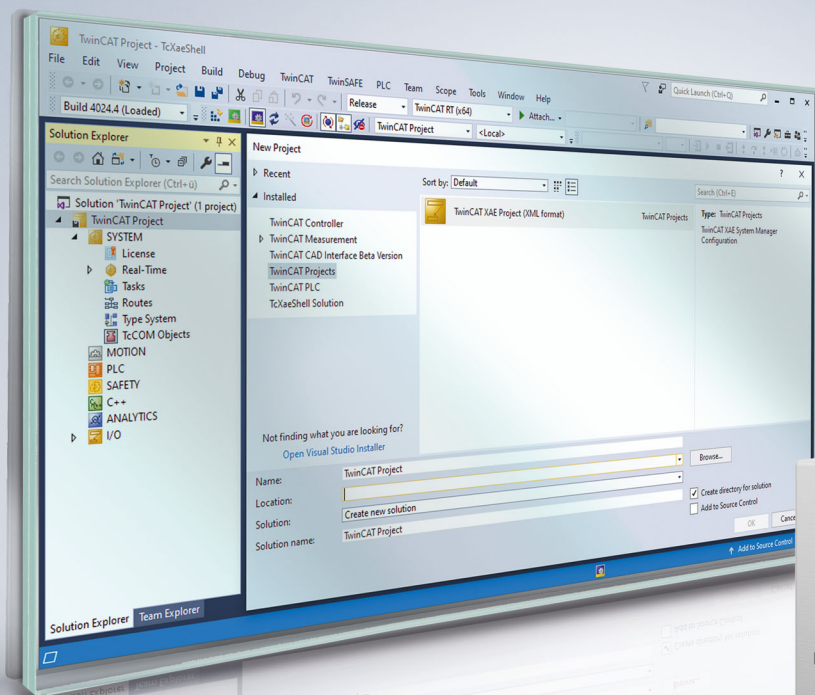


BECKHOFF New Automation Technology

Handbuch | DE

TE1000

TwinCAT 3 | PLC-Bibliothek: Tc3_PackML_V3



Inhaltsverzeichnis

1	Vorwort.....	5
1.1	Hinweise zur Dokumentation	5
1.2	Zu Ihrer Sicherheit.....	6
1.3	Hinweise zur Informationssicherheit	7
2	Packaging Machine State	8
3	Packaging Machine Tags	10
3.1	Tag-Arten	10
3.2	Tag-Details	11
4	Funktionsbausteine	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	23
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	Datentypen.....	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	Allgemein	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 Globale Parameter 57

7 Globale Konstanten / Variablen 58

8 Interfaces 59

 8.1 I_PMLUnitStateActing 59

 8.2 I_PMLUnitStateWaiting 59

9 Support und Service 60

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.

Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.

Das Fachpersonal ist verpflichtet, stets die aktuell gültige Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiterentwickelt.

Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.

Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® und XTS® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.

Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Kennzeichnungen führen.



EtherCAT® ist eine eingetragene Marke und patentierte Technologie, lizenziert durch die Beckhoff Automation GmbH, Deutschland.

Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.

Zuwiderhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

Fremdmarken

In dieser Dokumentation können Marken Dritter verwendet werden. Die zugehörigen Markenvermerke finden Sie unter: <https://www.beckhoff.com/trademarks>.

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.

Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt



Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem <https://www.beckhoff.de/secguide>.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

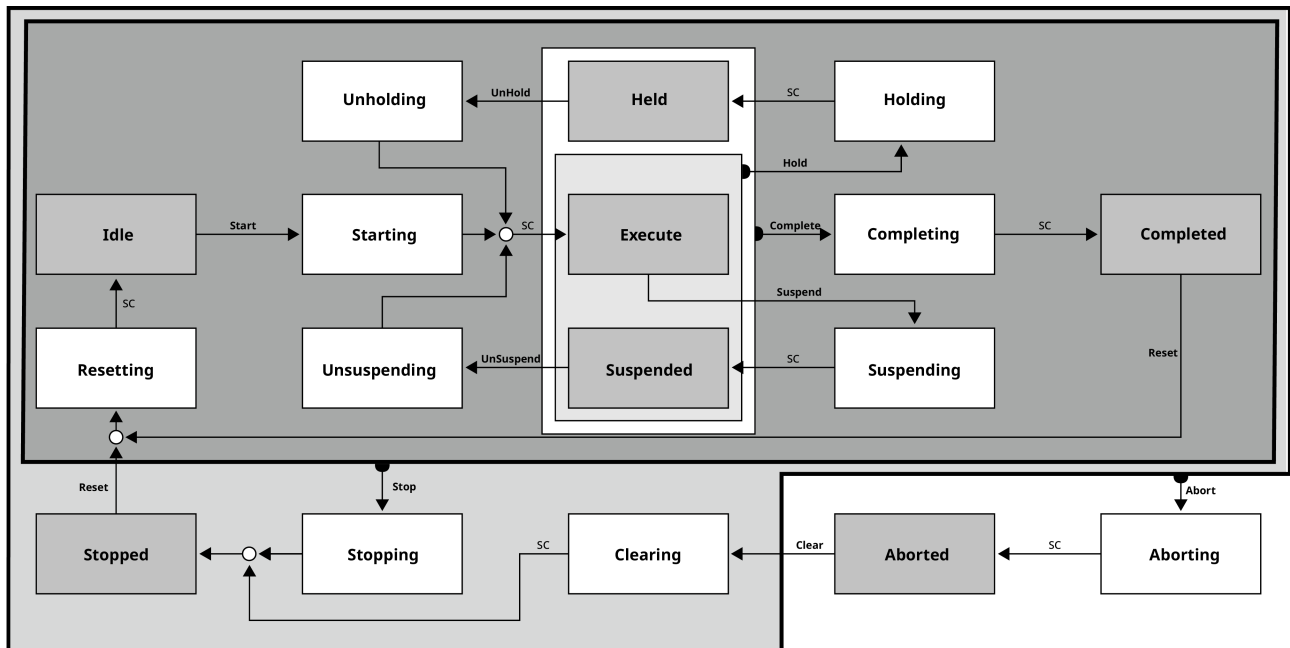
Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter <https://www.beckhoff.de/secinfo>.

2 Packaging Machine State

Die „Packaging Machine State“-Funktionsbausteine haben eine gemeinsame Schnittstelle zu den existierenden „PackML Machine State Model“-Ausführungen.

Es wird erwartet, dass anwendungsspezifische Logik, wie Zustandsübergänge, in externen Funktionsbausteinen programmiert wird und der „Packaging Machine State“-Funktionsbaustein die zentrale Logik der Zustandsmaschine und die Zustandsdarstellung übernimmt. Deswegen gibt es für diesen Funktionsbaustein eine Empfehlung, wie er mit anderer Logik kombiniert werden kann.

Der Zustandsübergang in einer Maschinenanwendung ist immer anwendungsspezifisch. Deswegen implementieren Sie am besten mit der PackML State Machine V3 verknüpfte Methoden in ihren Funktionsbausteinen, um die Standardisierung zu vereinfachen. Dazu können Sie die innerhalb dieser Bibliothek vorgegebenen Interfaces verwenden und dann die Methoden anwendungsspezifisch ausprogrammieren. Die „Acting“-Methoden erfassen anwendungsspezifische Signale und stellen die Übergangslogik zu angrenzenden Zuständen dar (siehe PackML Statemodell unten). Die „Waiting“-Methoden geben Feedback an die übergeordnete PackML State Machine V3, wodurch eine Standard-Zustandsmaschine und Zustandsmeldung möglich wird. Die Methoden enthalten den Maschinenausführungscode und die anwendungsspezifische Übergangslogik.



Die verfügbaren Methoden sind unten gelistet und werden so programmiert, dass Integrität und Funktionalität der PackML State Machine gewahrt bleiben.

Namen der „PackML State Machine V3“ Acting-Methoden:

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

Namen der „PackML State Machine V3“ Waiting-Methoden:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

3 Packaging Machine Tags

PackTags stellen einen einheitlichen Satz Namenskonventionen für Datenelemente bereit, die in den prozeduralen Elementen des Basiszustandsmodells verwendet werden. Wie beschrieben, stellt das Base State Model (Basiszustandsmodell) einen einheitlichen Satz Maschinenzustände bereit, sodass alle automatisierten Maschinen gleich betrachtet werden können. PackTags sind mit Namen versehene Datenelemente für interoperablen Datenaustausch bei automatisierten Maschinen offener Architekturen. In dieser Dokumentation finden Sie die wesentlichen Namen der Datenelemente, Datentyp, Werte, Bereiche und gegebenenfalls Datenstrukturen. PackTags werden für die Maschine-zu-Maschine-Kommunikation eingesetzt, z. B. zwischen einem Abfüller und einem Kappenaufsetzer. PackTags können auch für den Datenaustausch zwischen Maschine und übergeordnetem Informationssystem wie Manufacturing Operations Management und Enterprise Information Systemen eingesetzt werden.

