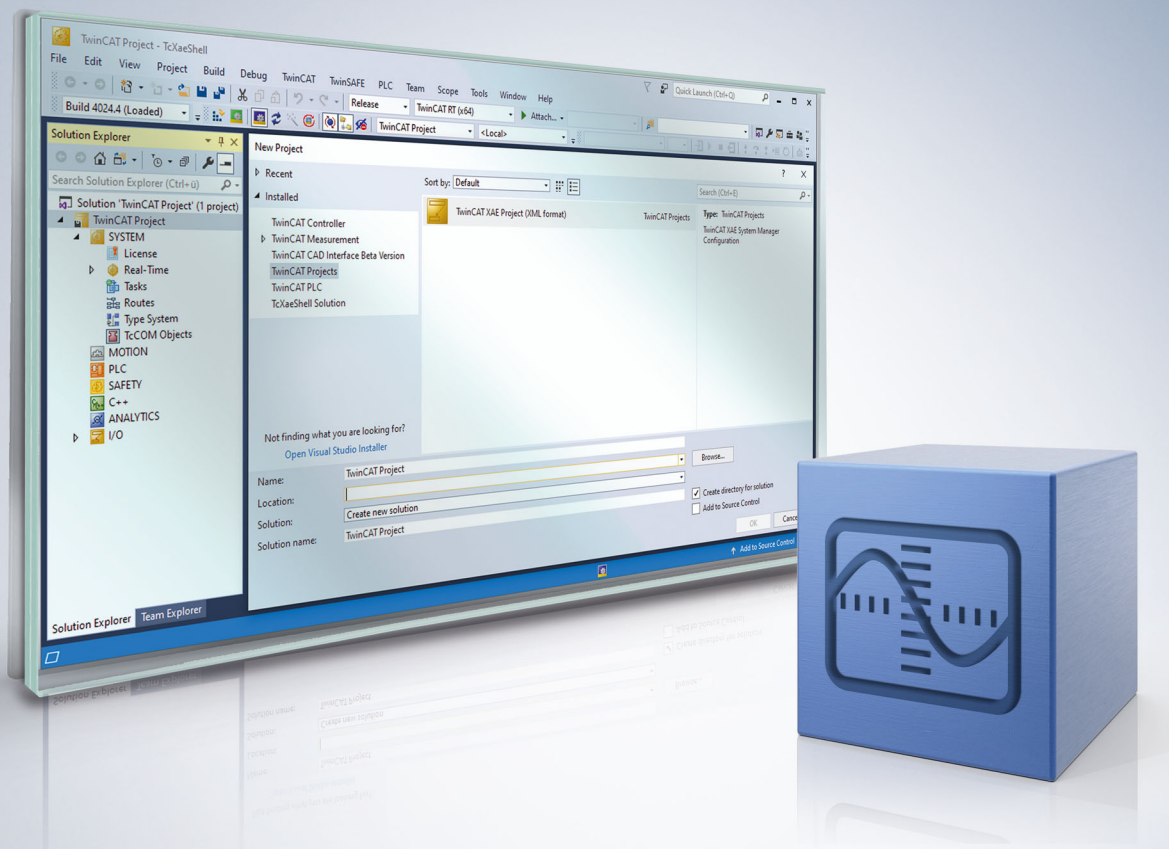
BECKHOFF New Automation Technology

Handbuch | DE

TF3510

TwinCAT 3 | Analytics Library



11.07.2023 | Version: 1.3.0

Inhaltsverzeichnis

1 Vorwort 5
1.1 Hinweise zur Dokumentation 5
1.2 Zu Ihrer Sicherheit 6
1.3 Hinweise zur Informationssicherheit 7
2 Übersicht 8
3 Installation 12
3.1 Systemvoraussetzungen 12
3.2 Installation 12
3.3 Lizenzierung 15
4 Technische Einführung 18
5 PLC API 20
5.1 Funktionsbausteine 20
5.1.1 Algorithmen 20
5.1.2 System 320
5.1.3 IoT 328
5.2 Funktionen 351
5.2.1 F_RawTimespan_TO_Structured 351
5.2.2 F_StructuredTimespan_TO_Raw 351
5.3 Datentypen 352
5.3.1 Allgemeines 352
5.3.2 Config 353
5.3.3 Time 358
6 Beispiele 359
6.1 Kombination von Algorithmen mit lokalen Eingängen 359
7 Anhang 360
7.1 Returncodes 360
7.2 FAQ – Häufig gestellte Fragen und Antworten 362

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, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche 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 weiter entwickelt.
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®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® 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 Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

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

Copyright

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 Übersicht

Die TwinCAT Analytics Library ist eine Sammlung von Basisanalyse-Algorithmen. Funktionen, wie Zeit- und Lebenszyklusanalyse, Peakerkennung, Komparatoren, grundlegende mathematische Operationen, logische Operatoren und viele weitere sind verfügbar. Diese Bibliothek kann als Standard-SPS-Bibliothek für Ihre Maschinenanwendung, aber auch in einer TwinCAT Analytics Runtime (TF3550) für eine zentrale SPS-basierte 24-stündige Analyseanwendung verwendet werden. Im zweiten Fall wird die Bibliothek in einer automatischen SPS-Codegenerierung, die von der TE3500 TwinCAT Analytics Workbench gesteuert wird, eingesetzt.

