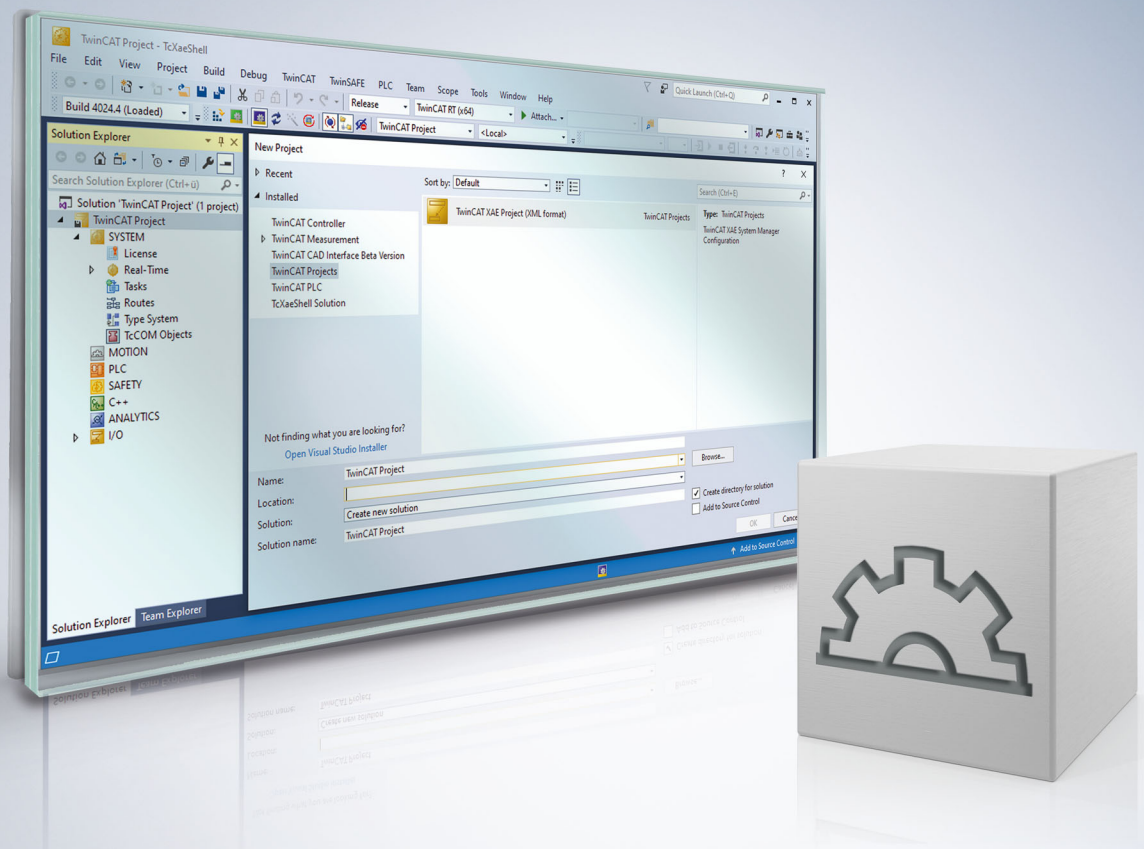
BECKHOFF New Automation Technology

Manual | EN

TE1000

TwinCAT 3 | PLC Library: Tc3_PackML_V3



2025-08-25 | Version: 1.0.0

Table of contents

1 Foreword 5
1.1 Notes on the documentation 5
1.2 For your safety 5
1.3 Notes on information security 7
2 Packaging Machine State 8
3 Packaging Machine Tags 10
3.1 Tag Types 10
3.2 Tag Details 11
4 Function Blocks 19
4.1 Conversion 19
4.1.1 Time 19
4.1.2 Timestamp 20
4.1.3 F_PMLStateCommandToString 22
4.1.4 F_PMLUnitModeToString 22
4.2 General 22
4.2.1 FB_PMLAdminAlarm 23
4.2.2 FB_PMLAdminTime 33
4.3 Packaging Machine State 35
4.3.1 FB_PMLStateMachine 35
4.3.2 FB_PMLUnitModeConfig 38
4.3.3 FB_PMLUnitModeManager 41
5 Data types 44
5.1 General 44
5.1.1 ST_PMLAdminTimeOptions 44
5.2 Packaging machine state and mode 44
5.2.1 E_PMLCommand 44
5.2.2 E_PMLProtectedUnitMode 44
5.2.3 E_PMLState 46
5.2.4 ST_PMLStateMachineOptions 46
5.2.5 ST_PMLSubUnitInfo 46
5.2.6 ST_PMLSubUnitInfoRef 47
5.2.7 ST_PMLUnitModeConfiguration 47
5.3 PackTags 47
5.3.1 Alarm 48
5.3.2 Common 49
5.3.3 ST_PMLa 52
5.3.4 ST_PMLaMin 53
5.3.5 ST_PMLc 53
5.3.6 ST_PMLcMin 54
5.3.7 ST_PMLs 54
5.3.8 ST_PMLsMin 55
5.3.9 ST_PMLV2022 55
5.3.10 ST_PMLV2022Min 55

6 Global parameters 56

7 Global constants / variables 57

8 Interfaces 58

8.1 I_PMLUnitStateActing 58

8.2 I_PMLUnitStateWaiting 58

9 Support and Service 59

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.
The documentation and the following notes and explanations must be complied with when installing and commissioning the components.
The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been compiled 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.
Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

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

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

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.
The distribution and reproduction of this document, as well as the use and communication of its contents without express authorization, are prohibited.
Offenders will be held liable for the payment of damages. All rights reserved in the event that a patent, utility model, or design are registered.

Third-party trademarks

Trademarks of third parties may be used in this documentation. You can find the trademark notices here: https://www.beckhoff.com/trademarks.

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.

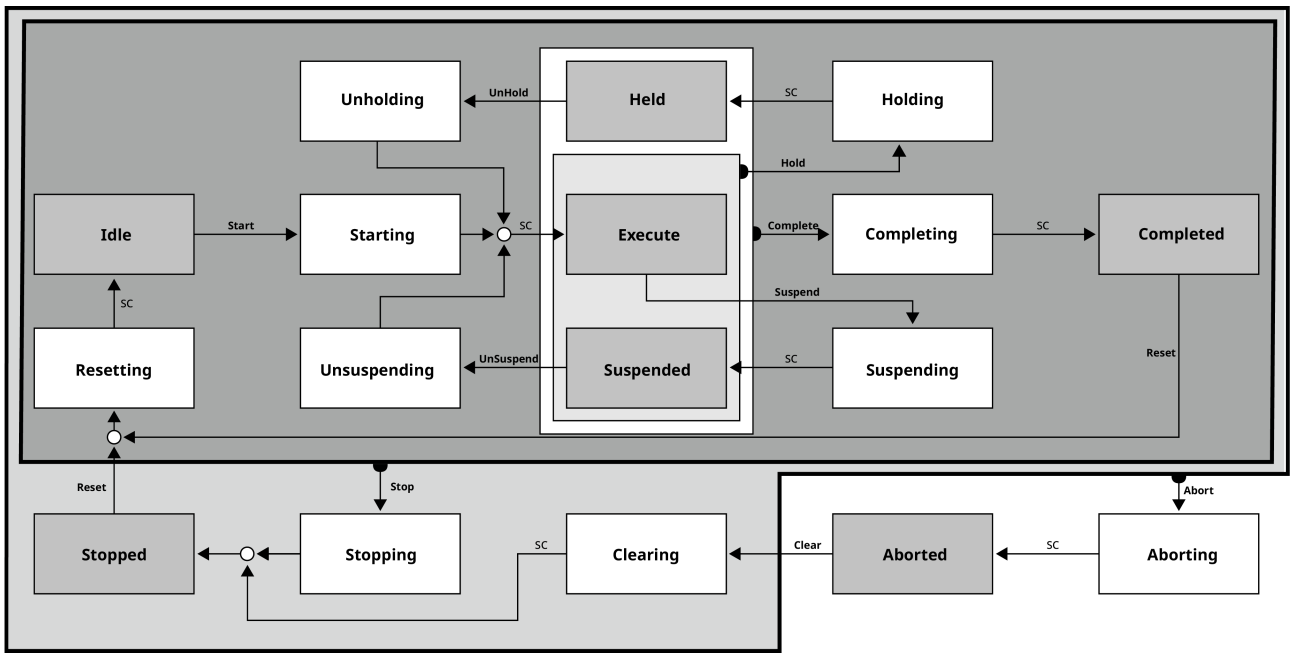
To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Packaging Machine State

The “Packaging Machine State” function blocks have a common interface to the existing “PackML Machine State Model” versions.

It is expected that application-specific logic such as state transitions are programmed in external function blocks and that the “Packaging Machine State” function block takes over the central logic of the state machine and the state display. For that reason there is a recommendation for this function block with regard to how it can be combined with other logic.

The state transition in a machine application is always application-specific. To simplify standardization, it is best to implement methods linked to the PackML State Machine V3 in your function blocks. To do this, you can use the interfaces provided within this library and then program the methods for the specific application. The “acting” methods acquire application-specific signals and represent the transition logic to adjacent states (see PackML state model below). The "Waiting" methods provide feedback to the higher-level, higher-order PackML State Machine V3, enabling a standard state machine and state message. The methods contain the machine execution code and the application-specific transition logic.



The available methods are listed below and are programmed in a way that preserves the integrity and functionality of the PackML state machine.

Names of the "PackML State Machine V3" Acting methods:

- M_Starting
- M_Completing
- M_Resetting
- M_Holding
- M_UnHolding
- M_Suspending
- M_UnSuspending
- M_Clearing
- M_Stopping
- M_Aborting
- M_Execute

Names of the "PackML State Machine V3" Waiting methods:

- M_Completed
- M_Idle
- M_Held
- M_Suspended
- M_Stopped
- M_Aborted

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

3 Packaging Machine Tags

PackTags provides a uniform set of naming conventions for data elements that are used in the procedural elements of the Base State Model. As described, the Base State Model provides a uniform set of machine states, so that all automated machines can be considered in the same way. PackTags are data elements provided with names for the interoperable data exchange between automated machines with open architectures. This documentation contains the key names of the data elements, data types, values, fields and data structures, if applicable. PackTags are used for machine-to-machine communication, e.g. between a bottle filler and a cap fitter. PackTags can also be used for the exchange of data between a machine and superordinated information systems such as Manufacturing Operations Management and Enterprise Information Systems.

The documentation describes all PackTags for the navigation through a state model and for the definition and actuation of the system control mode. Furthermore, this documentation defines a list of PackTags that provide potentially important information about a machine. All PackTags must be used in order to conform to the principles of integrated connectivity with systems with the same implementation.

The tags required are those that are needed for the function of the automated machine or for the connectivity to control or remote systems.

3.1 Tag Types

PackTags are broken down into three groups: Command, Status and Administration. Command and State tags contain data for interfacing the machine with the line control for coordination or for downloading recipes/ parameters. Command tags are transferred as “information recipients” to the machine program and “consumed” by it. State tags are created and read by the machine program. Administration tags contain data, which are collected by higher-level systems for machine performance analysis or operator information.

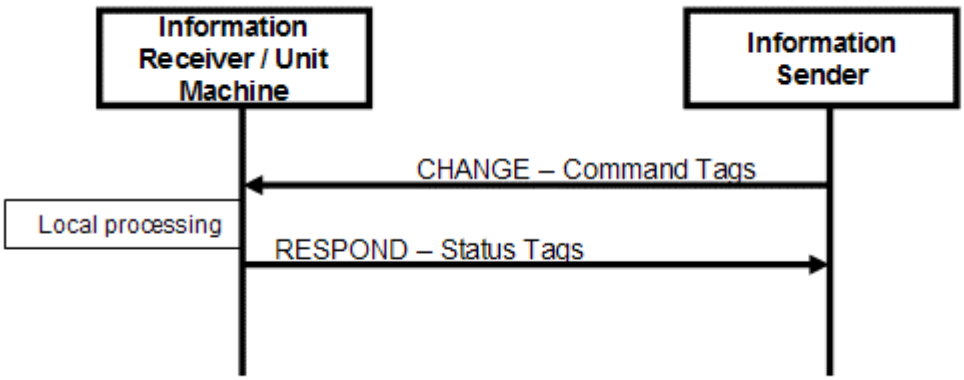
The grouping of data should take place in adjacent registers, in order to optimize the communication.

Information data are usually transferred via OPC in an Ethernet-based communication network.

The prefix of Command tags is “PMLc”.

The prefix of State tags is “PMLs”.

The prefix of Administration tags is “PMLa”.



3.2 Tag Details

The following section provides an overview of the tags. Command, state and administration PackTags are listed in the following tables.