Diese Dokumentation beschreibt alle PackTags für die Navigation durch ein Zustandsmodell und für die Definition und Betätigung des Anlagensteuerungsmodus. Des Weiteren definiert diese Dokumentation eine Liste von PackTags, die mögliche wichtige Informationen einer Maschine bereitstellen. Alle PackTags müssen genutzt werden, um den Prinzipien integrierter Konnektivität mit Systemen mit der gleichen Implementierungsmethode zu entsprechen.

Notwendig sind die Tags, die für die Funktion der automatisierten Maschine oder die Konnektivität zu Kontroll- oder Fernsystemen benötigt werden.

3.1 Tag-Arten

PackTags werden in drei Gruppen aufgegliedert: Command (Befehl), Status (Zustand) und Administration (Verwaltung). Befehl- und Zustand-Tags enthalten Daten für die Anbindung der Maschine an die Liniensteuerung zur Koordination oder zum Herunterladen von Rezepten/Parametern. Befehl-Tags werden als „Informationsempfänger“ an das Maschinenprogramm „übergeben“ und von ihm konsumiert, Zustand-Tags werden vom Maschinenprogramm erzeugt und ausgelesen. Verwaltung-Tags enthalten Daten, die übergeordnete Systeme zur Analyse der Maschinenleistung oder zur Information der Bediener sammeln.

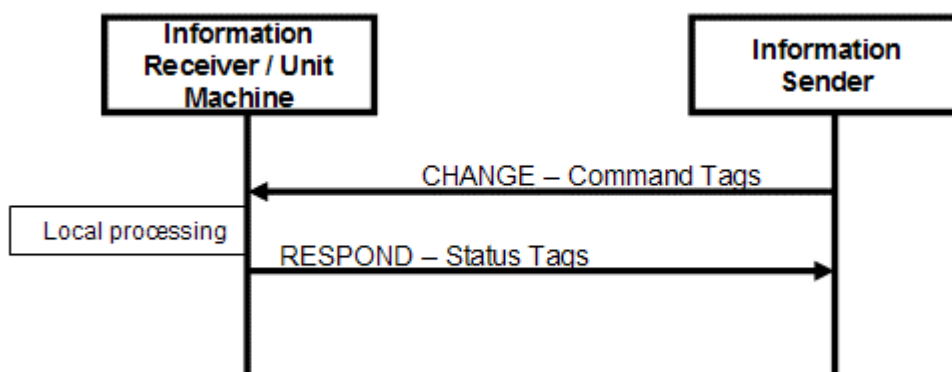
Die Gruppierung von Daten sollte in benachbarten Registern erfolgen, um die Kommunikation zu optimieren.

Normalerweise werden Informationsdaten via OPC in einem Ethernet-basierten Kommunikationsnetzwerk übertragen.

Präfix von Befehl-Tags ist „PMLc“.

Präfix von Zustand-Tags ist „PMLs“.

Präfix von Verwaltung-Tags ist „PMLa“.



3.2 Tag-Details

Der folgende Abschnitt gibt einen Überblick über die Tags. In den nachfolgenden Tabellen sind Befehl-, Zustand- und Verwaltung-PackTags aufgelistet.

Befehlsstruktur 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

Zustandsstruktur Status

					Tag Name	Data Type
Status					Status	ST_PMLs
	UnitModeCurrent				Status.UnitModeCurrent	DINT
	UnitModeRequested				Status.UnitModeRequested	DINT
	UnitModeChangeInProcesses				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

Verwaltungsstruktur 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[#].ConsumedCount	DINT
		ProcessedCount		Admin.ProductData[#].ProcessCount	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_PMLDateA ndTime
		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 Funktionsbausteine

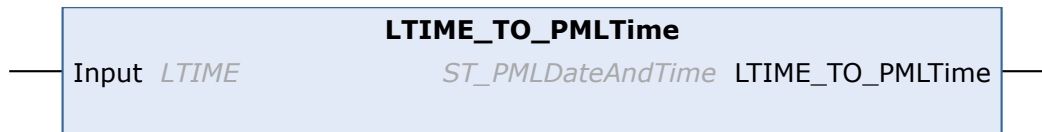
4.1 Conversion

Hier befinden sich Funktionen zur Konvertierung von Zeiten bzw. Zeitstempeln im TwinCAT-Format in das PackML-Format, sowie Konvertierung eines PackML UnitMode oder PackML State (beides DINT) in einen entsprechenden STRING.

4.1.1 Time

Diese Funktionen konvertieren Zeitwerte in das PackML-konforme Format.

4.1.1.1 LTIME_TO_PMLTime



Die Funktion LTIME_TO_PMLTime konvertiert einen Zeitwert im LTIME-Format in das PackML-konforme Format.

FUNCTION LTIME_TO_PMLTime : ST_PMLDateAndTime

Eingang

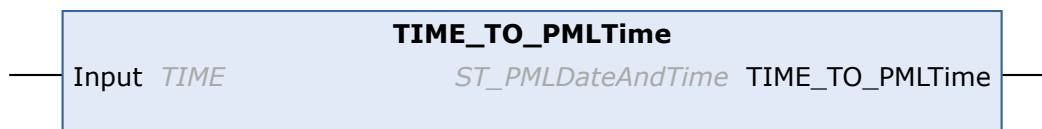
```
VAR_INPUT
    Input      : LTIME;
END_VAR
```

Name	Typ	Beschreibung
Input	LTIME	Der zu konvertierende Zeitwert

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.1.2 TIME_TO_PMLTime



Die Funktion TIME_TO_PMLTime konvertiert einen Zeitwert im TIME-Format in das PackML-konforme Format.

FUNCTION TIME_TO_PMLTime : ST_PMLDateAndTime

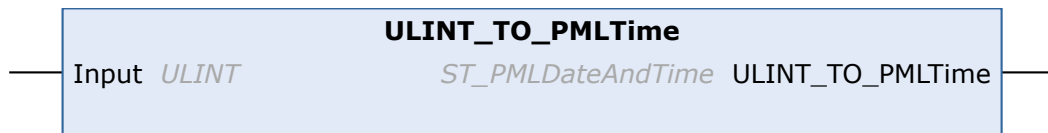
Eingang

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

Name	Typ	Beschreibung
Input	TIME	Der zu konvertierende Zeitwert

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.1.3 ULINT_TO_PMLTime

Die Funktion ULINT_TO_PMLTime konvertiert einen Zeitwert im ULINT-Format in das PackML-konforme Format.

FUNCTION ULINT_TO_PMLTime : ST_PMLDateAndTime **Eingang**

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

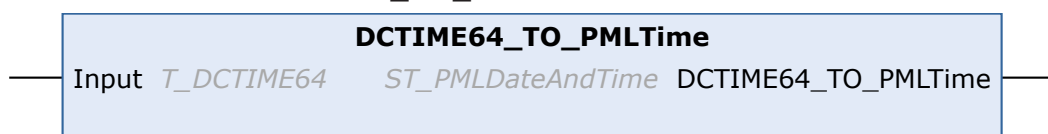
Name	Typ	Beschreibung
Input	ULINT	Der zu konvertierende Zeitwert

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.2 Timestamp

Diese Funktionen konvertieren Zeitstempel in das PackML-konforme Format

4.1.2.1 DCTIME64_TO_PMLTime

Die Funktion DCTIME64_TO_PMLTime konvertiert einen Zeitpunkt im DCTIME64-Format in das PackML-konforme Format.

FUNCTION DCTIME64_TO_PMLTime : ST_PMLDateAndTime **Eingang**

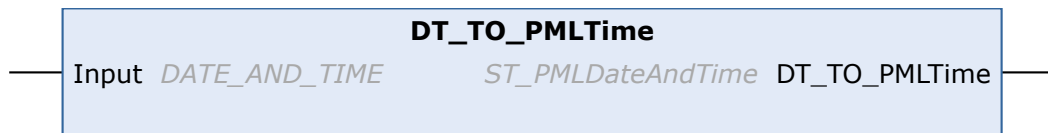
```
VAR_INPUT
    Input      : DCTIME64;
END_VAR
```

Name	Typ	Beschreibung
Input	DCTIME64	Der zu konvertierende Zeitpunkt

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.2.2 DT_TO_PMLTime



Die Funktion DT_TO_PMLTime konvertiert einen Zeitpunkt im DT-Format in das PackML-konforme Format.

