documentation.

Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT v3.1.4024.35	PC or CX (x64, x86)	Tc3_PlasticBaseApplication (>= v3.12.5.0)

More Information:
www.beckhoff.com/en-us/products/automation/twincat/tfxxxx-twincat-3-functions/tf8xxx-industry-specific

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