Command structure Command

					Tag Name	Data Type
Command					Command	ST_PMLc
	UnitMode				Command.UnitMode	DINT
	UnitModeChangeRequest				Command.UnitModeChangeRequest	BOOL
	MachSpeed				Command.MachSpeed	REAL
	MaterialInterlock				Command.MaterialInterlock	DWORD
	CntrlCmd				Command.CntrlCmd	DINT
	CmdChangeRequest				Command.CmdChangeRequest	BOOL
	Parameter_REAL[#]				Command.Parameter_REAL[#]	ST_PMLParameterReal
		Id			Command.Parameter_REAL[#].Id	DINT
		Name			Command.Parameter_REAL[#].Name	STRING
		Unit			Command.Parameter_REAL[#].Unit	STRING(6)
		Value			Command.Parameter_REAL[#].Value	REAL
	Parameter_STRING[#]				Command.Parameter_STRING[#]	ST_PMLParameterString
		Id			Command.Parameter_STRING[#].Id	DINT
		Name			Command.Parameter_STRING[#].Name	STRING
		Unit			Command.Parameter_STRING[#].Unit	STRING(6)
		Value			Command.Parameter_STRING[#].Value	STRING
	Parameter_LREAL[#]				Command.Parameter_LREAL[#]	ST_PMLParameterLreal
		Id			Command.Parameter_LREAL[#].Id	DINT
		Name			Command.Parameter_LREAL[#].Name	STRING
		Unit			Command.Parameter_LREAL[#].Unit	STRING(6)
		Value			Command.Parameter_LREAL[#].Value	LREAL
	Parameter_DINT[#]				Command.Parameter_DINT[#]	ST_PMLParameterDint
		Id			Command.Parameter_DINT[#].Id	DINT
		Name			Command.Parameter_DINT[#].Name	STRING
		Unit			Command.Parameter_DINT[#].Unit	STRING(6)
		Value			Command.Parameter_DINT[#].Value	DINT
	SelectedRecipe				Command.SelectedRecipe	DINT
	RecipeChangeRequest				Command.RecipeChangeRequest	BOOL
	Recipe[#]				Command.Recipe[#]	ST_PMLRecipe
		Id			Command.Recipe[#].Id	DINT
		Name			Command.Recipe[#].Name	STRING
		Unit			Command.Recipe[#]..unit	STRING(6)
		PrimaryQty			Command.Recipe[#].PrimaryQty	REAL
		ProcessVariables			Command.Recipe[#].ProcessVariables	ST_PMLProcessVariables
			Parameter_REAL[#]		Command.Recipe[#].ProcessVariables.Parameter_REAL[#]	ST_PMLParameterReal
				Id	Command.Recipe[#].ProcessVariables.Parameter_REAL[#].Id	DINT
				Name	Command.Recipe[#].ProcessVariables.Parameter_REAL[#].Name	STRING
				Unit	Command.Recipe[#].ProcessVariables.Parameter_REAL[#].Unit	STRING(6)
				Value	Command.Recipe[#].ProcessVariables.Parameter_REAL[#].Value	REAL

			Parameter_STRING[#]		Command.Recipe[#].ProcessVariables.Parameter_STRING[#]	ST_PMLParameterString
				Id	Command.Recipe[#].ProcessVariables.Parameter_STRING[#].Id	DINT
				Name	Command.Recipe[#].ProcessVariables.Parameter_STRING[#].Name	STRING
				Unit	Command.Recipe[#].ProcessVariables.Parameter_STRING[#].Unit	STRING(6)
				Value	Command.Recipe[#].ProcessVariables.Parameter_STRING[#].Value	STRING
			Parameter_LREAL[#]		Command.Recipe[#].ProcessVariables.Parameter_LREAL[#]	ST_PMLParameterLreal
				Id	Command.Recipe[#].ProcessVariables.Parameter_LREAL[#].Id	DINT
				Name	Command.Recipe[#].ProcessVariables.Parameter_LREAL[#].Name	STRING
				Unit	Command.Recipe[#].ProcessVariables.Parameter_LREAL[#].Unit	STRING(6)
				Value	Command.Recipe[#].ProcessVariables.Parameter_LREAL[#].Value	LREAL
			Parameter_DINT[#]		Command.Recipe[#].ProcessVariables.Parameter_DINT[#]	ST_PMLParameterDint
				Id	Command.Recipe[#].ProcessVariables.Parameter_DINT[#].Id	DINT
				Name	Command.Recipe[#].ProcessVariables.Parameter_DINT[#].Name	STRING
				Unit	Command.Recipe[#].ProcessVariables.Parameter_DINT[#].Unit	STRING(6)
				Value	Command.Recipe[#].ProcessVariables.Parameter_DINT[#].Value	DINT
		Ingredients			Command.Product[#].Ingredients	ST_PMLIngredient
			Parameter_REAL[#]		Command.Recipe[#].Ingredients.Parameter_REAL[#]	ST_PMLParameterReal
				Id	Command.Recipe[#].Ingredients.Parameter_REAL[#].Id	DINT
				Name	Command.Recipe[#].Ingredients.Parameter_REAL[#].Name	STRING
				Unit	Command.Recipe[#].Ingredients.Parameter_REAL[#].Unit	STRING(6)
				Value	Command.Recipe[#].Ingredients.Parameter_REAL[#].Value	REAL
			Parameter_STRING[#]		Command.Recipe[#].Ingredients.Parameter_STRING[#]	ST_PMLParameterString
				Id	Command.Recipe[#].Ingredients.Parameter_STRING[#].Id	DINT
				Name	Command.Recipe[#].Ingredients.Parameter_STRING[#].Name	STRING
				Unit	Command.Recipe[#].Ingredients.Parameter_STRING[#].Unit	STRING(6)
				Value	Command.Recipe[#].Ingredients.Parameter_STRING[#].Value	STRING
			Parameter_LREAL[#]		Command.Recipe[#].Ingredients.Parameter_LREAL[#]	ST_PMLParameterLreal
				Id	Command.Recipe[#].Ingredients.Parameter_LREAL[#].Id	DINT
				Name	Command.Recipe[#].Ingredients.Parameter_LREAL[#].Name	STRING
				Unit	Command.Recipe[#].Ingredients.Parameter_LREAL[#].Unit	STRING(6)
				Value	Command.Recipe[#].Ingredients.Parameter_LREAL[#].Value	LREAL

			Parameter_DINT[#]		Command.Recipe[#].Ingredients.Parameter_DINT[#]	ST_PMLParameterDint
				Id	Command.Recipe[#].Ingredients.Parameter_DINT[#].Id	DINT
				Name	Command.Recipe[#].Ingredients.Parameter_DINT[#].Name	STRING
				Unit	Command.Recipe[#].Ingredients.Parameter_DINT[#].Unit	STRING(6)
				Value	Command.Recipe[#].Ingredients.Parameter_DINT[#].Value	DINT

State structure Status

					Tag Name	**Data Type**
Status					Status	ST_PMLs
	UnitModeCurrent				Status.UnitModeCurrent	DINT
	UnitModeRequested				Status.UnitModerequested	DINT
	UnitModeChangeInProcess				Status.UnitModeChangeInProcess	BOOL
	StateCurrent				Status.StateCurrent	DINT
	StateRequested				Status.StateRequested	DINT
	StateChangeInProcess				Status.StateChangeInProcess	BOOL
	MachSpeed				Status.MachineSpeed	REAL
	CurMachSpeed				Status.CurMachineSpeed	REAL
	MaterialInterlock				Status.MaterialInterlock	DWORD
	EquipmentInterlock				Status.EquipmentInterlock	ST_PMLEquipment
		Blocked			Status.EquipmentInterlock.Blocked	BOOL
		Starved			Status.EquipmentInterlock.Starved	BOOL
	Parameter_REAL[#]				Status.Parameter_REAL[#]	ST_PMLParameterReal
		Id			Status.Parameter_REAL[#].Id	DINT
		Name			Status.Parameter_REAL[#].Name	STRING
		Unit			Status.Parameter_REAL[#].Unit	STRING(6)
		Value			Status.Parameter_REAL[#].Value	REAL
	Parameter_STRING[#]				Status.Parameter_STRING[#]	ST_PMLParameterString
		Id			Status.Parameter_STRING[#].Id	DINT
		Name			Status.Parameter_STRING[#].Name	STRING
		Unit			Status.Parameter_STRING[#].Unit	STRING(6)
		Value			Status.Parameter_STRING[#].Value	STRING
	Parameter_LREAL[#]				Status.Parameter_LREAL[#]	ST_PMLParameterLreal
		Id			Status.Parameter_LREAL[#].Id	DINT
		Name			Status.Parameter_LREAL[#].Name	STRING
		Unit			Status.Parameter_LREAL[#].Unit	STRING(6)
		Value			Status.Parameter_LREAL[#].Value	LREAL
	Parameter_DINT[#]				Status.Parameter_DINT[#]	ST_PMLParameterDint
		Id			Status.Parameter_DINT[#].Id	DINT
		Name			Status.Parameter_DINT[#].Name	STRING
		Unit			Status.Parameter_DINT[#].Unit	STRING(6)
		Value			Status.Parameter_DINT[#].Value	DINT
	RecipeCurrent				Status.RecipeCurrent	DINT
	RecipeRequested				Status.RecipeRequested	DINT
	RecipeChangeInProcess				Status.RecipeChangeInProcess	BOOL
	Recipe[#]				Status.Recipe[#]	ST_PMLRecipe
		Id			Status.Recipe[#].Id	DINT
		Name			Status.Recipe[#].Name	STRING
		Unit			Status.Recipe[#]..Unit	STRING(6)
		PrimaryQty			Status.Recipe[#].PrimaryQty	REAL

		ProcessVariables			Status.Recipe[#].ProcessVariables	ST_PMLProcessVariables
			Parameter_REAL[#]		Status.Recipe[#].ProcessVariables.Parameter_REAL[#]	ST_PMLParameterReal
				Id	Status.Recipe[#].ProcessVariables.Parameter_REAL[#].Id	DINT
				Name	Status.Recipe[#].ProcessVariables.Parameter_REAL[#].Name	STRING
				Unit	Status.Recipe[#].ProcessVariables.Parameter_REAL[#].Unit	STRING(6)
				Value	Status.Recipe[#].ProcessVariables.Parameter_REAL[#].Value	REAL
			Parameter_STRING[#]		Status.Recipe[#].ProcessVariables.Parameter_STRING[#]	ST_PMLParameterString
				Id	Status.Recipe[#].ProcessVariables.Parameter_STRING[#].Id	DINT
				Name	Status.Recipe[#].ProcessVariables.Parameter_STRING[#].Name	STRING
				Unit	Status.Recipe[#].ProcessVariables.Parameter_STRING[#].Unit	STRING(6)
				Value	Status.Recipe[#].ProcessVariables.Parameter_STRING[#].Value	STRING
			Parameter_LREAL[#]		Status.Recipe[#].ProcessVariables.Parameter_LREAL[#]	ST_PMLParameterLreal
				Id	Status.Recipe[#].ProcessVariables.Parameter_LREAL[#].Id	DINT
				Name	Status.Recipe[#].ProcessVariables.Parameter_LREAL[#].Name	STRING
				Unit	Status.Recipe[#].ProcessVariables.Parameter_LREAL[#].Unit	STRING(6)
				Value	Status.Recipe[#].ProcessVariables.Parameter_LREAL[#].Value	LREAL
			Parameter_DINT[#]		Status.Recipe[#].ProcessVariables.Parameter_DINT[#]	ST_PMLParameterDint
				Id	Status.Recipe[#].ProcessVariables.Parameter_DINT[#].Id	DINT
				Name	Status.Recipe[#].ProcessVariables.Parameter_DINT[#].Name	STRING
				Unit	Status.Recipe[#].ProcessVariables.Parameter_DINT[#].Unit	STRING(6)
				Value	Status.Recipe[#].ProcessVariables.Parameter_DINT[#].Value	DINT
		Ingredients			Status.Product[#].Ingredients	ST_PMLIngredient
			Parameter_REAL[#]		Status.Recipe[#].Ingredients.Parameter_REAL[#]	ST_PMLParameterReal
				Id	Status.Recipe[#].Ingredients.Parameter_REAL[#].Id	DINT
				Name	Status.Recipe[#].Ingredients.Parameter_REAL[#].Name	STRING
				Unit	Status.Recipe[#].Ingredients.Parameter_REAL[#].Unit	STRING(6)
				Value	Status.Recipe[#].Ingredients.Parameter_REAL[#].Value	REAL
			Parameter_STRING[#]		Status.Recipe[#].Ingredients.Parameter_STRING[#]	ST_PMLParameterString
				Id	Status.Recipe[#].Ingredients.Parameter_STRING[#].Id	DINT
				Name	Status.Recipe[#].Ingredients.Parameter_STRING[#].Name	STRING
				Unit	Status.Recipe[#].Ingredients.Parameter_STRING[#].Unit	STRING(6)

				Value	Status.Recipe[#].Ingredients.Parameter_STRING[#].Value	STRING
			Parameter_LREAL[#]		Status.Recipe[#].Ingredients.Parameter_LREAL[#]	ST_PMLParameterLreal
				Id	Status.Recipe[#].Ingredients.Parameter_LREAL[#].Id	DINT
				Name	Status.Recipe[#].Ingredients.Parameter_LREAL[#].Name	STRING
				Unit	Status.Recipe[#].Ingredients.Parameter_LREAL[#].Unit	STRING(6)
				Value	Status.Recipe[#].Ingredients.Parameter_LREAL[#].Value	LREAL
			Parameter_DINT[#]		Status.Recipe[#].Ingredients.Parameter_DINT[#]	ST_PMLParameterDint
				Id	Status.Recipe[#].Ingredients.Parameter_DINT[#].Id	DINT
				Name	Status.Recipe[#].Ingredients.Parameter_DINT[#].Name	STRING
				Unit	Status.Recipe[#].Ingredients.Parameter_DINT[#].Unit	STRING(6)
				Value	Status.Recipe[#].Ingredients.Parameter_DINT[#].Value	DINT
	StackLight[#]				Status.StackLight[#]	DWORD