FUNCTION DT_TO_PMLTime : ST_PMLDateAndTime

Eingang

```

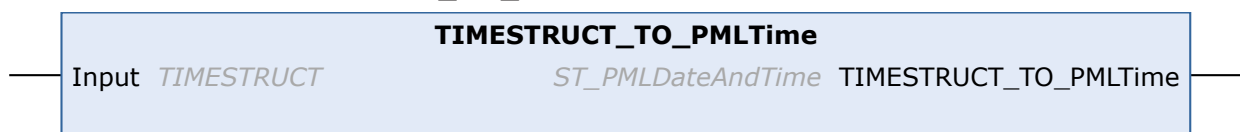
VAR_INPUT
    Input      : DATE_AND_TIME;
END_VAR
  
```

Name	Typ	Beschreibung
Input	DATE_AND_TIME	Der zu konvertierende Zeitpunkt

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.2.3 TIMESTRUCT_TO_PMLTime



Die Funktion TIMESTRUCT_TO_PMLTime konvertiert einen Zeitpunkt im TIMESTRUCT-Format in das PackML-konforme Format.

FUNCTION TIMESTRUCT_TO_PMLTime : ST_PMLDateAndTime

Eingang

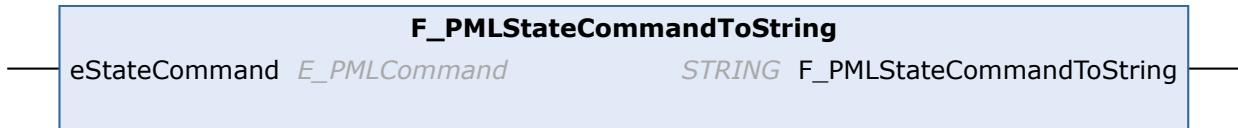
```

VAR_INPUT
    Input      : TIMESTRUCT;
END_VAR
  
```

Name	Typ	Beschreibung
Input	TIMESTRUCT	Der zu konvertierende Zeitpunkt

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.3 F_PMLStateCommandToString

Die Funktion F_PMLStateCommandToString gibt den Namen eines State-Kommandos als Zeichenkette aus.

FUNCTION F_PMLStateCommandToString : STRING;

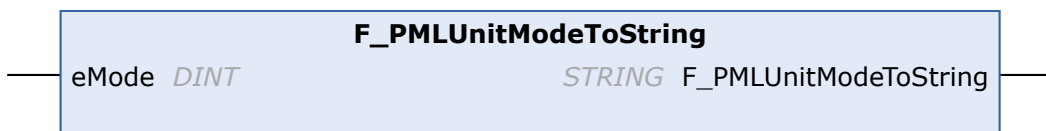
Eingang

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

Name	Typ	Beschreibung
eStateCommand	E_PMLCommand	Das State-Kommando, zu dem der Name ermittelt werden soll.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.1.4 F_PMLUnitModeToString

Die Funktion F_PMLUnitModeToString liefert den Namen eines Unit Mode als Zeichenkette zurück.

FUNCTION F_PMLUnitModeToString : STRING;

Eingang

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

Name	Typ	Beschreibung
eMode	DINT	Der Unit Mode, zu dem der Name ermittelt werden soll.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2 General

Allgemeine Funktionsbausteine zur Vereinfachung des Umgangs mit den PackML Strukturen

4.2.1 FB_PMLAdminAlarm

Dieser Funktionsbaustein unterstützt beim Eintragen, Quittieren und Löschen von Alarm, Warning und StopReason der Admin-PackTags. Dazu stellt der Funktionsbaustein verschiedene Methoden bereit.

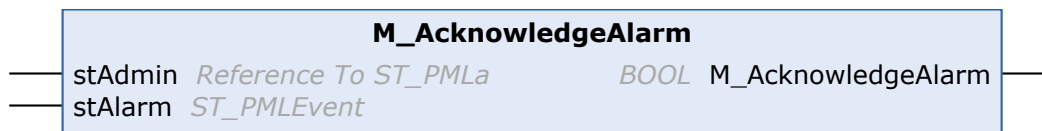
Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1 Alarm

Methoden zum Handling von Alarmen

4.2.1.1.1 M_AcknowledgeAlarm



Diese Methode quittiert einen Alarm in den Admin-Tags. Alarm[].Trigger wird FALSE gesetzt und in Alarm[].AckDateTime der Wert aus Admin.PlcDateTime eingetragen. Konnte der Alarm erfolgreich gefunden und quittiert werden liefert die Methode TRUE zurück. Die Quittierung des Alarms löscht diesen nicht. Der Alarm verbleibt im Alarm-Array bis auch ein Aufruf M_ClearAlarm erfolgt ist, dann wird dieser in das AlarmHistory-Array verschoben. Sollte das AlarmHistory-Array bereits mit Einträgen gefüllt sein wird der älteste Eintrag dadurch gelöscht.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

```

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.2 M_AcknowledgeAllAlarms

M_AcknowledgeAllAlarms

— stAdmin *Reference To ST_PMLa* *BOOL* M_AcknowledgeAllAlarms —

Diese Methode quittiert alle Alarme in den Admin-Tags. Alarm[].Trigger wird FALSE gesetzt und in Alarm[].AckDateTime der Wert aus Admin.PlcDateTime eingetragen. Sind alle Alarme quittiert worden liefert die Methode TRUE zurück. Die Quittierung der Alarme löscht diese nicht. Die Alarme verbleiben im Alarm-Array bis auch ein Aufruf M_ClearAlarm bzw. M_ClearAllAlarms erfolgt ist, dann werden die betreffenden Alarme in das AlarmHistory-Array verschoben. Sollte das AlarmHistory-Array bereits mit Einträgen gefüllt sein werden die ältesten Einträge dadurch gelöscht.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.3 M_ClearAlarm

M_ClearAlarm

— stAdmin *Reference To ST_PMLa* *BOOL* M_ClearAlarm —
— stAlarm *ST_PMLEvent*

Diese Methode löscht einen Alarm in den Admin-Tags. Alarm[].Trigger wird FALSE gesetzt. Konnte der Alarm erfolgreich gelöscht werden liefert die Methode TRUE zurück. Der Alarm verbleibt im Alarm-Array bis auch ein Aufruf M_AcknowledgeAlarm erfolgt ist, dann wird dieser in das AlarmHistory-Array verschoben. Sollte das AlarmHistory-Array bereits mit Einträgen gefüllt sein wird der älteste Eintrag dadurch gelöscht.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.4 M_ClearAllAlarms**M_ClearAllAlarms**

— stAdmin *Reference To ST_PMLa* *BOOL* M_ClearAllAlarms —

Diese Methode löscht alle Alarm in den Admin-Tags. Alarm[].Trigger wird FALSE gesetzt. Konnten die Alarmer erfolgreich gelöscht werden liefert die Methode TRUE zurück. Die Alarmer verbleiben im Alarm-Array bis auch ein Aufruf M_AcknowledgeAlarm bzw. M_AcknowledgeAllAlarms erfolgt ist, dann werden diese in das AlarmHistory-Array verschoben. Sollte das AlarmHistory-Array bereits mit Einträgen gefüllt sein werden die ältesten Einträge dadurch gelöscht.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.5 M_GetAlarmCategory**M_GetAlarmCategory**

— stAdmin *Reference To ST_PMLa* *DINT* M_GetAlarmCategory —

Diese Methode liefert die höchste AlarmCategory (kleinster Wert) in den Alarmen der Admin-Tags zurück.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.6 M_HasAlarm

M_HasAlarm

— stAdmin *Reference To ST_PMLa* *BOOL* M_HasAlarm —

Diese Methode prüft, ob ein aktiver Alarm in den Admin-Tags eingetragen ist und die Methode liefert TRUE zurück

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.1.7 M_SetAlarm

M_SetAlarm

— stAdmin *Reference To ST_PMLa* *BOOL* M_SetAlarm —
— stAlarm *ST_PMLEvent* —

Diese Methode fügt einen Alarm in den Admin-Tags ein. Alarm[].Trigger wird TRUE gesetzt und in Alarm[].DateTime der Wert aus Admin.PlcDateTime eingetragen. Die anderen Werte werden aus der übergebenen Alarm-Struktur übernommen. Konnte der Alarm erfolgreich eingetragen werden liefert die Methode TRUE zurück



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.2 StopReason

Methoden zum Handling von Stopursachen

4.2.1.2.1 M_AcknowledgeStopReason

M_AcknowledgeStopReason

— stAdmin *Reference To ST_PMLa* *BOOL* M_AcknowledgeStopReason —

Diese Methode quittiert das StopReason in den Admin-Tags. StopReason.Trigger wird FALSE gesetzt und in StopReason.AckDateTime der Wert aus Admin.PlcDateTime eingetragen. Konnte das StopReason erfolgreich quittiert werden liefert die Methode TRUE zurück. Das StopReason verbleibt in den Admin-Tags, bis es durch nachfolgende StopReason verdrängt wird.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.2.2 M_ClearStopReason

M_ClearStopReason

— stAdmin *Reference To ST_PMLa* *BOOL* M_ClearStopReason —

Diese Methode löscht das StopReason in den Admin-Tags. StopReason.Trigger wird FALSE gesetzt. Konnte das StopReason erfolgreich gelöscht werden liefert die Methode TRUE zurück. Das StopReason verbleibt in den Admin-Tags, bis es durch nachfolgende StopReason verdrängt wird.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT	PC (i386)	Ab Tc3_PackML_V3 1.0.3

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
• 3.1 Build 4024.63		

4.2.1.2.3 M_HasStopReason

M_HasStopReason

— stAdmin *Reference To ST_PMLa* *BOOL* M_HasStopReason —

Diese Methode prüft, ob eine aktive StopReason in den Admin-Tags eingetragen ist und die Methode liefert TRUE zurück

Syntax

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

Beispielaufwurf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.2.4 M_SetStopReason

M_SetStopReason

— stAdmin *Reference To ST_PMLa* *BOOL* M_SetStopReason —
 — stStopReason *ST_PMLaEvent*

Diese Methode fügt ein StopReason in den Admin-Tags ein. StopReason[].Trigger wird TRUE gesetzt und in StopReason[].DateTime der Wert aus Admin.PlcDateTime eingetragen. Die anderen Werte werden aus der übergebenen StopReason-Struktur übernommen. Konnte das StopReason erfolgreich eingetragen werden liefert die Methode TRUE zurück. Sollte das StopReason-Array bereits mit Einträgen gefüllt sein wird der älteste Eintrag dadurch gelöscht.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

Beispielaufwurf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3 Warning

Methoden zum Handling von Warnungen

4.2.1.3.1 M_AcknowledgeAllWarning**M_AcknowledgeAllWarning**

— stAdmin *Reference To ST_PMLa* *BOOL* M_AcknowledgeAllWarning —

Diese Methode quittiert alle Warning in den Admin-Tags. Warning[].Trigger wird FALSE gesetzt und in Warning[].AckDateTime der Wert aus Admin.PlcDateTime eingetragen. Sind alle Warning quittiert, liefert die Methode TRUE zurück. Die Warning verbleiben im Warning-Array, bis sie durch nachfolgende Warning aus dem Array verdrängt werden.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3.2 M_AcknowledgeWarning**M_AcknowledgeWarning**

— stAdmin *Reference To ST_PMLa* *BOOL* M_AcknowledgeWarning —
 — stWarning *ST_PMLEvent*

Diese Methode quittiert ein Warning in den Admin-Tags. Warning[].Trigger wird FALSE gesetzt und in Warning[].AckDateTime der Wert aus Admin.PlcDateTime eingetragen. Konnte das Warning erfolgreich gefunden und quittiert werden liefert die Methode TRUE zurück. Das Warning verbleibt im Warning-Array, bis es durch nachfolgende Warning aus dem Array verdrängt wird.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