Administration structure Admin

					Tag Name	**Data Type**
Admin					Admin	ST_PMLa
	Parameter_REAL[#]				Admin.Parameter_REAL[#]	ST_PMLParameterReal
		Id			Admin.Parameter_REAL[#].Id	DINT
		Name			Admin.Parameter_REAL[#].Name	STRING
		Unit			Admin.Parameter_REAL[#].Unit	STRING(6)
		Value			Admin.Parameter_REAL[#].Value	REAL
	Parameter_STRING[#]				Admin.Parameter_STRING[#]	ST_PMLParameterString
		Id			Admin.Parameter_STRING[#].Id	DINT
		Name			Admin.Parameter_STRING[#].Name	STRING
		Unit			Admin.Parameter_STRING[#].Unit	STRING(6)
		Value			Admin.Parameter_STRING[#].Value	STRING
	Parameter_LREAL[#]				Admin.Parameter_LREAL[#]	ST_PMLParameterLreal
		Id			Admin.Parameter_LREAL[#].Id	DINT
		Name			Admin.Parameter_LREAL[#].Name	STRING
		Unit			Admin.Parameter_LREAL[#].Unit	STRING(6)
		Value			Admin.Parameter_LREAL[#].Value	LREAL
	Parameter_DINT[#]				Admin.Parameter_DINT[#]	ST_PMLParameterDint
		Id			Admin.Parameter_DINT[#].Id	DINT
		Name			Admin.Parameter_DINT[#].Name	STRING
		Unit			Admin.Parameter_DINT[#].Unit	STRING(6)
		Value			Admin.Parameter_DINT[#].Value	DINT
	Alarm[#]				Admin.Alarm[#]	ST_PMLEvent
		Trigger			Admin.Alarm[#].Trigger	BOOL
		Id			Admin.Alarm[#].Id	DINT
		Value			Admin.Alarm[#].Value	DINT
		Message			Admin.Alarm[#].Message	STRING
		Category			Admin.Alarm[#].Category	DINT
		DateTime			Admin.Alarm[#].DateTime	ST_PMLDateAndTime
			Year		Admin.Alarm[#].DateTime.Year	DINT
			Month		Admin.Alarm[#].DateTime.Month	DINT

			Day		Admin.Alarm[#].DateTime.Day	DINT
			Hour		Admin.Alarm[#].DateTime.Hour	DINT
			Minute		Admin.Alarm[#].DateTime.Minute	DINT
			Second		Admin.Alarm[#].DateTime.Second	DINT
			mSec		Admin.Alarm[#].DateTime.mSec	DINT
		AckDateTime			Admin.Alarm[#].AckDateTime	ST_PMLDateAndTime
			Year		Admin.Alarm[#].AckDateTime.Year	DINT
			Month		Admin.Alarm[#].AckDateTime.Month	DINT
			Day		Admin.Alarm[#].AckDateTime.Day	DINT
			Hour		Admin.Alarm[#].AckDateTime.Hour	DINT
			Minute		Admin.Alarm[#].AckDateTime.Minute	DINT
			Second		Admin.Alarm[#].AckDateTime.Second	DINT
			mSec		Admin.Alarm[#].AckDateTime.mSec	DINT
	AlarmExtent				Admin.AlarmExtent	DINT
	AlarmHistory[#]				Admin.AlarmHistory[#]	ST_PMLEvent
		Trigger			Admin.AlarmHistory[#].Trigger	BOOL
		Id			Admin.AlarmHistory[#].Id	DINT
		Value			Admin.AlarmHistory[#].Value	DINT
		Message			Admin.AlarmHistory[#].Message	STRING
		Category			Admin.AlarmHistory[#].Category	DINT
		DateTime			Admin.AlarmHistory[#].DateTime	ST_PMLDateAndTime
			Year		Admin.AlarmHistory[#].DateTime.Year	DINT
			Month		Admin.AlarmHistory[#].DateTime.Month	DINT
			Day		Admin.AlarmHistory[#].DateTime.Day	DINT
			Hour		Admin.AlarmHistory[#].DateTime.Hour	DINT
			Minute		Admin.AlarmHistory[#].DateTime.Minute	DINT
			Second		Admin.AlarmHistory[#].DateTime.Second	DINT
			mSec		Admin.AlarmHistory[#].DateTime.mSec	DINT
		AckDateTime			Admin.AlarmHistory[#].AckDateTime	ST_PMLDateAndTime
			Year		Admin.AlarmHistory[#].AckDateTime.Year	DINT
			Month		Admin.AlarmHistory[#].AckDateTime.Month	DINT
			Day		Admin.AlarmHistory[#].AckDateTime.Day	DINT
			Hour		Admin.AlarmHistory[#].AckDateTime.Hour	DINT
			Minute		Admin.AlarmHistory[#].AckDateTime.Minute	DINT
			Second		Admin.AlarmHistory[#].AckDateTime.Second	DINT
			mSec		Admin.AlarmHistory[#].AckDateTime.msec	DINT
	AlarmHistoryExtent				Admin.AlarmHistoryExtent	DINT
	StopReason				Admin.StopReason	ST_PMLEvent
		Trigger			Admin.StopReason.Trigger	BOOL
		Id			Admin.StopReason.Id	DINT
		Value			Admin.StopReason.Value	DINT
		Message			Admin.StopReason.Message	STRING
		Category			Admin.StopReason.Category	DINT
		DateTime			Admin.StopReason.DateTime	ST_PMLDateAndTime
			Year		Admin.StopReason.DateTime.Year	DINT
			Month		Admin.StopReason.DateTime.Month	DINT
			Day		Admin.StopReason.DateTime.Day	DINT
			Hour		Admin.StopReason.DateTime.Hour	DINT
			Minute		Admin.StopReason.DateTime.Minute	DINT
			Second		Admin.StopReason.DateTime.Second	DINT
			mSec		Admin.StopReason.DateTime.mSec	DINT
		AckDateTime			Admin.StopReason.AckDateTime	ST_PMLDateAndTime
			Year		Admin.StopReason.AckDateTime.Year	DINT
			Month		Admin.StopReason.AckDateTime.Month	DINT
			Day		Admin.StopReason.AckDateTime.Day	DINT

			Hour		Admin.StopReason.AckDateTime.Hour	DINT
			Minute		Admin.StopReason.AckDateTime.Minute	DINT
			Second		Admin.StopReason.AckDateTime.Second	DINT
			mSec		Admin.StopReason.AckDateTime.mSec	DINT
	Warning[#]				Admin.Warning[#]	ST_PMLEvent
		Trigger			Admin.Warning [#].Trigger	BOOL
		Id			Admin.Warning[#].Id	DINT
		Value			Admin.Warning[#].Value	DINT
		Message			Admin.Warning[#].Message	STRING
		Category			Admin.Warning[#].Category	DINT
		DateTime			Admin.Warning[#].DateTime	ST_PMLDateAndTime
			Year		Admin.Warning[#].DateTime.Year	DINT
			Month		Admin.Warning[#].DateTime.Month	DINT
			Day		Admin.Warning[#].DateTime.Day	DINT
			Hour		Admin.Warning[#].DateTime.Hour	DINT
			Minute		Admin.Warning[#].DateTime.Minute	DINT
			Second		Admin.Warning[#].DateTime.Second	DINT
			mSec		Admin.Warning[#].DateTime.mSec	DINT
		AckDateTime			Admin.Warning[#].AckDateTime	ST_PMLDateAndTime
			Year		Admin.Warning[#].AckDateTime.Year	DINT
			Month		Admin.Warning[#].AckDateTime.Month	DINT
			Day		Admin.Warning[#].AckDateTime.Day	DINT
			Hour		Admin.Warning[#].AckDateTime.Hour	DINT
			Minute		Admin.Warning[#].AckDateTime.Minute	DINT
			Second		Admin.Warning[#].AckDateTime.Second	DINT
			mSec		Admin.Warning[#].AckDateTime.mSec	DINT
	WarningExtent				Admin.WarningExtent	DINT
	ModeTimeCurrent				Admin.ModeTimeCurrent	DINT
	StateTimeCurrent				Admin.StateTimeCurrent	DINT
	CummulativeTimes[#]				Admin.CummulativeTimes[#]	ST_PMLCumulativeTimes
		AccTimeSinceReset			Admin.CummulativeTime[#].AccTimeSinceReset	DINT
		ModeStateTimes[#]			Admin.CummulativeTime[#].ModeStateTimes[#]	ST_PMLModeStateTimes
			Mode		Admin.CummulativeTime[#].ModeStateTimes[#].Mode	DINT
			State[#]		Admin.CummulativeTime[#].ModeStateTimes[#].State[#]	ARRAY[#] OF DINT
	ProductData[#]				Admin.ProductData[#]	ST_PMLProductData
		Id			Admin.ProductData[#].Id	DINT
		Name			Admin.ProductData[#].Name	STRING
		Unit			Admin.ProductData[#].Unit	STRING(6)
		PrimaryQty			Admin.ProductData[#].PrimaryQty	DINT
		ConsumedCount			Admin.ProductData[#].ConsumedCout	DINT
		ProcessedCount			Admin.ProductData[#]ProcessCout	DINT
		DefectiveCount			Admin.ProductData[#].DefectiveCount	DINT
		AccConsumedCount			Admin.ProductData[#].AccConsumedCount	DINT
		AccProcessedCount			Admin.ProductData[#].AccProcessedCount	DINT
		AccDefectiveCount			Admin.ProductData[#].AccDefectiveCount	DINT
	MachDesignSpeed				Admin.MachDesignSpeed	REAL
	DisabledStatesCfg[#]				Admin.DisabledStatesCfg[#]	ARRAY [#] OF DWORD
	CurDisabledStates				Admin.CurDisabledStates	DWORD

	EnabledModesCfg				Admin.EnabledModesCfg	DWORD
	ModeTransitionCfg[#]				Admin.ModeTransitionCfg[#]	ARRAY [#] OF DWORD
	PlcDateTime				Admin.PlcDateTime	ST_PMLDateAndTime
		Year			Admin.PlcDateTime.Year	DINT
		Month			Admin.PlcDateTime.Month	DINT
		Day			Admin.PlcDateTime.Day	DINT
		Hour			Admin.PlcDateTime.Hour	DINT
		Minute			Admin.PlcDateTime.Minute	DINT
		Second			Admin.PlcDateTime.Second	DINT
		mSec			Admin.PlcDateTime.mSec	DINT

4 Function Blocks

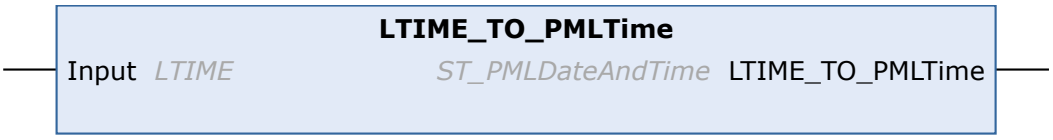
4.1 Conversion

Here you will find functions for converting times or timestamps in TwinCAT format to PackML format, as well as converting a PackML UnitMode or PackML State (both DINT) to a corresponding STRING.

4.1.1 Time

These function convert time values into the PackML-compliant array.

4.1.1.1 LTIME_TO_PMLTime



The LTIME_TO_PMLTime function converts a time value in LTIME format to the PackML-compliant format.

FUNCTION LTIME_TO_PMLTime : ST_PMLDateAndTime

Input

```
VAR_INPUT
    Input          : LTIME;
END_VAR
```

Name	Type	Description
Input	LTIME	The time value to be converted

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.1.2 TIME_TO_PMLTime

The TIME_TO_PMLTime function converts a time value in TIME format to the PackML-compliant format.

FUNCTION TIME_TO_PMLTime : ST_PMLDateAndTime

Input

```
VAR_INPUT
    Input          : TIME;
END_VAR
```

Name	Type	Description
Input	TIME	The time value to be converted

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.1.3 ULINT_TO_PMLTime

The ULINT_TO_PMLTime function converts a time value in ULINT format to the PackML-compliant format.

FUNCTION ULINT_TO_PMLTime : ST_PMLDateAndTime

Input

```
VAR_INPUT
    Input          : ULINT;
END_VAR
```

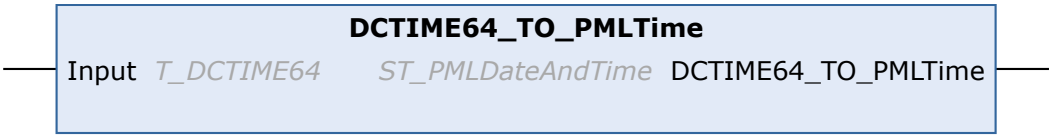
Name	Type	Description
Input	ULINT	The time value to be converted

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.2 Timestamp

4.1.2.1 DCTIME64_TO_PMLTime



The DCTIME64_TO_PMLTime function converts a time in DCTIME64 format to the PackML-compliant format.

FUNCTION DCTIME64_TO_PMLTime : ST_PMLDateAndTime

Input

```
VAR_INPUT
    Input          : DCTIME64;
END_VAR
```

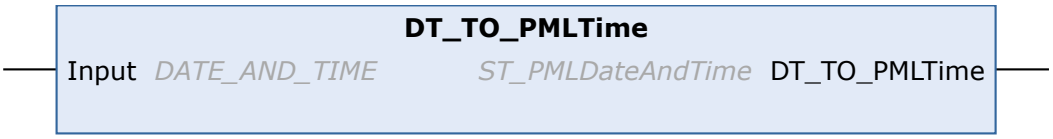
Name	Type	Description
Input	DCTIME64	The time to be converted.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT	PC (i386)	From Tc3_PackML_ V3 1.0.3

Development Environment	Target platform	PLC library to include
• 3.1 Build 4024.63		

4.1.2.2 DT_TO_PMLTime



The DT_TO_PMLTime function converts a time in DT format to PackML-compliant format.

FUNCTION DT_TO_PMLTime : ST_PMLDateAndTime

Input

```
VAR_INPUT
    Input          : DATE_AND_TIME;
END_VAR
```

Name	Type	Description
Input	DATE_AND_TIME	The time to be converted.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.2.3 TIMESTRUCT_TO_PMLTime

The TIMESTRUCT_TO_PMLTime function converts a time in TIMESTRUCT format to the PackML-compliant format.

FUNCTION TIMESTRUCT_TO_PMLTime : ST_PMLDateAndTime

Input

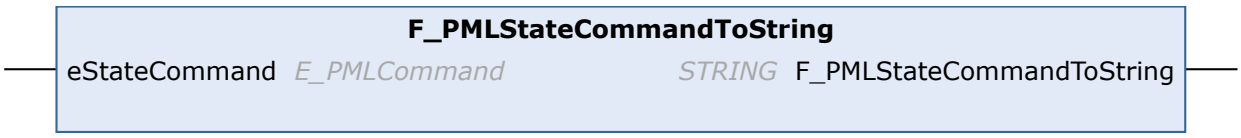
```
VAR_INPUT
    Input          : TIMESTRUCT;
END_VAR
```

Name	Type	Description
Input	TIMESTRUCT	The time to be converted.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.3 F_PMLStateCommandToString



The F_PMLStateCommandToString function outputs the name of a state command as a string.

FUNCTION F_PMLStateCommandToString : STRING;

Input

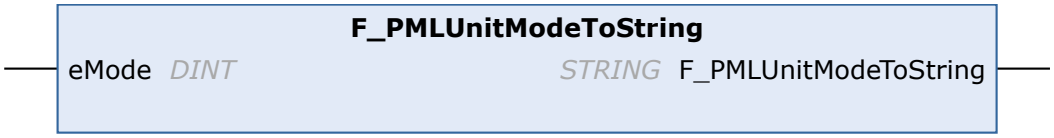
```
VAR_INPUT
    eStateCommand         : E_PMLCommand;
END_VAR
```

Name	Type	Description
eStateCommand	E_PMLCommand	The state command for which the name is to be determined.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.1.4 F_PMLUnitModeToString



The F_PMLUnitModeToString function returns the name of a Unit Mode as a string.

FUNCTION F_PMLUnitModeToString : STRING;

Input

```
VAR_INPUT
    eMode                 : DINT;
END_VAR
```

Name	Type	Description
eMode	DINT	The Unit Mode for which the name is to be determined.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2 General

General function blocks to simplify the handling of PackML structures

4.2.1 FB_PMLAdminAlarm

This function block supports entering, acknowledging, and deleting alarms, warnings, and StopReasons for Admin PackTags. The function block provides different methods for this.

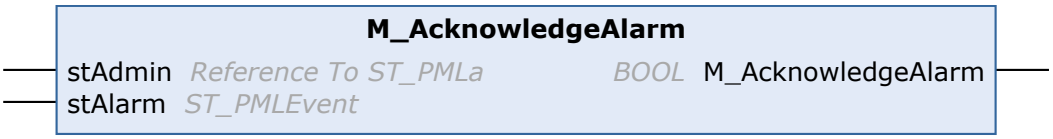
Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1 Alarm

Methods for handling alarms

4.2.1.1.1 M_AcknowledgeAlarm



This method acknowledges an alarm in the Admin-Tags. Alarm[].Trigger is set to FALSE and the value from Admin.PlcDateTime is entered in Alarm[].AckDateTime. The method returns TRUE if the alarm was found and acknowledged successfully. Acknowledging the alarm does not delete it. The alarm remains in the Alarm array until an M_ClearAlarm has been called, then it is moved to the AlarmHistory array. If the AlarmHistory array is already full of entries, the oldest entry is deleted as a result.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_AcknowledgeAlarm : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stAlarm          : ST_PMLEvent;
END_VAR
```

Example call:

```
AlarmAcknowledged := fbAdminAlarm.M_AcknowledgeAlarm(stAdmin := PackTags.Admin, stAlarm := Alarm);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.2 M_AcknowledgeAllAlarms

This method acknowledges all alarms in the admin tags. Alarm[].Trigger is set to FALSE and the value from Admin.PlcDateTime is entered in Alarm[].AckDateTime. If all alarms have been acknowledged, the method returns TRUE. Acknowledging the alarms does not delete them. The alarms remain in the alarm array until an M_ClearAlarm or M_ClearAllAlarms call has also been made, at which point the alarms in question are moved to the AlarmHistory array. If the AlarmHistory array is already full of entries, the oldest entries are deleted as a result.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_AcknowledgeAllAlarms : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

Example call:

```
AllAlarmsAcknowledged := fbAdminAlarm.M_AcknowledgeAllAlarms(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.3 M_ClearAlarm

This method deletes an alarm from the Admin-Tags. Alarm[].Trigger is set to FALSE. The method returns TRUE if the alarm was deleted successfully. The alarm remains in the Alarm array until an M_AcknowledgeAlarm has been called, then it is moved to the AlarmHistory array. If the AlarmHistory array is already full of entries, the oldest entry is deleted as a result.

Syntax

```
METHOD M_ClearAlarm : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stAlarm          : ST_PMLEvent;
END_VAR
```

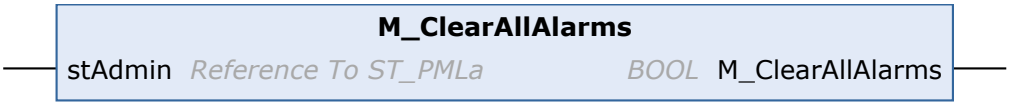
Example call:

```
AlarmCleared := fbAdminAlarm.M_ClearAlarm(stAdmin := PackTags.Admin, stAlarm := Alarm);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.4 M_ClearAllAlarms



This method deletes all alarms in the Admin-Tags. Alarm[].Trigger is set to FALSE. If the alarms were successfully deleted, the method returns TRUE. The alarms remain in the alarm array until an M_AcknowledgeAlarm or M_AcknowledgeAllAlarms call has also been made, at which point they are moved to the AlarmHistory array. If the AlarmHistory array is already full of entries, the oldest entries are deleted as a result.

Syntax

```
METHOD M_ClearAllAlarms : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

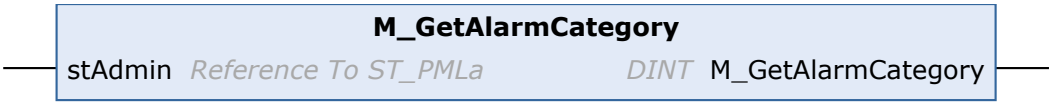
Example call:

```
AllAlarmsCleared := fbAdminAlarm.M_ClearAllAlarms(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.5 M_GetAlarmCategory



This method returns the highest AlarmCategory (smallest value) in the alarms of the Admin-Tags.

Syntax

```
METHOD M_GetAlarmCategory : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

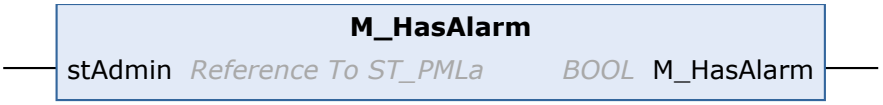
Example call:

```
AlarmCategory := fbAdminAlarm.M_GetAlarmCategory(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.6 M_HasAlarm



This method checks whether an active alarm is entered in the Admin-Tags and the method returns TRUE

Syntax

```
METHOD M_HasAlarm : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

Example call:

```
HasAlarm := fbAdminAlarm.M_HasAlarm(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.1.7 M_SetAlarm

This method inserts an alarm in the Admin-Tags. Alarm[].Trigger is set to TRUE and the value from Admin.PlcDateTime is entered in Alarm[].DateTime. The other values are taken from the transferred alarm structure. The method returns TRUE if the alarm was entered successfully

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_SetAlarm : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stAlarm          : ST_PMLEvent;
END_VAR
```

Example call:

```
AlarmInserted := fbAdminAlarm.M_SetAlarm(stAdmin := PackTags.Admin, stAlarm := Alarm);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.2 StopReason

Methods for handling stop causes

4.2.1.2.1 M_AcknowledgeStopReason

This method acknowledges the StopReason in the Admin-Tags. StopReason.Trigger is set to FALSE and the value from Admin.PlcDateTime is entered in StopReason.AckDateTime. If the StopReason was successfully acknowledged, the method returns TRUE. The StopReason remains in the Admin-Tags until it is replaced by a subsequent StopReason.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_AcknowledgeStopReason: BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

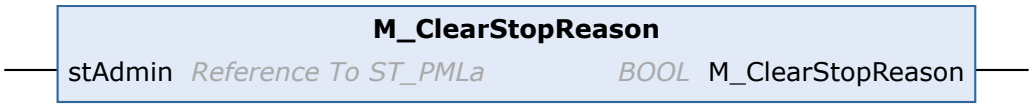
Example call:

```
StopReasonAcknowledged := fbAdminAlarm.M_AcknowledgeStopReason(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.2.2 M_ClearStopReason



This method deletes the StopReason from the Admin-Tags. StopReason.Trigger is set to FALSE. The method returns TRUE if the StopReason was deleted successfully. The StopReason remains in the Admin-Tags until it is replaced by a subsequent StopReason.

Syntax

```
METHOD M_ClearStopReason : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

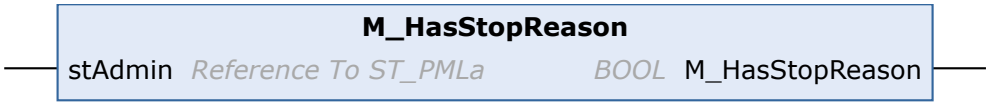
Example call:

```
StopReasonCleared := fbAdminAlarm.M_ClearStopReason(stAdmin := PackTags.Admin)
;
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.2.3 M_HasStopReason



This method checks whether an active StopReason is entered in the Admin-Tags and the method returns TRUE

Syntax

```
METHOD M_SetStopReason : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

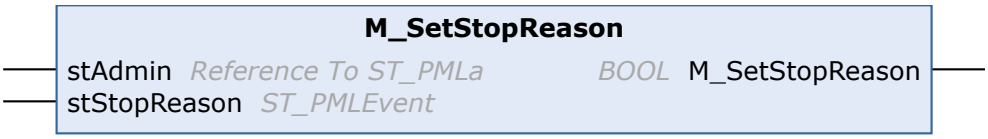
Example call:

```
HasStopReason := fbAdminAlarm.M_HasStopReason (stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.2.4 M_SetStopReason



This method inserts a StopReason in the Admin-Tags. StopReason[].Trigger is set to TRUE and the value from Admin.PlcDateTime is entered in StopReason[].DateTime. The other values are taken from the transferred StopReason structure. The method returns TRUE if the StopReason was entered successfully. If the StopReason array is already full of entries, the oldest entry is deleted as a result.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_SetStopReason : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stStopReason     : ST_PMLEvent;
END_VAR
```

Example call:

```
StopReasonInserted := fbAdminAlarm.M_SetStopReason
(stAdmin := PackTags.Admin, stStopReason := StopReason);
```

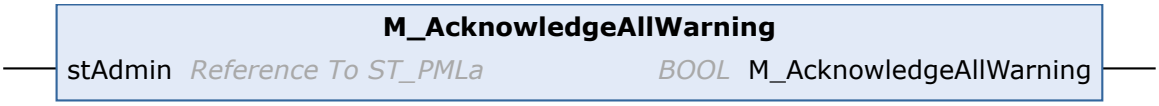
Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.3 Warning

Methods for handling warnings

4.2.1.3.1 M_AcknowledgeAllWarning



This method acknowledges all warnings in the Admin-Tags. Warning[].Trigger is set to FALSE and the value from Admin.PlcDateTime is entered in Warning[].AckDateTime. If all warnings are acknowledged, the method returns TRUE. The warnings remain in the Warning array until they are pushed out of the array by the next warnings.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_AcknowledgeAllWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