```

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

```

Beispielaufruf:

```

WarningAcknowledged := fbAdminAlarm.M_AcknowledgeWarning(stAdmin := PackTags.Admin, stWarning := Warning);

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3.3 M_ClearAllWarning**M_ClearAllWarning**

— stAdmin *Reference To ST_PMLa* *BOOL* M_ClearAllWarning —

Diese Methode löscht alle Warning in den Admin-Tags. Warning[].Trigger wird FALSE gesetzt. Konnte das Warning erfolgreich gelöscht werden liefert die Methode TRUE zurück. Die Warning verbleiben im Warning-Array, bis sie durch nachfolgende Warning aus dem Array verdrängt werden.

Syntax

```

METHOD M_ClearAllWarning : BOOL
VAR_IN_OUT
    stAdmin          : ST_PMLa;
END_VAR

```

Beispielaufruf:

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3.4 M_ClearWarning**M_ClearWarning**

— stAdmin *Reference To ST_PMLa* *BOOL* M_ClearWarning —
 — stWarning *ST_PMLEvent*

Diese Methode löscht ein Warning in den Admin-Tags. Warning[].Trigger wird FALSE gesetzt. Konnte das Warning erfolgreich gelöscht werden liefert die Methode TRUE zurück. Das Warning verbleibt im Warning-Array, bis es durch nachfolgende Warning aus dem Array verdrängt wird.

Syntax

```

METHOD M_ClearWarning : BOOL
VAR_IN_OUT
    stAdmin          : ST_PMLa;
END_VAR

```

```
VAR_INPUT
    stWarning      : ST_PMLEvent;
END_VAR
```

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3.5 M_HasWarning

M_HasWarning

— stAdmin *Reference To ST_PMLa* *BOOL* M_HasWarning —

Diese Methode prüft, ob ein aktives Warning in den Admin-Tags eingetragen ist und die Methode liefert TRUE zurück

Syntax

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

Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.1.3.6 M_SetWarning

M_SetWarning

— stAdmin *Reference To ST_PMLa* *BOOL* M_SetWarning —
 — stWarning *ST_PMLEvent*

Diese Methode fügt ein Warning in den Admin-Tags ein. Warning[].Trigger wird TRUE gesetzt und in Warning[].DateTime der Wert aus Admin.PlcDateTime eingetragen. Die anderen Werte werden aus der übergebenen Warning-Struktur übernommen. Konnte das Warning erfolgreich eingetragen werden liefert die Methode TRUE zurück. Sollte das Warning-Array bereits mit Einträgen gefüllt sein wird der älteste Eintrag dadurch gelöscht.



Damit ein gültiger Zeitstempel eingetragen werden kann, sollte der Baustein PML_AdminTime zyklisch im Programm aufgerufen werden.

Syntax

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

```
VAR_INPUT
    stWarning      : ST_PMLEvent;
END_VAR
```

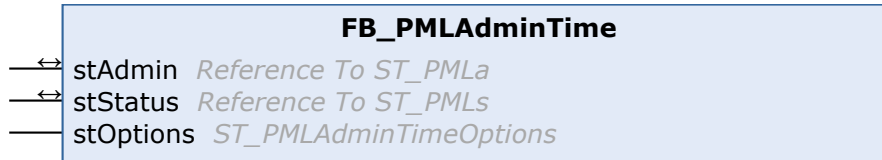
Beispielaufruf:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.2.2 FB_PMLAdminTime



Der Funktionsbaustein FB_PMLAdminTime sollte zyklisch aufgerufen werden, um die folgenden Admin-PackTags zu füllen:

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

Hiermit wird aufgezeichnet, für welche Zeit die Maschine sich in den verschiedenen States befunden hat. Dadurch können im Weiteren Rückschlüsse über die Maschineneffizienz gewonnen werden. Damit die Zeiten korrekt berechnet werden, wird vorausgesetzt, dass die Status-PackTags *UnitCurrent* und *StateCurrent* bereits sinnvoll beschrieben werden.

Ein Reset der einzelnen Timer ist über die Verwendung der Methode M_ResetCumulativeTime durchzuführen.

Eingang

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

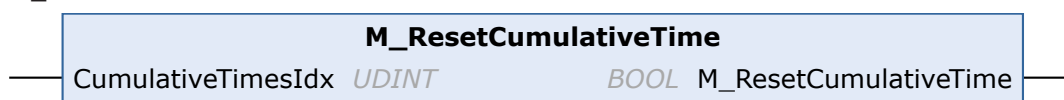
Name	Typ	Beschreibung
stOptions	ST_PMLAdminTimeOptions	Zusätzliche Optionen des Bausteins

Ein/Ausgänge

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

Name	Typ	Beschreibung
stAdmin	ST_PMLa	Übergabe der Admin-PackTags
stStatus	ST_PMLs	Übergabe der Status-PackTags

M_ResetCumulativeTime



Die Methode M_ResetCumulativeTime setzt den Eintrag Admin.CumulativeTimes[CumulativeTimesIdx] zurück.