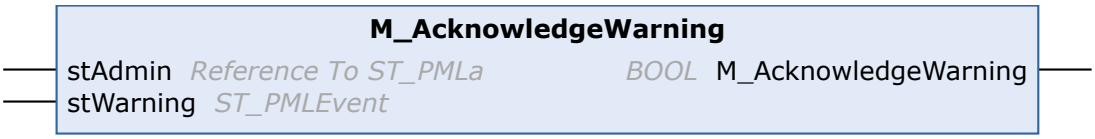
Example call:

```
AllWarningAcknowledged := fbAdminAlarm.M_AcknowledgeAllWarning(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.3.2 M_AcknowledgeWarning



This method acknowledges a warning in the Admin-Tags. Warning[].Trigger is set to FALSE and the value from Admin.PlcDateTime is entered in Warning[].AckDateTime. The method returns TRUE if the warning was found and acknowledged successfully. The warning remains in the Warning array until it is pushed out of the array by the next warning.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_AcknowledgeWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stWarning        : ST_PMLEvent;
END_VAR
```

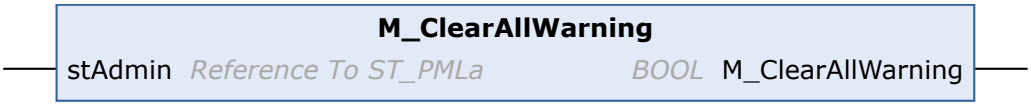
Example call:

```
WarningAcknowledged := fbAdminAlarm.M_AcknowledgeWarning(stAdmin := PackTags.Admin, stWarning := War
ning);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.3.3 M_ClearAllWarning



This method deletes all warnings in the Admin-Tags. Warning[].Trigger is set to FALSE. The method returns TRUE if the warning was deleted successfully. The warnings remain in the Warning array until they are pushed out of the array by the next warnings.

Syntax

```
METHOD M_ClearAllWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

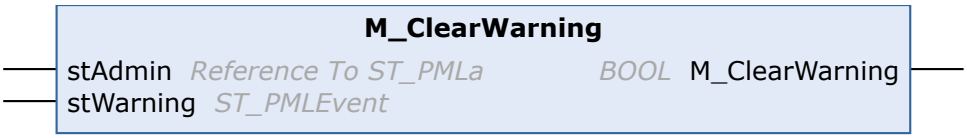
Example call:

```
AllWarningCleared := fbAdminAlarm.M_ClearAllWarning(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.3.4 M_ClearWarning



This method deletes a warning in the Admin-Tags. Warning[].Trigger is set to FALSE. The method returns TRUE if the warning was deleted successfully. The warning remains in the Warning array until it is pushed out of the array by the next warning.

Syntax

```
METHOD M_ClearWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stWarning        : ST_PMLEvent;
END_VAR
```

Example call:

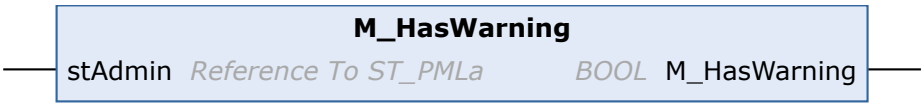
```
WarningCleared := fbAdminAlarm.M_ClearWarning(stAdmin := PackTags.Admin, stWarning := Warning);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT	PC (i386)	From Tc3_PackML_ V3 1.0.3

Development Environment	Target platform	PLC library to include
• 3.1 Build 4024.63		

4.2.1.3.5 M_HasWarning



This method checks whether an active warning is entered in the Admin-Tags and the method returns TRUE

Syntax

```
METHOD M_HasWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
```

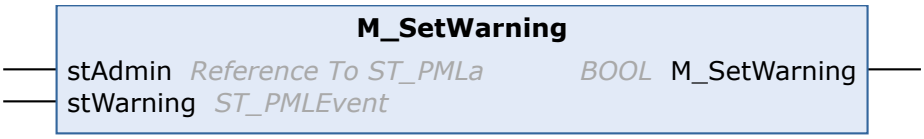
Example call:

```
HasWarning := fbAdminAlarm.M_HasWarning(stAdmin := PackTags.Admin);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.2.1.3.6 M_SetWarning



This method inserts a warning in the Admin-Tags. Warning[].Trigger is set to TRUE and the value from Admin.PlcDateTime is entered in Warning[].DateTime. The other values are taken from the transferred warning structure. The method returns TRUE if the warning was entered successfully. If the Warning array is already full of entries, the oldest entry is deleted as a result.

So that a valid timestamp can be entered, the PML_AdminTime function block should be called cyclically in the program.

Syntax

```
METHOD M_SetWarning : BOOL
VAR_IN_OUT
  stAdmin          : ST_PMLa;
END_VAR
VAR_INPUT
  stWarning        : ST_PMLEvent;
END_VAR
```

Example call:

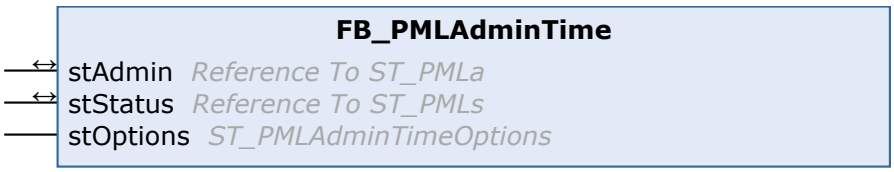
```
WarningInserted := fbAdminAlarm.M_SetWarning(stAdmin := PackTags.Admin, stWarning := Warning);
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT	PC (i386)	From Tc3_PackML_ V3 1.0.3

Development Environment	Target platform	PLC library to include
• 3.1 Build 4024.63		

4.2.2 FB_PMLAdminTime



The FB_PMLAdminTime function block should be called cyclically to fill the following Admin-PackTags:

- PlcDateTime
- ModeTimeCurrent
- StateTimeCurrent
- CumulativeTimes[ ].AccTimeSinceReset
- CumulativeTimes[ ].ModeStateTimes[ ].Mode
- CumulativeTimes[ ].ModeStateTimes[ ].State[ ]

The length of time for which the machine was in the different states is thus recorded. In the further process, this allows conclusions to be drawn about the machine efficiency. To ensure that the times are calculated correctly, it is a prerequisite that the *UnitCurrent* and *StateCurrent* Status-PackTags have already been written coherently.

The individual timers can be reset using the M_ResetCumulativeTime method.

Input

```
VAR_INPUT
    stOptions        : ST_AdminTimeOptions;
END_VAR
```

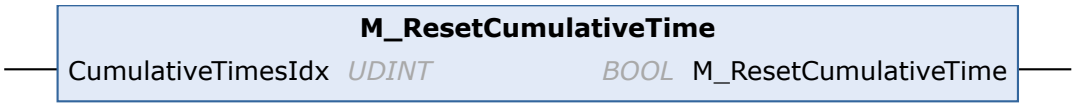
Name	Type	Description
stOptions	ST_PMLAdminTimeOptions	Additional options of the function block

Inputs/Outputs

```
VAR_IN_OUT
    stAdmin          : ST_PMLa;
    stStatus         : ST_PMLs;
END_VAR
```

Name	Type	Description
stAdmin	ST_PMLa	Transfer of the Admin-PackTags
stStatus	ST_PMLs	Transfer of the Status-PackTags

M_ResetCumulativeTime



The M_ResetCumulativeTime method resets the Admin.CumulativeTimes[CumulativeTimesIdx] entry.

Syntax

```
METHOD M_ResetCumulativeTime : BOOL
VAR_INPUT
  CumulativeTimesIdx : UDINT;
END_VAR
```

Example call:

```
  fbAdminTime.M_ResetCumulativeTime(CumulativeTimesIdx := 1);
```

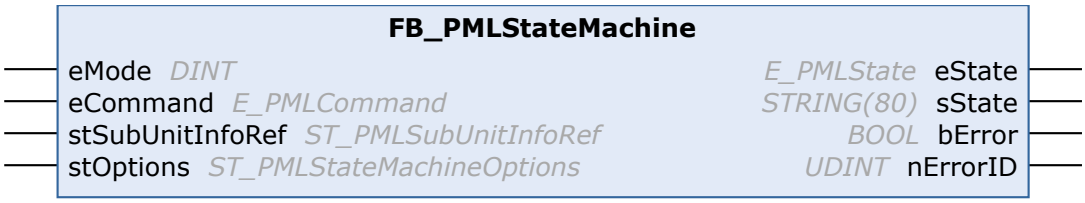
Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.3 Packaging Machine State

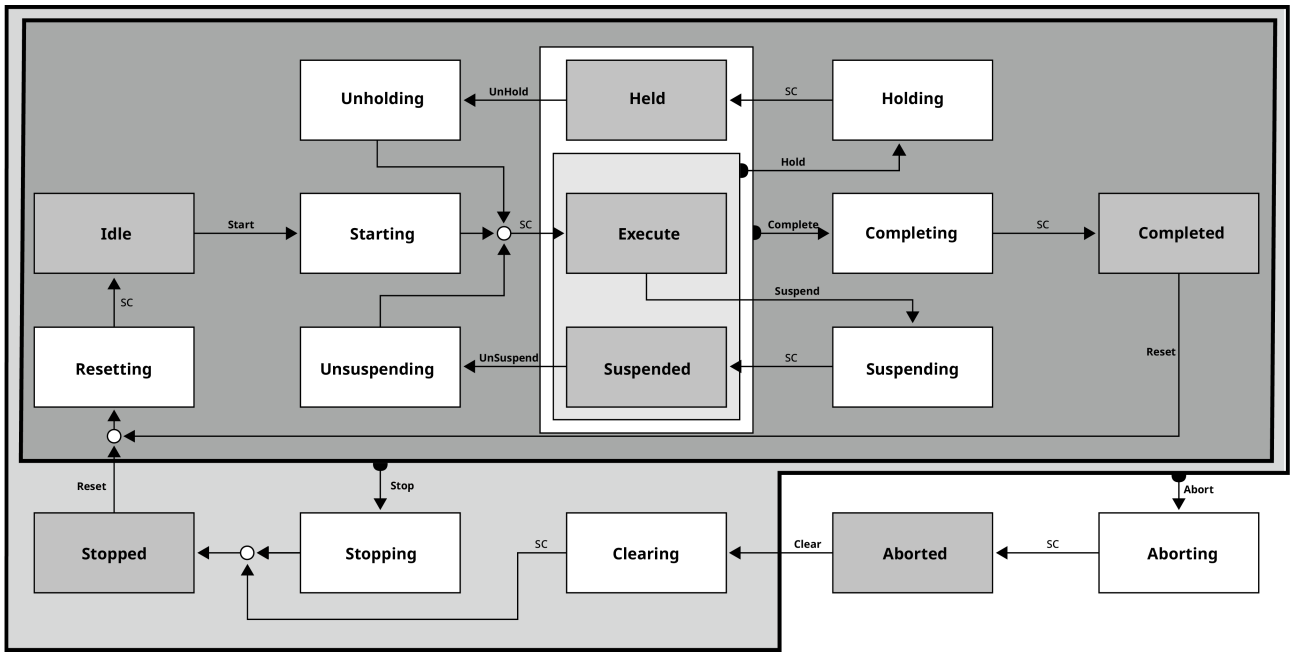
Function blocks for simplified handling of the PackML state machine and unit modes

4.3.1 FB_PMLStateMachine

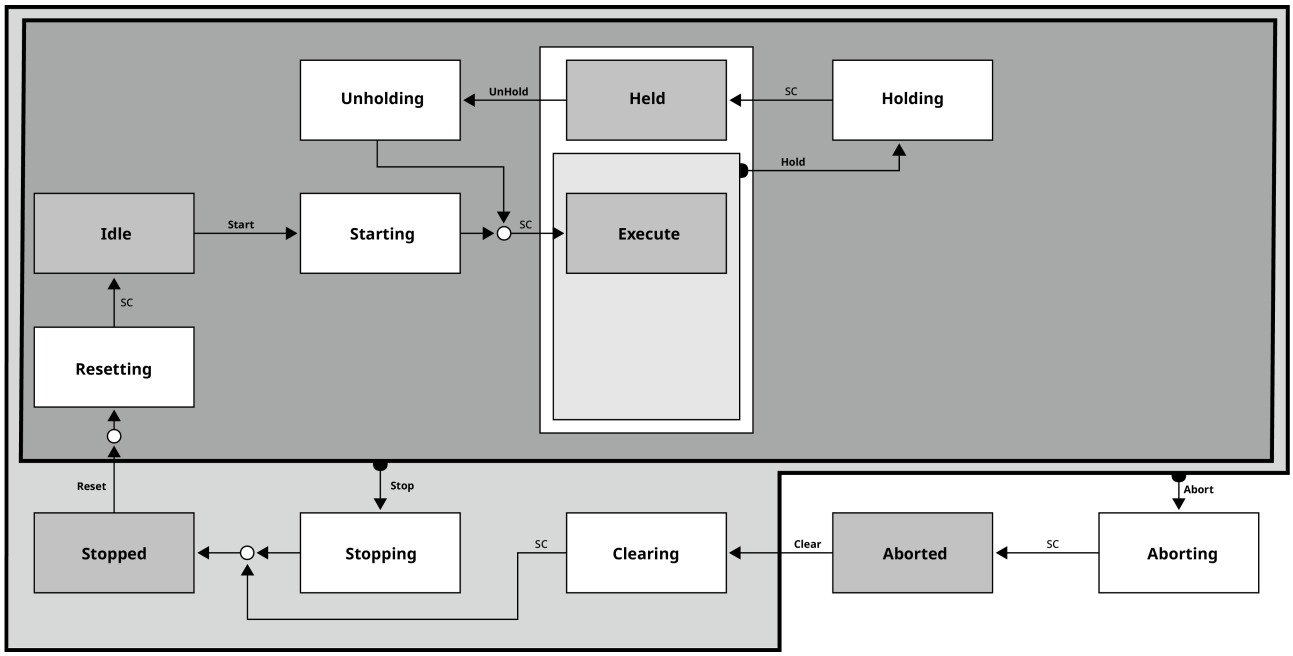


In its updated form, the PML_StateMachine function block has a common interface to the PackML Machine State Model V3. It is expected that application-specific logic, such as state transitions, will be programmed in external function blocks and that the PML_StateMachine function block will take over the central logic of the state machine and the state representation. The Machine State Model has a different appearance due to the currently active UnitMode (eMode). Three basic models are preconfigured for this purpose (E_PMLProtectedUnitMode [▶ 44]).

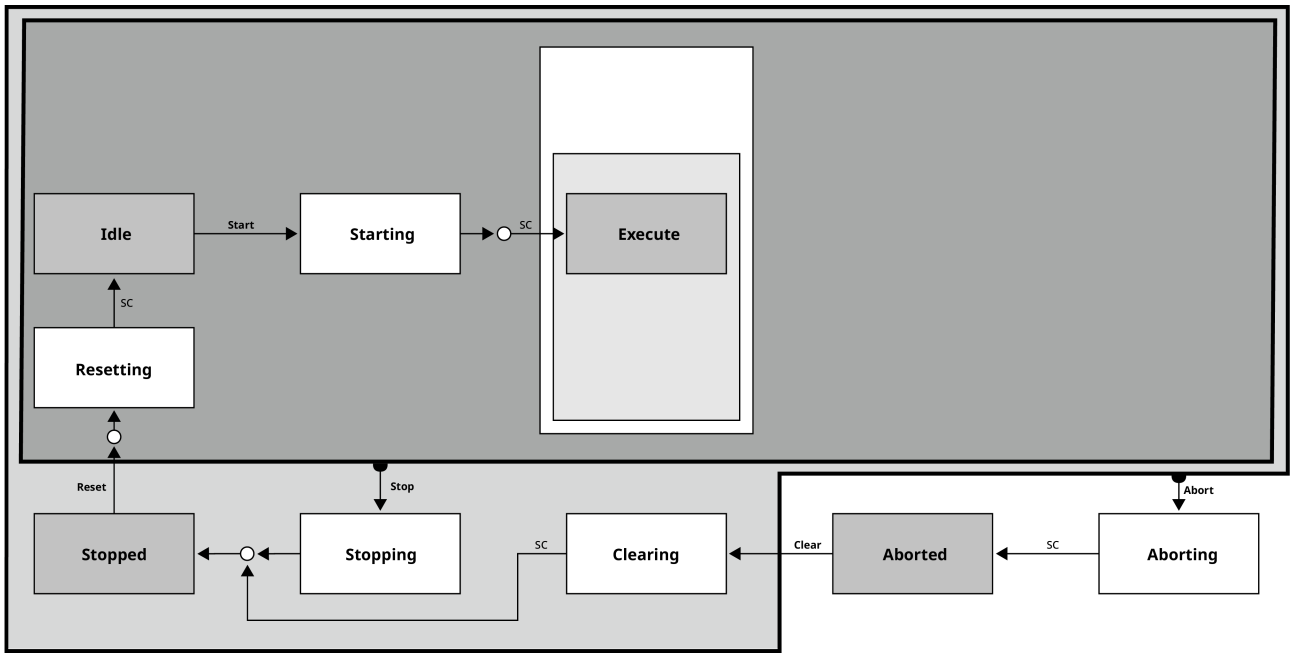
ePMLProtUnitMode_Production



ePMLProtUnitMode_Maintenance



ePMLProtUnitMode_Manual



Furthermore, other user-specific models can be created in a simple manner with the aid of the FB_PMLUnitModeConfig [▶ 38] function block and are thus very flexible in use.

The logic for transitions, in particular between production, maintenance and manual mode, depends on the application. The states in which UnitMode changes are permissible for the basic models are described more precisely in the description of the FB_PMLUnitModeManager [▶ 41] function block.

Inputs

```
VAR_INPUT
    eMode               : DINT;
    eCommand            : E_PMLCommand;
    stSubUnitInfoRef    : ST_PMLSubUnitInfoRef;
    stOptions           : ST_PMLStateMachineOptions;
END_VAR
```

Name	Type	Description
eMode	DINT	Current PML UnitMode
eCommand	E_PMLCommand	Enumeration [▶ 44] with the various PML commands of the function block.
stSubUnitInfoRef	ST_PMLSubUnitInfoRef	Structure [▶ 47] that points to an array of the current PML states of subordinated machine units
stOptions	ST_PMLStateMachineOptions	Not used at present

Outputs

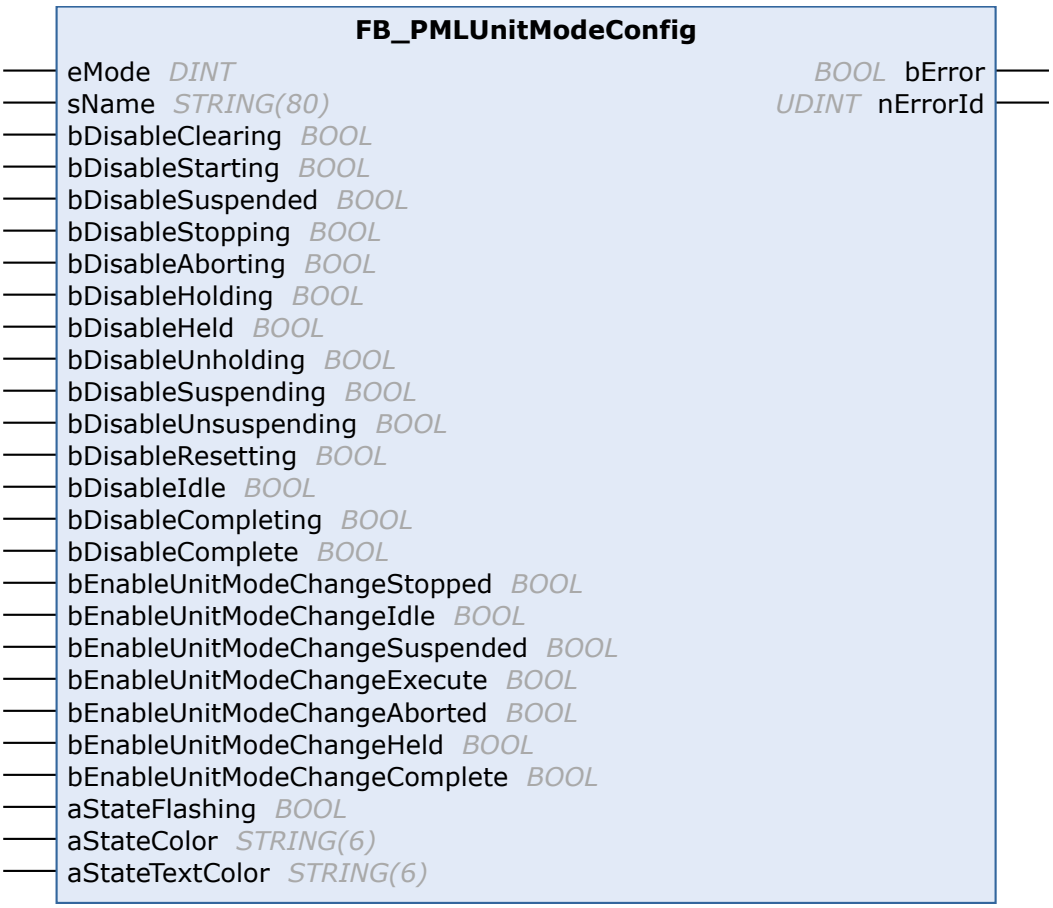
```
VAR_OUTPUT
    eState      : E_PMLState;
    sState      : STRING;
    bError      : BOOL;
    nErrorId    : UDINT;
END_VAR
```

Name	Type	Description
eState	E_PMLState	Enumeration [▶ 46] that delivers the current PML state of the automatic state machine.
sState	STRING	Name of the current PML state of the automatic state machine.
bError	BOOL	Becomes TRUE as soon as an error occurs.
nErrorId	UDINT	Supplies the error number when the bError output is set.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.3.2 FB_PMLUnitModeConfig



Machines may have unit modes other than "Production", "Maintenance" and "Manual". This function block enables the user to configure further models (UnitModes).

The number of the new model, the existing states and the states in which a model change is possible can be freely defined.

From version 1.0.5.0 of the library, the activation of a flashing and the state and state text color can also be stored here.

The configuration is then part of the global structure ST_PMLUnitModeConfiguration [▶ 47].

Inputs

```
VAR_INPUT
    eMode                           : DINT;
    sName                           : STRING;
    bDisableClearing                : BOOL;
    bDisableStarting                : BOOL;
    bDisableSuspended               : BOOL;
    bDisableStopping                : BOOL;
    bDisableAborting                : BOOL;
    bDisableHolding                 : BOOL;
    bDisableHeld                    : BOOL;
    bDisableUnholding               : BOOL;
    bDisableSuspending              : BOOL;
    bDisableUnsuspending            : BOOL;
    bDisableResetting               : BOOL;
    bDisableIdle                    : BOOL;
    bDisableCompleting              : BOOL;
    bDisableComplete                : BOOL;
    bEnableUnitModeChangeStopped    : BOOL;
    bEnableUnitModeChangeIdle       : BOOL;
    bEnableUnitModeChangeSuspended  : BOOL;
    bEnableUnitModeChangeExecute    : BOOL;
    bEnableUnitModeChangeAborted    : BOOL;
    bEnableUnitModeChangeHeld       : BOOL;
```

```
    bEnableUnitModeChangeComplete   : BOOL;

    aStateFlashing                  : ARRAY [E_PMLState.Undefined..E_PMLState.Completed] OF BOOL :=
aStateFlashingDefault; //Extension Tc3_PackML_V3 >= 1.0.5.0
    aStateColor                     : ARRAY [E_PMLState.Undefined..E_PMLState.Completed] OF
STRING[6] := aStateColorDefault; //Extension Tc3_PackML_V3 >= 1.0.5.0
    aStateTextColor                 : ARRAY [E_PMLState.Undefined..E_PMLState.Completed] OF
STRING[6] := aStateTextColorDefault; //Extension Tc3_PackML_V3 >= 1.0.5.0
END_VAR
```

Name	Type	Description
eMode	DINT	Number of the new PML UnitMode [4..31]
sName	STRING	Name of the new PML UnitMode
bDisableClearing	BOOL	Disables the "Clearing" PML state.
bDisableStarting	BOOL	Disables the "Starting" PML state.
bDisableSuspended	BOOL	Disables the "Suspended" PML state. Disabling the static state also causes the "Suspending" and "Unsuspending" PML states to be disabled.
bDisableStopping	BOOL	Disables the "Stopping" PML state.
bDisableAborting	BOOL	Disables the "Aborting" PML state.
bDisableHolding	BOOL	Disables the "Holding" PML state.
bDisableHeld	BOOL	Disables the "Held" PML state. Disabling the static state also causes the "Holding" and "Unholding" PML states to be disabled.
bDisableUnholding	BOOL	Disables the "Unholding" PML state.
bDisableSuspending	BOOL	Disables the "Suspending" PML state.
bDisableUnsuspending	BOOL	Disables the "Unsuspending" PML state.
bDisableResetting	BOOL	Disables the "Resetting" PML state.
bDisableIdle	BOOL	Disables the "Idle" PML state. Disabling the static state also causes the "Resetting" PML state to be disabled.
bDisableCompleting	BOOL	Disables the "Completing" PML state.
bDisableComplete	BOOL	Disables the "Complete" PML state. Disabling the static state also causes the "Completing" PML state to be disabled.
bEnableUnitModeChangeStopped	BOOL	Enables a mode change in the "Stopped" PML state.
bEnableUnitModeChangeIdle	BOOL	Enables a mode change in the "Idle" PML state.
bEnableUnitModeChangeSuspended	BOOL	Enables a mode change in the "Suspended" PML state.
bEnableUnitModeChangeExecute	BOOL	Enables a mode change in the "Execute" PML state.
bEnableUnitModeChangeAborted	BOOL	Enables a mode change in the "Aborted" PML state.
bEnableUnitModeChangeHeld	BOOL	Enables a mode change in the "Held" PML state.
bEnableUnitModeChangeComplete	BOOL	Enables a mode change in the "Complete" PML state.
aStateFlashing	ARRAY[0..17] OF BOOL	Defines the flashing of the states when active.
aStateColor	ARRAY[0..17] OF STRING[6]	Defines the color of the states when active.
aStateTextColor	ARRAY[0..17] OF STRING[6]	Defines the text color of the states when active