Syntax

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

Beispielaufruf:

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

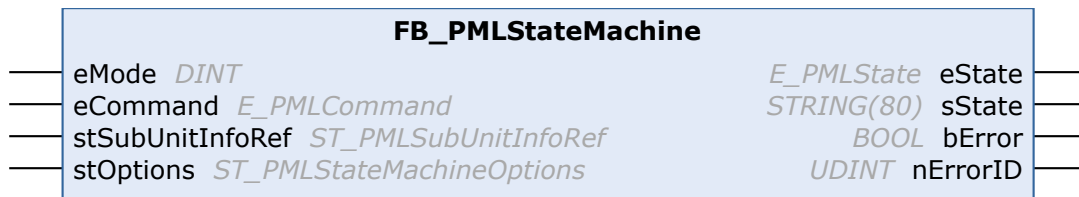
Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.3 Packaging Machine State

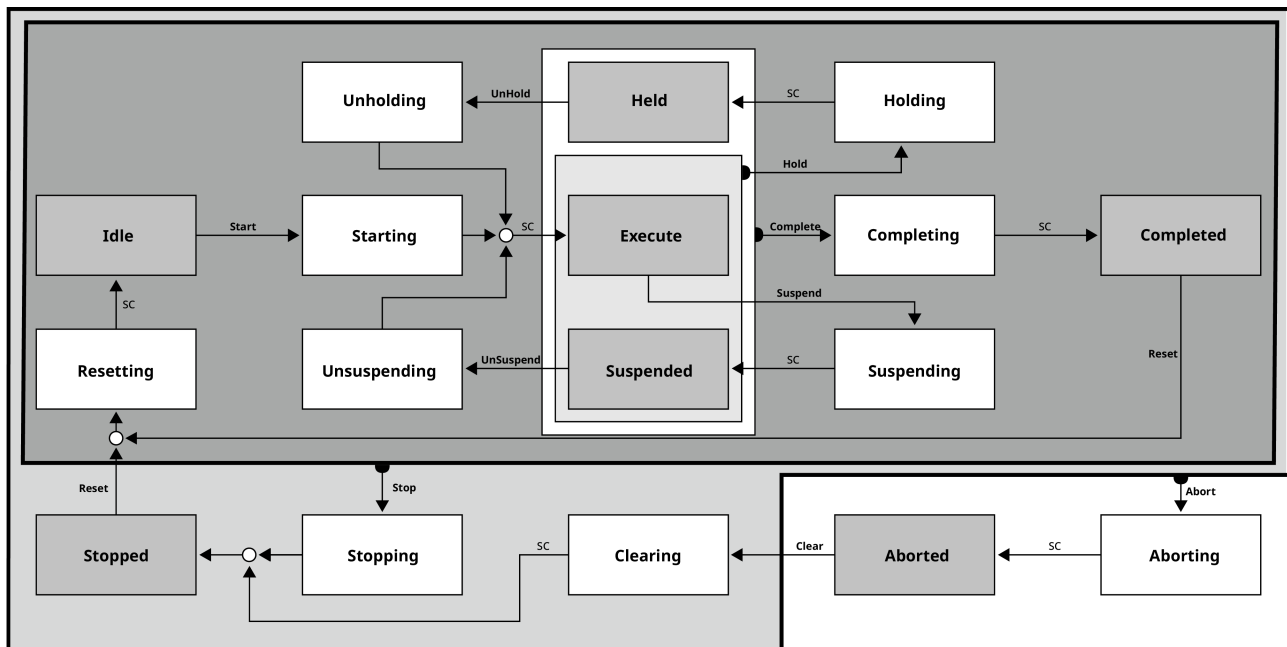
Funktionsbausteine zum vereinfachten Handling der PackML-Statemachine und -UnitModes

4.3.1 FB_PMLStateMachine

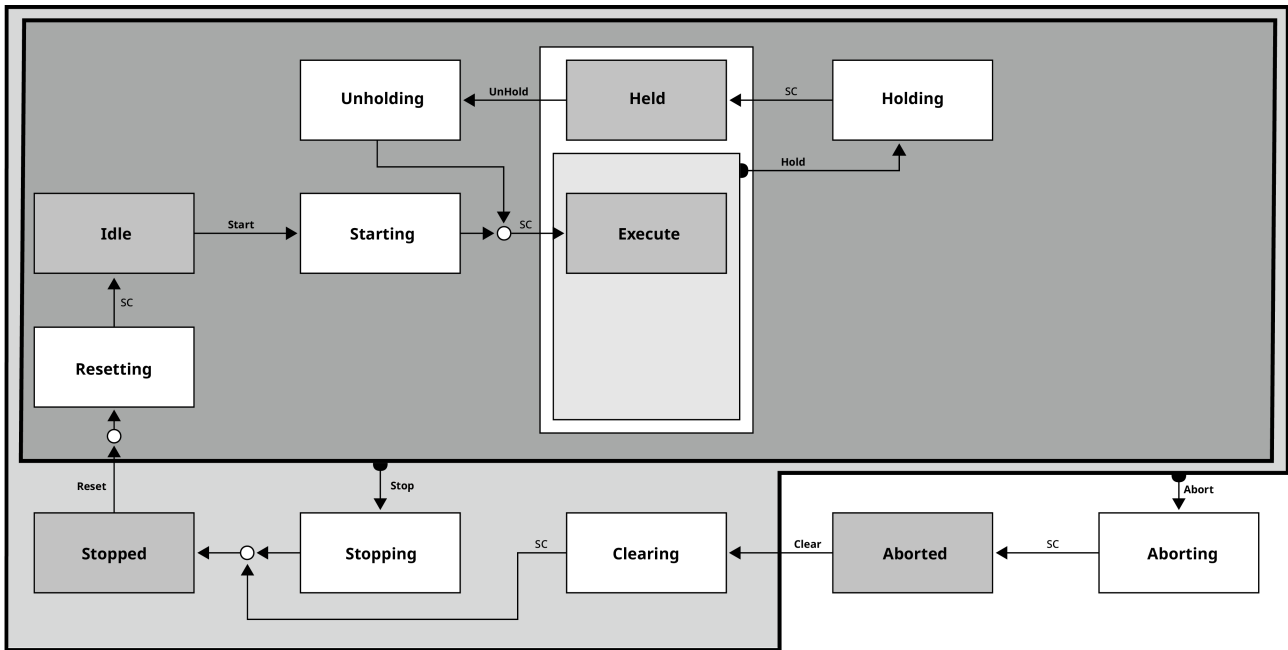


Der PML_StateMachine-Funktionsbaustein hat in der aktualisierten Form eine gemeinsame Schnittstelle zum PackML Machine State Model V3. Es wird erwartet, dass anwendungsspezifische Logik, wie Zustandsübergänge, in externen Funktionsbausteinen programmiert wird und der PML_StateMachine-Funktionsbaustein die zentrale Logik der Zustandsmaschine und die Zustandsdarstellung übernimmt. Aufgrund des aktuell aktiven UnitMode (eMode) stellt sich das Machine State Model unterschiedlich dar. Dazu sind drei Grundmodelle vorkonfiguriert (E_PMLProtectedUnitMode [► 44]).