Outputs

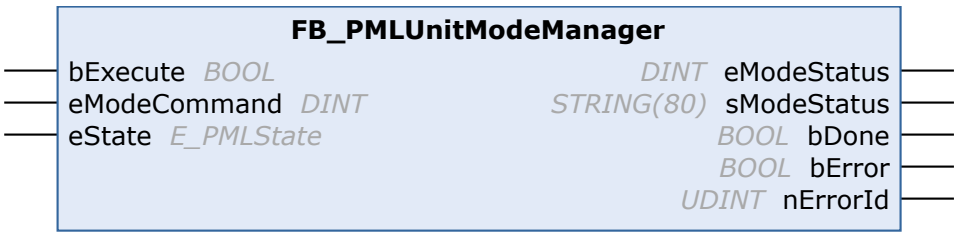
```
VAR_OUTPUT
    bError     : BOOL;
    nErrorID   : UDINT;
END_VAR
```

Name	Type	Description
bError	BOOL	Becomes TRUE as soon as an error occurs.
nErrorId	UDINT	Supplies the error number when the bError output is set.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

4.3.3 FB_PMLUnitModeManager

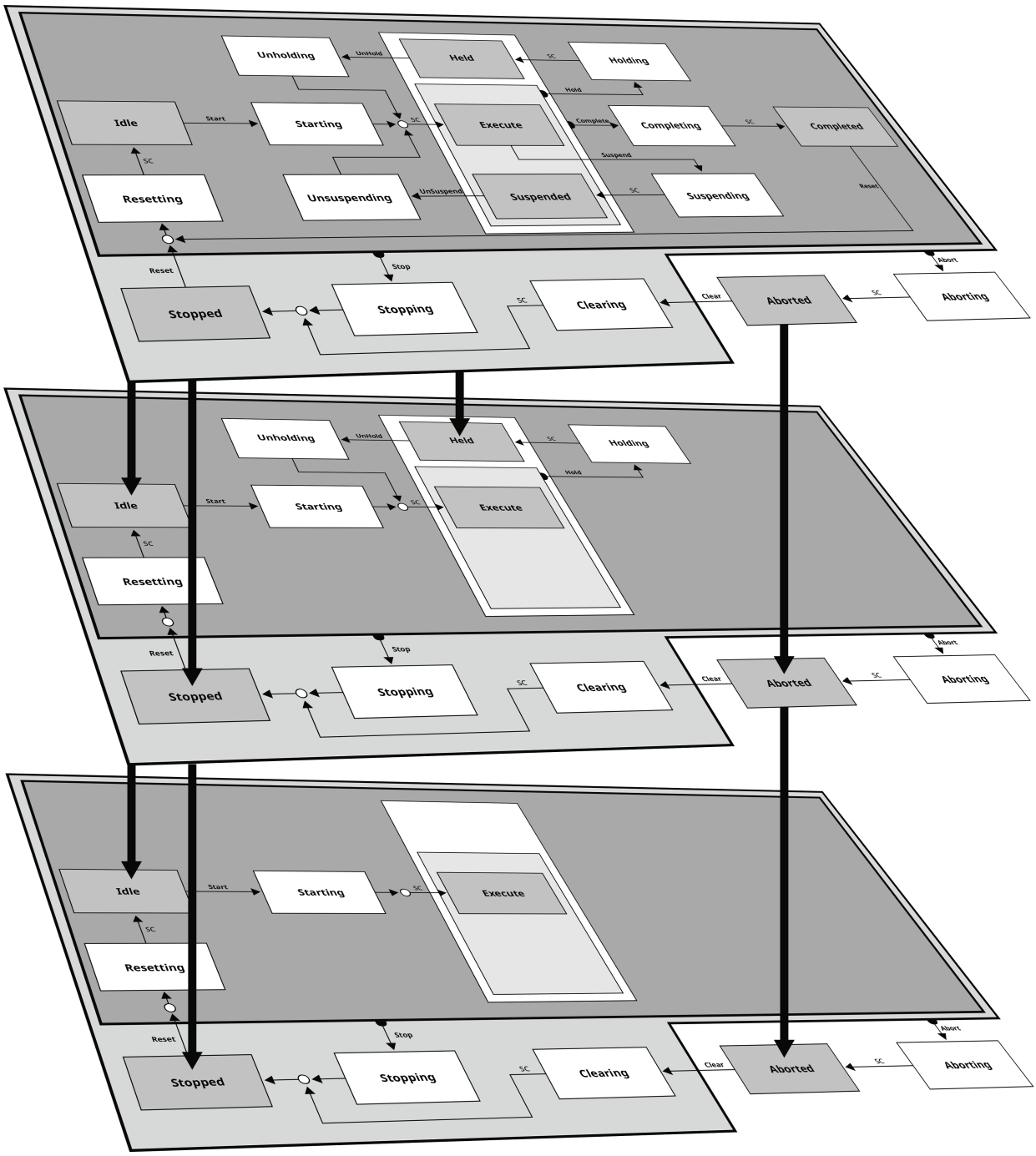


Machines have other system modes apart from “Production”. Each unit mode is defined by its own state model. A "Mode Manager" must be defined for transitions between the modes. The “Mode Manager” decides how and in which state a machine can change unit modes; i.e. built-in barriers prevent the machine from changing to unsuitable states. These barriers are predefined for the base modes “Production”, “Maintenance” and “Manual”, as the illustration below shows. This can be individually specified for other modes defined via the FB_PMLUnitModeConfig [▶ 38] function block.

⚠ WARNING

Adhere to proper mode changes

The logic for transitions between the modes depends on the application, especially for transitions between "Manual" and "Production" mode. In addition, hardware barriers or safety equipment may be necessary for such mode changes. The responsibility for proper mode changes lies with whoever implements them.



Inputs

```
VAR_INPUT
    bExecute        : BOOL;
    eModeCommand    : DINT;
    eState          : E_PMLState;
END_VAR
```

Name	Type	Description
bExecute	BOOL	Mode change on rising edge
eModeCommand	DINT	Requested mode
eState	E_PMLState	Enumeration [▶ 46] that delivers the current PML state of the automatic state machine.

Outputs

```
VAR_OUTPUT
    eModeStatus     : DINT;
    sModeStatus     : STRING;
    bDone           : BOOL;
    bError          : BOOL;
    bErrorID        : UDINT;
END_VAR
```

Name	Type	Description
eModeStatus	DINT	Current PML UnitMode
sModeStatus	STRING	Name of the current PML UnitMode
bDone	BOOL	Becomes TRUE as soon as the mode change has been successfully carried out.
bError	BOOL	Becomes TRUE as soon as an error occurs.
nErrorID	UDINT	Supplies the error number when the bError output is set.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5 Data types

Data types that are used in the PackML library.

5.1 General

General data types

5.1.1 ST_PMLAdminTimeOptions

```
TYPE ST_PMLAdminTimeOptions :
STRUCT
    UseExternalTime          : BOOL;
    ExternalPackMLTime       : ST_PMLDateAndTime;
END_STRUCT
END_TYPE
```

Name	Description
UseExternalTime	If this flag is set to TRUE, the time set at the input ExternalPackMLTime is used instead of the system time information.
ExternalPackMLTime	Externally set time

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2 Packaging machine state and mode

PackML state and mode handling data types

5.2.1 E_PMLCommand

```
TYPE E_PMLCommand :
(
    Undefined   := 0,
    Reset       := 1,
    Start       := 2,
    Stop        := 3,
    Hold        := 4,
    Unhold      := 5,
    Suspend     := 6,
    Unsuspend   := 7,
    Abort       := 8,
    Clear       := 9,
    Complete    := 10
);
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.2 E_PMLProtectedUnitMode

```
TYPE E_PMLProtectedUnitMode :
(
    Invalid         := 0,
    Production      := 1,
    Maintenance     := 2,
```

```
    Manual          := 3
);
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.3 E_PMLState

```
TYPE E_PMLState :
(
    Undefined     := 0,
    Clearing      := 1,
    Stopped       := 2,
    Starting      := 3,
    Idle          := 4,
    Suspended     := 5,
    Execute       := 6,
    Stopping      := 7,
    Aborting      := 8,
    Aborted       := 9,
    Holding       := 10,
    Held          := 11,
    Unholding     := 12,
    Suspending    := 13,
    Unsuspending  := 14,
    Resetting     := 15,
    Completing    := 16,
    Completed     := 17
);
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.4 ST_PMLStateMachineOptions

```
TYPE ST_PMLStateMachineOptions :
STRUCT
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.5 ST_PMLSubUnitInfo

```
TYPE ST_PMLSubUnitInfo :
STRUCT
    Active   : BOOL;
    eState   : E_PMLState := E_PMLState.Aborted;
END_STRUCT
END_TYPE
```

Name	Description
Active	Signals that this subordinated machine part is active and follows the state presets of the state machine.
eState	Enumeration, that reflects the current state of the subordinated machine part.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.6 ST_PMLSubUnitInfoRef

```
TYPE ST_PMLSubUnitInfoRef :
STRUCT
    pArray          : PVOID;
    ArraySize       : UDINT;
    NoOfSubUnits    : UDINT;
END_STRUCT
END_TYPE
```

Name	Description
pArray	Address of a one-dimensional array of the type ST_PMLSubUnitInfo [▶ 46]. Each array element contains the state of a subordinated machine part. Sample: stSubUnitInfo : ARRAY[1..10] OF ST_PMLSubUnitInfo [▶ 46]; pArray := ADR(stSubUnitInfo);
ArraySize	Size of the one-dimensional array, which can be determined with the SIZEOF function. Sample: ArraySize := SIZEOF(stSubUnitInfo);
NoOfSubUnits	Number of relevant subordinated machine parts.

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.2.7 ST_PMLUnitModeConfiguration

This structure is used for the GlobalVariables *stPMLUnitModeConfiguration*. The information about the permitted states and transitions configured with FB_PMLUnitModeConfig [▶ 38] is stored there.

```
TYPE ST_PMLUnitModeConfiguration :
STRUCT
    bActive             : BOOL;

    aStateDisabled      : ARRAY[E_PMLState.Undefined..E_PMLState.Completed] OF BOOL;
    aModeChangeValid    : ARRAY[E_PMLState.Undefined..E_PMLState.Completed] OF BOOL;

    sName               : STRING;

    aStateFlashing      : ARRAY[E_PMLState.Undefined..E_PMLState.Completed] OF BOOL; //
Extension Tc3_PackML_V3 >= 1.0.5.0
    aStateColor         : ARRAY[E_PMLState.Undefined..E_PMLState.Completed] OF STRING[6];  //
Extension Tc3_PackML_V3 >= 1.0.5.0
    aStateTextColor     : ARRAY[E_PMLState.Undefined..E_PMLState.Completed] OF STRING[6];  //
Extension Tc3_PackML_V3 >= 1.0.5.0
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3 PackTags

Data types of the PackML PackTags

5.3.1 Alarm

Data types for alarm, warning and stop cause handling

5.3.1.1 ST_PMLDateAndTime

This structure is used for saving the date and time of an event or for the acknowledgement of an event.

```
TYPE ST_PMLDateAndTime :
STRUCT
    Year                : DINT;
    Month               : DINT;
    Day                 : DINT;
    Hour                : DINT;
    Minute              : DINT;
    Second              : DINT;
    mSec                : DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.1.2 ST_PMLEvent

Collection of tags for describing alarm events.

```
TYPE ST_PMLEvent :
STRUCT
    Trigger             : BOOL;
    Id                  : DINT;
    Value               : DINT;
    Message             : STRING;
    Category            : DINT;

    DateTime            : ST_PMLDateAndTime;
    AckDateTime         : ST_PMLDateAndTime;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.1.3 ST_PMLEventMin

Collection of the minimum tags required for describing alarm events.

```
TYPE ST_PMLEventMin :
STRUCT
    Id                  : DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2 Common

5.3.2.1 ST_PMLCumulativeTimes

Collection of tags for recording times in the various states and UnitModes in the machine.

By using FB_PMLAdminTime [▶ 33], these are filled automatically and can be reset using the M_ResetCumulativeTime() method of the FB.

```
TYPE ST_PMLCumulativeTimes :
STRUCT
    AccTimeSinceReset   : DINT;
    ModeStateTimes      : ARRAY[0..cMaxDefinedUnitMode-1] OF ST_PMLModeStateTimes;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.2 ST_PMLEquipment

Collection of tags for signaling problems in the downstream (Blocked) or upstream (Starved) of the machine.

```
TYPE ST_PMLEquipment :
STRUCT
    Blocked             : BOOL;
    Starved             : BOOL;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.3 ST_PMLIngredients

Collection of tags for describing the raw materials required for the product.

```
TYPE ST_PMLIngredients :
STRUCT
    Parameter_REAL              : ARRAY[0..
cMaxRecipeIngredientsParametersReal-1] OF ST_PMLParameterReal;
    Parameter_STRING            : ARRAY[0..
cMaxRecipeIngredientsParametersString-1] OF ST_PMLParameterString;
    Parameter_LREAL             : ARRAY[0..
cMaxRecipeIngredientsParametersLreal-1] OF ST_PMLParameterLreal;
    Parameter_DINT              : ARRAY[0..
cMaxRecipeIngredientsParametersDint-1] OF ST_PMLParameterDint;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.4 ST_PMLModeStateTimes

Collection of tags for recording times in the various states and UnitModes in the machine.