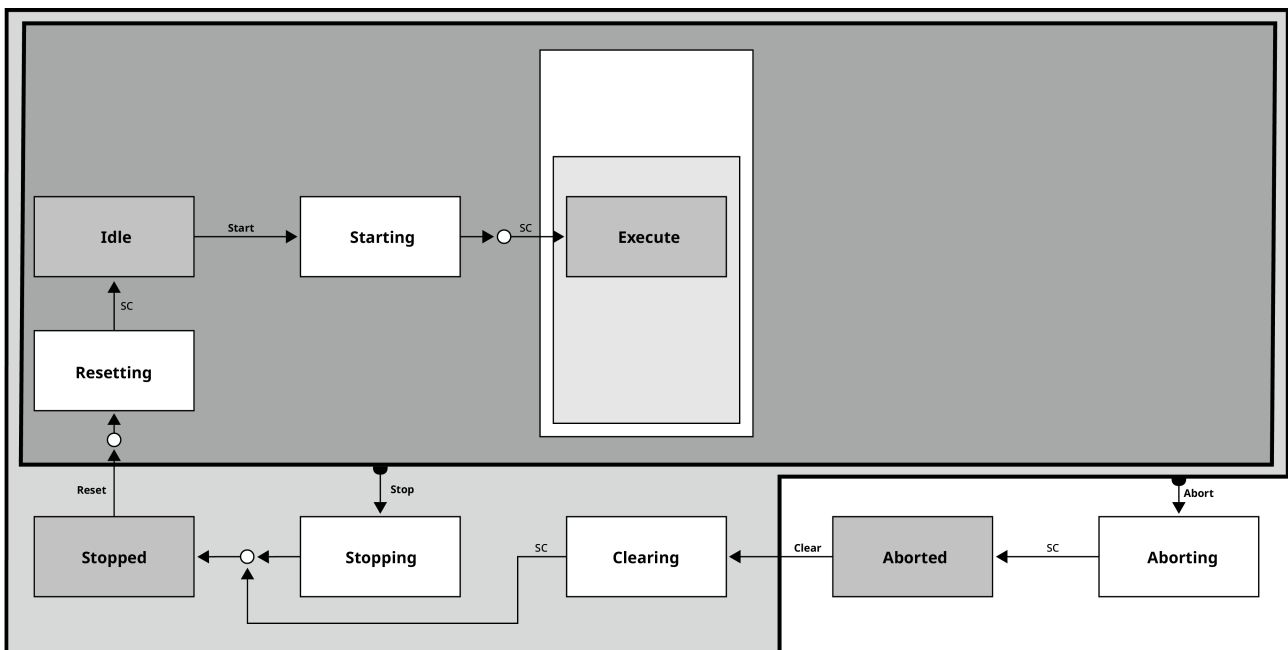
ePMLProtUnitMode_Production



ePMLProtUnitMode_Maintenance



ePMLProtUnitMode_Manual



Weiterhin können weitere anwenderspezifische Modelle in einfacher Weise mit Hilfe des Funktionsbaustein [FB_PMLUnitModeConfig](#) [► 38] erstellt werden und sind so sehr flexibel einsetzbar.

Die Logik für Übergänge, insbesondere zwischen Production, Maintenance und Manual Mode, ist abhängig von der Anwendung. In welchen Zuständen UnitMode-Wechsel für die Grundmodelle zulässig sind, ist in der Beschreibung des Funktionsbaustein [FB_PMLUnitModeManager](#) [► 41] genauer beschrieben.

Eingänge

```

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

Name	Typ	Beschreibung
eMode	DINT	Aktueller PML-UnitMode
eCommand	E_PMLCommand	Enumeration [► 44] mit den verschiedenen PML-Kommandos des Bausteins.
stSubUnitInfoRef	ST_PMLSubUnitInfoRef	Struktur [► 47] zum Verweis auf ein Array der aktuellen PML-States von unterlagerten Maschineneinheiten
stOptions	ST_PMLStateMachineOptions	Momentan nicht verwendet

Ausgänge

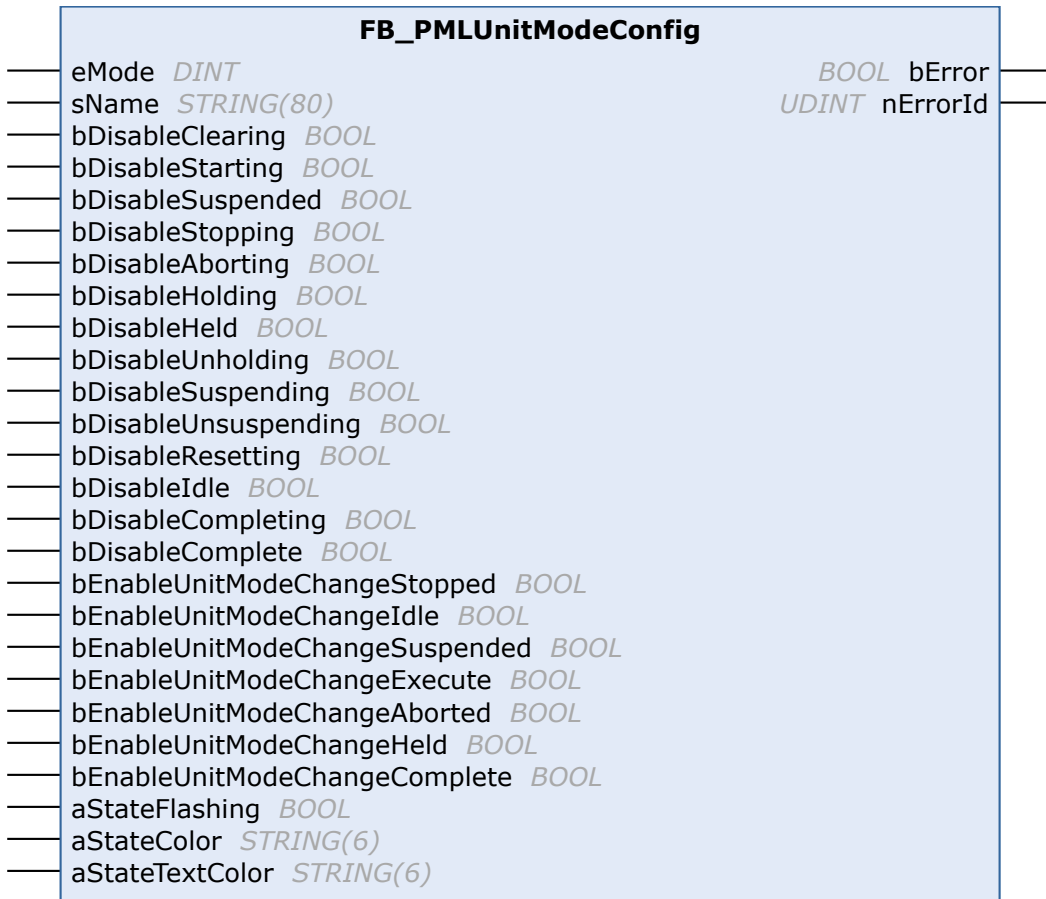
```
VAR_OUTPUT
    eState      : E_PMLState;
    sState      : STRING;
    bError      : BOOL;
    nErrorId    : UDINT;
END_VAR
```

Name	Typ	Beschreibung
eState	E_PMLState	Enumeration [► 46], die den aktuellen PML-Zustand der automatischen Zustandsmaschine liefert.
sState	STRING	Name des aktuellen PML-Zustand der automatischen Zustandsmaschine.
bError	BOOL	Wird TRUE, sobald ein Fehler eintritt.
nErrorId	UDINT	Liefert bei einem gesetzten bError-Ausgang die Fehlernummer.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.3.2 FB_PMLUnitModeConfig



Maschinen haben unter Umständen noch andere Anlagenmodi als „Production“, „Maintenance“ und „Manual“. Dieser Baustein ermöglicht es dem Anwender weitere Modelle (UnitModes) zu konfigurieren.

Dabei können die Nummer des neuen Modells, die vorhandenen Zustände und die Zustände, in denen ein Modell-Wechsel möglich ist, frei definiert werden.

Ab Version 1.0.5.0 der Bibliothek können hier auch das Aktivieren eines Blinkens sowie die State- und Statetext-Farbe hinterlegt werden.

Die Konfiguration ist dann Bestandteil der globalen Struktur ST_PMLUnitModeConfiguration [► 47].

Eingänge

```
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	Typ	Beschreibung
eMode	DINT	Nummer der neuen PML-UnitMode [4..31]
sName	STRING	Name des neuen PML-UnitMode
bDisableClearing	BOOL	Deaktiviert den PMLState „Clearing“.
bDisableStarting	BOOL	Deaktiviert den PMLState „Starting“.
bDisableSuspended	BOOL	Deaktiviert den PMLState „Suspended“. Durch das Deaktivieren des statischen Zustandes werden die PMLState „Suspending“ und „Unsuspending“ ebenfalls deaktiviert.
bDisableStopping	BOOL	Deaktiviert den PMLState „Stopping“.
bDisableAborting	BOOL	Deaktiviert den PMLState „Aborting“.
bDisableHolding	BOOL	Deaktiviert den PMLState „Holding“.
bDisableHeld	BOOL	Deaktiviert den PMLState „Held“. Durch das Deaktivieren des statischen Zustandes werden die PMLState „Holding“ und „Unholding“ ebenfalls deaktiviert.
bDisableUnholding	BOOL	Deaktiviert den PMLState „Unholding“.
bDisableSuspending	BOOL	Deaktiviert den PMLState „Suspending“.
bDisableUnsuspending	BOOL	Deaktiviert den PMLState „Unsuspending“.
bDisableResetting	BOOL	Deaktiviert den PMLState „Resetting“.
bDisableIdle	BOOL	Deaktiviert den PMLState „Idle“. Durch das Deaktivieren des statischen Zustandes wird der PMLState „Resetting“ ebenfalls deaktiviert.
bDisableCompleting	BOOL	Deaktiviert den PMLState „Completing“.
bDisableComplete	BOOL	Deaktiviert den PMLState „Complete“. Durch das Deaktivieren des statischen Zustandes wird der PMLState „Completing“ ebenfalls deaktiviert.
bEnableUnitModeChangeStopped	BOOL	Gibt einen Modus-Wechsel im PMLState „Stopped“ frei.
bEnableUnitModeChangeIdle	BOOL	Gibt einen Modus-Wechsel im PMLState „Idle“ frei.
bEnableUnitModeChangeSuspended	BOOL	Gibt einen Modus-Wechsel im PMLState „Suspended“ frei.
bEnableUnitModeChangeExecute	BOOL	Gibt einen Modus-Wechsel im PMLState „Execute“ frei.
bEnableUnitModeChangeAborted	BOOL	Gibt einen Modus-Wechsel im PMLState „Aborted“ frei.
bEnableUnitModeChangeHeld	BOOL	Gibt einen Modus-Wechsel im PMLState „Held“ frei.
bEnableUnitModeChangeComplete	BOOL	Gibt einen Modus-Wechsel im PMLState „Complete“ frei.
aStateFlashing	ARRAY[0..17] OF BOOL	Definiert das Blinken der States, wenn aktiv.
aStateColor	ARRAY[0..17] OF STRING[6]	Definiert, die Farbe der States, wenn aktiv.

Name	Typ	Beschreibung
aStateTextColor	ARRAY[0..17] OF STRING[6]	Definiert, die Text-Farbe der States, wenn aktiv

Ausgänge

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

Name	Typ	Beschreibung
bError	BOOL	Wird TRUE, sobald ein Fehler eintritt.
nErrorId	UDINT	Liefert bei einem gesetzten bError-Ausgang die Fehlernummer.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

4.3.3 FB_PMLUnitModeManager

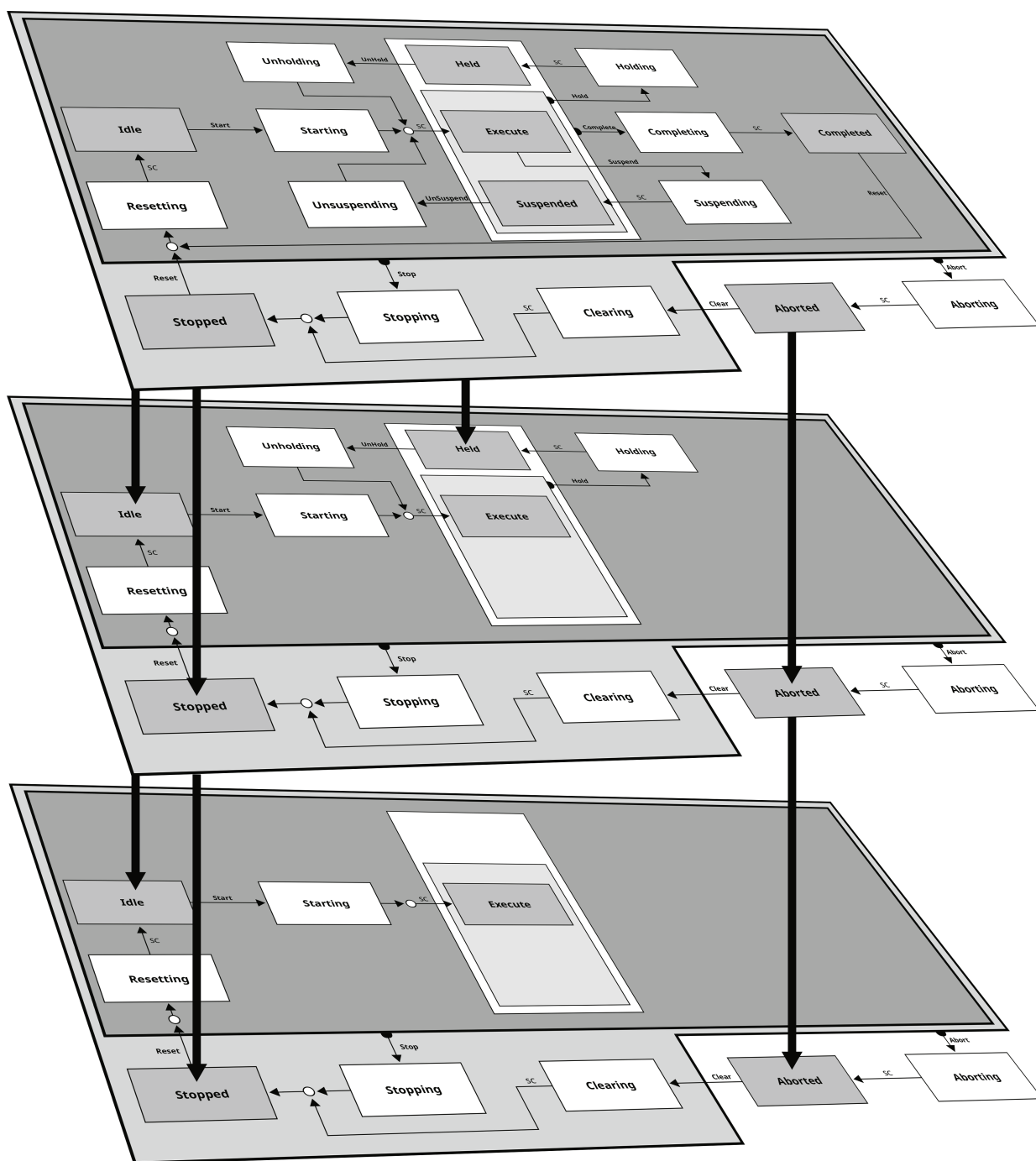


Maschinen haben andere Anlagenmodi als „Production“. Jeder Anlagenmodus ist durch ein eigenes Zustandsmodell definiert. Für Übergänge zwischen den Modi muss ein „Mode Manager“ definiert werden. Der „Mode Manager“ entscheidet, wie und in welchen Zustand eine Maschine Anlagenmodi ändern kann, d.h. eingebaute Sperren verhindern, dass die Maschine in ungeeignete Zustände wechselt. Für die Grundmodi „Production“, „Maintenance“ und „Manual“ sind diese Sperren vordefiniert, wie die Darstellung unten zeigt. Für weitere über den [FB_PMLUnitModeConfig](#) [► 38]-Baustein definierte Modi kann dies individuell festgelegt werden.

WARNUNG

Ordnungsgemäße Modusänderungen einhalten

Die Logik für Übergänge zwischen den Modi ist abhängig von der Anwendung, insbesondere für Übergänge zwischen „Manual“- und „Production“-Modus. Für solche Modusänderungen können darüber hinaus Hardwaresperren oder Sicherheitsausrüstung erforderlich sein. Die Verantwortung für ordnungsgemäße Moduswechsel liegt bei demjenigen, der sie implementiert.



Eingänge

```

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

```

Name	Typ	Beschreibung
bExecute	BOOL	Moduswechsel bei steigender Flanke
eModeCommand	DINT	Angeforderter Modus
eState	E_PMLState	Enumeration [► 46] , die den aktuellen PML-Zustand der automatischen Zustandsmaschine liefert.

 Ausgänge

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

Name	Typ	Beschreibung
eModeStatus	DINT	Aktueller PML-UnitMode
sModeStatus	STRING	Name des aktuellen PML-UnitMode
bDone	BOOL	Wird TRUE, sobald der Modus-Wechsel erfolgreich durchgeführt wurde.
bError	BOOL	Wird TRUE, sobald ein Fehler eintritt.
nErrorID	UDINT	Liefert bei einem gesetzten bError-Ausgang die Fehlernummer.

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5 Datentypen

Datentypen, die in der PackML-Bibliothek Verwendung finden.