```
TYPE ST_PMLModeStateTimes :
STRUCT
    Mode                : DINT;
    State               : ARRAY [0..cMaxMachineState] OF DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.5 ST_PMLParameterDint

Collection of tags for describing DINT parameters in the machine.

```
TYPE ST_PMLParameterDint :
STRUCT
    Id                  : DINT;
    Name                : STRING;
    Unit                : STRING(6);
    Value               : DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.6 ST_PMLParameterLreal

Collection of tags for describing LREAL parameters in the machine.

```
TYPE ST_PMLParameterLreal :
STRUCT
    Id                  : DINT;
    Name                : STRING;
    Unit                : STRING(6);
    Value               : LREAL;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.7 ST_PMLParameterReal

Collection of tags for describing REAL parameters in the machine.

```
TYPE ST_PMLParameterReal :
STRUCT
    Id                  : DINT;
    Name                : STRING;
    Unit                : STRING(6);
```

```
    Value               : REAL;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.8 ST_PMLParameterString

Collection of tags for describing STRING parameters in the machine.

```
TYPE ST_PMLParameterString :
STRUCT
    Id                  : DINT;
    Name                : STRING;
    Unit                : STRING(6);
    Value               : STRING;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.9 ST_PMLProcessVariables

Collection of tags for describing process variables in the machine.

```
TYPE ST_PMLProcessVariables :
STRUCT
    Parameter_REAL             : ARRAY[0..cMaxRecipeProcessVariablesParametersReal-1] OF
ST_PMLParameterReal;
    Parameter_STRING           : ARRAY[0..cMaxRecipeProcessVariablesParametersString-1] OF
ST_PMLParameterString;
    Parameter_LREAL            : ARRAY[0..cMaxRecipeProcessVariablesParametersLreal-1] OF
ST_PMLParameterLreal;
    Parameter_DINT             : ARRAY[0..cMaxRecipeProcessVariablesParametersDint-1] OF
ST_PMLParameterDint;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.10 ST_PMLProductData

Collection of tags for describing the product manufactured on the machine.

```
TYPE ST_PMLProductData :
STRUCT
    ID                      : DINT;
    Name                    : STRING;
    Unit                    : STRING[6];
    PrimaryQty              : DINT;
    ConsumedCount           : DINT;
    ProcessedCount          : DINT;
    DefectiveCount          : DINT;
```

```
    AccConsumedCount          : DINT;
    AccProcessedCount         : DINT;
    AccDefectiveCount         : DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.11 ST_PMLProductDataMin

Collection of all tags that are at least necessary for describing the product manufactured on the machine.

```
TYPE ST_PMLProductDataMin:
STRUCT
    ProcessedCount            : DINT;
    DefectiveCount            : DINT;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.2.12 ST_PMLRecipe

Collection of tags for describing recipes in the machine.

```
TYPE ST_PMLRecipe :
STRUCT
    Id                    : DINT;
    Name                  : STRING;
    Unit                  : STRING(6);
    PrimaryQty            : REAL;
    ProcessVariables      : ST_PMLProcessVariables;
    Ingredients           : ST_PMLIngredients;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.3 ST_PMLa

Collection of all Administration tags of the PackTag structure.

```
TYPE ST_PMLa :
STRUCT
    Parameter_REAL                  : ARRAY [0..cMaxAdminParametersReal-1] OF ST_PMLParameterReal;
    Parameter_STRING                : ARRAY [0..cMaxAdminParametersString-1] OF ST_PMLParameterStr
ing;
    Parameter_LREAL                 : ARRAY [0..cMaxAdminParametersLreal-1] OF ST_PMLParameterLrea
l;
    Parameter_DINT                  : ARRAY [0..cMaxAdminParametersDint-1] OF ST_PMLParameterDint;

    Alarm                           : ARRAY [0..cMaxAlarms-1] OF ST_PMLEvent;
```

```
    AlarmExtent                         : DINT := cMaxAlarms;
    AlarmHistory                        : ARRAY [0..cMaxHistoryAlarms-1] OF ST_PMLEvent;
    AlarmHistoryExtent                  : DINT := cMaxHistoryAlarms;
    StopReason                          : ST_PMLEvent;
    Warning                             : ARRAY [0..cMaxWarnings-1] OF ST_PMLEvent;
    WarningExtent                       : DINT := cMaxWarnings;

    ModeTimeCurrent                     : DINT;
    StateTimeCurrent                    : DINT;
    CumulativeTimes                     : ARRAY [0..cMaxAdminCumulativeTimes-1] OF ST_PMLCumulativeTim
es;

    ProductData                         : ARRAY [0..cMaxAdminProductData-1] OF ST_PMLProductData;

    MachDesignSpeed                     : REAL;
    DisabledStatesCfg                   : ARRAY [0..cMaxDefinedUnitMode-1] OF DWORD;
    CurDisabledStates                   : DWORD;
    EnabledModesCfg                     : DWORD;
    ModeTransitionCfg                   : ARRAY [0..cMaxDefinedUnitMode-1] OF DWORD;

    PlcDateTime                         : ST_PMLDateAndTime;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.4 ST_PMLaMin

Collection of all minimum necessary Administration tags of the PackTag structure.

```
TYPE ST_PMLaMin :
STRUCT
    StopReason                          : ST_PMLEventMin;

    ProductData                         : ARRAY [0..cMaxAdminProductData-1] OF ST_PMLProductDataMin;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.5 ST_PMLc

Collection of all Command tags of the PackTag structure.

```
TYPE ST_PMLc :
STRUCT
    UnitMode                : DINT;
    UnitModeChangeRequest   : BOOL;

    MachSpeed               : REAL;

    MaterialInterlock       : DWORD;

    CntrlCmd                : DINT;
    CmdChangeRequest        : BOOL;

    Parameter_REAL          : ARRAY [0..cMaxCommandParametersReal-1] OF ST_PMLParameterReal;
    Parameter_STRING        : ARRAY [0..cMaxCommandParametersString-1] OF ST_PMLParameterString;
    Parameter_LREAL         : ARRAY [0..cMaxCommandParametersLreal-1] OF ST_PMLParameterLreal;
    Parameter_DINT          : ARRAY [0..cMaxCommandParametersDint-1] OF ST_PMLParameterDint;
```

```
    SelectedRecipe          : DINT;
    RecipeChangeRequest     : BOOL;

    Recipe                  : ARRAY [0.. cMaxCommandRecipes-1] OF ST_PMLRecipe;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.6 ST_PMLcMin

Collection of all minimum necessary Command tags of the PackTag structure.

```
TYPE ST_PMLcMin :
STRUCT
    UnitMode                : DINT;
    UnitModeChangeRequest   : BOOL;

    MachSpeed               : REAL;

    CntrlCmd                : DINT;
    CmdChangeRequest        : BOOL;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.7 ST_PMLs

Collection of all State tags of the PackTag structure.

```
TYPE ST_PMLs :
STRUCT
    UnitModeCurrent           : DINT;
    UnitModeRequested         : DINT;
    UnitModeChangeInProcess   : BOOL;

    StateCurrent              : DINT;
    StateRequested            : DINT;
    StateChangeInProcess      : BOOL;

    MachSpeed                 : REAL;
    CurMachSpeed              : REAL;

    MaterialInterlock         : DWORD;
    EquipmentInterlock        : ST_PMLEquipment;

    Parameter_REAL            : ARRAY [0..cMaxStatusParametersReal-1] OF ST_PMLParameterReal;
    Parameter_STRING          : ARRAY [0..cMaxStatusParametersString-1] OF ST_PMLParameterString;
    Parameter_LREAL           : ARRAY [0..cMaxStatusParametersLreal-1] OF ST_PMLParameterLreal;
    Parameter_DINT            : ARRAY [0..cMaxStatusParametersDint-1] OF ST_PMLParameterDint;

    RecipeCurrent             : DINT;
    RecipeRequested           : DINT;
    RecipeChangeInProcess     : BOOL;

    Recipe                    : ARRAY [0..cMaxStatusRecipes-1] OF ST_PMLRecipe;

    Stacklight                : ARRAY [0..cMaxStatusStacklights-1] OF DWORD;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.8 ST_PMLsMin

Collection of all minimum necessary State tags of the PackTag structure.

```
TYPE ST_PMLsMin :
STRUCT
    UnitModeCurrent           : DINT;

    StateCurrent              : DINT;

    MachSpeed                 : REAL;
    CurMachSpeed              : REAL;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.9 ST_PMLV2022

Collection of all tags in the PackTag structure.

```
TYPE ST_PMLV2022 :
STRUCT
    Command             : ST_PMLc;
    Status              : ST_PMLs;
    Admin               : ST_PMLa;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

5.3.10 ST_PMLV2022Min

Collection of all minimum required tags of the PackTag structure.

```
TYPE ST_PMLV2022Min :
STRUCT
    Command             : ST_PMLcMin;
    Status              : ST_PMLsMin;
    Admin               : ST_PMLaMin;
END_STRUCT
END_TYPE
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

6 Global parameters

Parameters for setting up the packaging machine tag structures. These can be adapted when inserting the library for the current project.

```
VAR_GOLBAL CONSTANT
    cMaxDefinedUnitMode                         : UDINT := 4;

    (* PMLc/PMLa *)
    cMaxRecipeProcessVariablesParametersReal    : UDINT := 10;
    cMaxRecipeProcessVariablesParametersString  : UDINT := 10;
    cMaxRecipeProcessVariablesParametersLreal   : UDINT := 10;
    cMaxRecipeProcessVariablesParametersDint    : UDINT := 10;

    cMaxRecipeIngredientsParametersReal         : UDINT := 10;
    cMaxRecipeIngredientsParametersString       : UDINT := 10;
    cMaxRecipeIngredientsParametersLreal        : UDINT := 10;
    cMaxRecipeIngredientsParametersDint         : UDINT := 10;

    (* PMLc *)
    cMaxCommandParametersReal                   : UDINT := 10;
    cMaxCommandParametersString                 : UDINT := 10;
    cMaxCommandParametersLreal                  : UDINT := 10;
    cMaxCommandParametersDint                   : UDINT := 10;

    cMaxCommandRecipes                          : UDINT := 5;

    (* PMLs *)
    cMaxStatusParametersReal                    : UDINT := 10;
    cMaxStatusParametersString                  : UDINT := 10;
    cMaxStatusParametersLreal                   : UDINT := 10
    cMaxStatusParametersDint                    : UDINT := 10;

    cMaxStatusRecipes                           : UDINT := 5;

    cMaxStatusStacklights                       : UDINT := 1;

    (* PMLa *)
    cMaxAdminParametersReal                     : UDINT := 10;
    cMaxAdminParametersString                   : UDINT := 10;
    cMaxAdminParametersLreal                    : UDINT := 10;
    cMaxAdminParametersDint                     : UDINT := 10;

    cMaxAlarms                                  : DINT := 10;
    cMaxHistoryAlarms                           : DINT := 10;
    cMaxWarnings                                : DINT := 10;

    cMaxAdminCumulativeTimes                    : UDINT := 1;
    cMaxAdminProductData                        : UDINT := 10;

    cMaxConsumedCounts                          : UDINT := 10;
    cMaxProductCounts                           : UDINT := 10;
END_VAR
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

7 Global constants / variables

Constants for setting up the packaging machine tag structures. These cannot be changed.

```
VAR_GLOBAL CONSTANT
    cMaxUnitMode                : INT := 31
    cMaxMachineState            : INT := 17
END_VAR
```

The information about the permitted states and transitions configured with the PML_UnitModeConfig [▶ 38] is stored in this structure.

```
VAR_GLOBAL
    stPMLUnitModeConfiguration  : ARRAY[0..cMaxUnitMode] OF ST_PMLUnitModeConfiguration
END_Var
```

Requirements

Development Environment	Target platform	PLC library to include
From TwinCAT • 3.1 Build 4024.63	PC (i386)	From Tc3_PackML_ V3 1.0.3

8 Interfaces

Definition of interfaces, that can be used for PackML handling.

8.1 I_PMLUnitStateActing

This interface can be implemented in the unit function blocks of the application and only provides the "Acting" methods of the Packaging State Model, which can then be filled with application code as required.

These methods are:

M_Aborting

M_Clearing

M_Completing

M_Execute

M_Holding

M_Resetting

M_Starting

M_StateComplete

M_Stopping

M_Suspending

M_Unholding

M_Unsuspending

8.2 I_PMLUnitStateWaiting

This interface can be implemented in the unit function blocks of the application and only provides the "Waiting" methods of the Packaging State Model, which can then be filled with application code as required.

These methods are:

M_Aborted

M_Completed

M_Held

M_Idle

M_Stopped

M_Suspended

M_Undefined

9 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Download finder

Our download finder contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963-157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963-460
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963-0
e-mail:	info@beckhoff.com
web:	www.beckhoff.com

Trademark statements

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

More Information:
www.beckhoff.com/te1000

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