5.1 General

Allgemeine Datentypen

5.1.1 ST_PMLAdminTimeOptions

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

Name	Beschreibung
UseExternalTime	Wird dieses Flag TRUE gesetzt, so wird anstelle der Zeitinformation des Systems nun die am Eingang ExternalPackMLTime vorgegebene Zeit verwendet.
ExternalPackMLTime	Extern vorgegebene Zeit

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.2 Packaging machine state and mode

Datentypen des PackML-State und -Mode Handlings

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
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.2.2 E_PMLProtectedUnitMode

```
TYPE E_PMLProtectedUnitMode :
(
    Invalid := 0,
```

```
    Production      := 1,  
    Maintenance    := 2,  
    Manual         := 3  
);  
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab 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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.2.4 ST_PMLStateMachineOptions

```

TYPE ST_PMLStateMachineOptions :
STRUCT
END_STRUCT
END_TYPE

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab 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	Beschreibung
Active	Signalisiert, dass dieser unterlagerte Maschinenteil aktiv ist und den Zustandsvorgaben der Zustandsmaschine folgt.
eState	Enumeration, die den aktuellen Zustand des unterlagerten Maschinenteils wiedergibt.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab 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	Beschreibung
pArray	Adresse eines eindimensionalen Arrays vom Typ <u>ST_PMLSubUnitInfo</u> [► 46]. Jedes Array-Element enthält den Zustand eines unterlagerten Maschinenteils. Beispiel: stSubUnitInfo : ARRAY[1..10] OF <u>ST_PMLSubUnitInfo</u> [► 46]; pArray := ADR(stSubUnitInfo);
ArraySize	Speichergroße des eindimensionalen Arrays, die mit der SIZEOF-Funktion ermittelt werden kann. Beispiel: ArraySize := SIZEOF(stSubUnitInfo);
NoOfSubUnits	Anzahl der relevanten unterlagerten Maschinenteile.

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.2.7 ST_PMLUnitModeConfiguration

Diese Struktur wird für die GlobalVariables *stPMLUnitModeConfiguration* verwendet. Dort werden die Informationen über die zulässigen States und Übergänge die mit dem FB_PMLUnitModeConfig [► 38] konfiguriert wurden hinterlegt.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3 PackTags

Datentypen der PackML PackTags

5.3.1 Alarm

Datentypen für das Alarm-, Warnungs- und Stoppursachen-Handling

5.3.1.1 ST_PMLDateAndTime

Diese Struktur dient zur Speicherung von Datum und Uhrzeit eines Ereignisses oder zum Quittieren eines Ereignisses.

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.1.2 ST_PMLEvent

Sammlung der Tags für die Beschreibung von Alarmereignissen.

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

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.1.3 ST_PMLEventMin

Sammlung der mindestens notwendigen Tags für die Beschreibung von Alarmereignissen.

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT	PC (i386)	Ab Tc3_PackML_V3 1.0.3

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
• 3.1 Build 4024.63		

5.3.2 Allgemein

Allgemeine Datentypen für die PackML PackTags

5.3.2.1 ST_PMLCumulativeTimes

Sammlung von Tags für die Erfassung von Zeiten in den verschiedenen States und UnitModes in der Maschine.

Durch die Verwendung von [FB_PMLAdminTime](#) [► 33] werden diese automatisch gefüllt und können über die Methode `M_ResetCumulativeTime()` des FB zurückgesetzt werden.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.2 ST_PMLEquipment

Sammlung von Tags für die Signalisierung von Problemen im Downstream (Blocked) oder Upstream(Starved) der Maschine.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.3 ST_PMLIngredients

Sammlung von Tags für die Beschreibung der benötigten Rohmaterialien für das Produkt.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.4 ST_PMLModeStateTimes

Sammlung von Tags für die Erfassung von Zeiten in den verschiedenen States und UnitModes in der Maschine.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.5 ST_PMLParameterDint

Sammlung von Tags für die Beschreibung von DINT-Parametern in der Maschine.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.6 ST_PMLParameterLreal

Sammlung von Tags für die Beschreibung von LREAL-Parametern in der Maschine.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.7 ST_PMLParameterReal

Sammlung von Tags für die Beschreibung von REAL-Parametern in der Maschine.

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.8 ST_PMLParameterString

Sammlung von Tags für die Beschreibung von STRING-Parametern in der Maschine.

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.9 ST_PMLProcessVariables

Sammlung von Tags für die Beschreibung von Prozessvariablen in der Maschine.

```
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
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.10 ST_PMLProductData

Sammlung von Tags für die Beschreibung des auf der Maschine gefertigten Produkts.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.11 ST_PMLProductDataMin

Sammlung aller mindestens notwendigen Tags für die Beschreibung des auf der Maschine gefertigten Produkts.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.2.12 ST_PMLRecipe

Sammlung von Tags für die Beschreibung von Rezepten in der Maschine.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.3 ST_PMLa

Sammlung aller Verwaltung-Tags der PackTag-Struktur.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.4 ST_PMLaMin

Sammlung aller mindestens notwendigen Verwaltung-Tags der PackTag-Struktur.

```

TYPE ST_PMLaMin :
STRUCT
    StopReason              : ST_PMLEventMin;

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.5 ST_PMLc

Sammlung aller Befehl-Tags der PackTag-Struktur.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.6 ST_PMLcMin

Sammlung aller mindestens notwendigen Befehl-Tags der PackTag-Struktur.

```

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

    MachSpeed              : REAL;

    CntrlCmd               : DINT;
    CmdChangeRequest       : BOOL;
END_STRUCT
END_TYPE

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.7 ST_PMLs

Sammlung aller Zustand-Tags der PackTag-Struktur.

```

TYPE ST_PMLs :
STRUCT
    UnitModeCurrent        : DINT;
    UnitModeRequested       : DINT;
    UnitModeChangeInProgress : BOOL;

    StateCurrent            : DINT;
    StateRequested          : DINT;
    StateChangeInProgress   : 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;
RecipeChangeInProgress : BOOL;

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

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.8 ST_PMLsMin

Sammlung aller mindestens notwendigen Zustand-Tags der PackTag-Struktur.

```

TYPE ST_PMLsMin :
STRUCT
    UnitModeCurrent      : DINT;

    StateCurrent         : DINT;

    MachSpeed            : REAL;
    CurMachSpeed         : REAL;
END_STRUCT
END_TYPE

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.9 ST_PMLV2022

Sammlung aller Tags der PackTag-Struktur.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

5.3.10 ST_PMLV2022Min

Sammlung aller mindestens notwendigen Tags der PackTag-Struktur.

```

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

```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzuverbindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

6 Globale Parameter

Parameter für den Aufbau der Packaging Maschine Tag Strukturen. Diese können beim Einfügen der Bibliothek für das aktuelle Projekt angepasst werden.

```
VAR_GLOBAL 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
```

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

7 Globale Konstanten / Variablen

Konstanten für den Aufbau der Packaging Maschine Tag-Strukturen. Diese können nicht verändert werden.

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

In dieser Struktur werden die Informationen über die zulässigen States und Übergänge, die mit dem PML_UnitModeConfig [► 38] konfiguriert wurden hinterlegt.

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

Voraussetzungen

Entwicklungsumgebung	Zielformat	Einzubindende SPS-Bibliothek
Ab TwinCAT • 3.1 Build 4024.63	PC (i386)	Ab Tc3_PackML_V3 1.0.3

8 Interfaces

Definition von Interfaces, die für das PackML Handling verwendet werden können.

8.1 I_PMLUnitStateActing

Dieses Interface kann in den Unit-Bausteinen der Applikation implementiert werden und stellt nur die „Acting“-Methoden des Packaging State Models zur Verfügung, die dann je nach Bedarf mit Applikations-Code gefüllt werden können.

Diese Methoden sind:

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

Dieses Interface kann in den Unit-Bausteinen der Applikation implementiert werden und stellt nur die „Waiting“-Methoden des Packaging State Models zur Verfügung, die dann je nach Bedarf mit Applikations-Code gefüllt werden können.

Diese Methoden sind:

M_Aborted
M_Completed
M_Held
M_Idle
M_Stopped
M_Suspended
M_Undefined

9 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Downloadfinder

Unser Downloadfinder beinhaltet alle Dateien, die wir Ihnen zum Herunterladen anbieten. Sie finden dort Applikationsberichte, technische Dokumentationen, technische Zeichnungen, Konfigurationsdateien und vieles mehr.

Die Downloads sind in verschiedenen Formaten erhältlich.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unserer Internetseite: www.beckhoff.com

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline: +49 5246 963-157

E-Mail: support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline: +49 5246 963-460

E-Mail: service@beckhoff.com

Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon: +49 5246 963-0

E-Mail: info@beckhoff.com

Internet: 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.

Mehr Informationen:
www.beckhoff.com/te1000

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