Komponenten

- Treiber TcAnalyticsKernel.sys
- SPS-Bibliothek Tc3_Analytics.compiled-library

Liste der Algorithmen

Base
FB_ALY_ContinuousPieceCounter_1Ch [▶ 24]
FB_ALY_DownsamplingBuffer_1Ch [▶ 28]
FB_ALY_EdgeCounter_1Ch [▶ 31]
FB_ALY_EdgeCounterOnOff_1Ch [▶ 35]
FB_ALY_EdgeCounterOnOff_2Ch [▶ 39]
FB_ALY_EventTimingAnalysis_1Ch [▶ 43]
FB_ALY_EventTimingAnalysis_2Ch [▶ 47]
FB_ALY_IntervalPieceCounter_1Ch [▶ 56]
FB_ALY_LifecycleAnalysis_1Ch [▶ 64]
FB_ALY_LifetimeAnalysis_1Ch [▶ 68]
FB_ALY_MinMaxAvg_1Ch [▶ 72]
FB_ALY_MinMaxAvgInterval_1Ch [▶ 74]
FB_ALY_MovingAvg_1Ch [▶ 77]
FB_ALY_MovingIntervalCounter_1Ch [▶ 80]
FB_ALY_ProductivityDiagnosis_3Ch [▶ 87]
FB_ALY_ProductivityInterval_1Ch [▶ 91]
FB_ALY_SignalGenerator_1Ch [▶ 95]
FB_ALY_TimeClock_1Ch [▶ 98]
FB_ALY_Timer_1Ch [▶ 100]
FB_ALY_TimingAnalysis_1Ch [▶ 104]
Classification
FB_ALY_BandwidthClassificator_1Ch [▶ 108]
FB_ALY_BandwidthClassificator_3Ch [▶ 111]
FB_ALY_CurveSketcher_1Ch [▶ 113]
FB_ALY_Histogram_1Ch [▶ 116]
FB_ALY_SectionTimer_1Ch [▶ 120]
FB_ALY_StateHistogram_1Ch [▶ 124]
FB_ALY_ThresholdClassificator_1Ch [▶ 128]
FB_ALY_ThresholdStringClassificator_1Ch [▶ 131]
FB_ALY_TimeBasedEnvelope_1Ch [▶ 134]
Clustering
FB_ALY_DenStream [▶ 144]
FB_ALY_SequentialKMeans [▶ 151]
Compare
FB_ALY_Demultiplexer [▶ 158]
FB_ALY_DetectStringChange_1Ch [▶ 161]
FB_ALY_LogicOperationCounter [▶ 175]
FB_ALY_Multiplexer [▶ 180]
FB_ALY_NumericalCompare_1Ch [▶ 183]
FB_ALY_NumericalCompare_2Ch [▶ 187]
FB_ALY_StringCompare_1Ch [▶ 190]
FB_ALY_StringCompare_2Ch [▶ 193]
Math
FB_ALY_Integrator_1Ch [▶ 199]

FB_ALY_MathOperation [▶ 202]
FB_ALY_MathOperation_1Ch [▶ 206]
FB_ALY_RMS_1Ch [▶ 208]
FB_ALY_SlopeAnalysis_1Ch [▶ 212]
Statistics
FB_ALY_CorrelationFunction [▶ 219]
FB_ALY_CorrelationFunctionReference [▶ 228]
FB_ALY_StandardDeviation [▶ 253]
FB_ALY_LinearRegressionFitting [▶ 240]
FB_ALY_LinearRegressionInference [▶ 248]
Training Base
FB_ALY_TimeBasedTeachPath_1Ch [▶ 268]
Wind Turbine
FB_ALY_WtTurbulence_1Ch [▶ 295]
XTS
FB_ALY_XtsAccelerationAnalysis_1Ch [▶ 288]
FB_ALY_XtsDistanceIntegrator_1Ch [▶ 290]
FB_ALY_XtsVelocityAnalysis_1Ch [▶ 293]
XY Path Analysis
FB_ALY_XyGateMonitor_2Ch [▶ 274]
FB_ALY_XyShapeMonitor_Circle_2Ch [▶ 278]
FB_ALY_XyShapeMonitor_Rectangle_2Ch [▶ 281]
FB_ALY_XyShapeMonitor_Triangle_2Ch [▶ 284]

3 Installation

Das Setup der TE3500 Analytics Workbench umfasst die TwinCAT Analytics Library. Die Workbench selbst ist für die Verwendung des Produkts TF3510 nicht erforderlich. Es reicht aus, TE3500 im Demomodus auf dem Engineering-System zu installieren. Wichtig ist die Lizenz für das Zielsystem. Alle notwendigen Treiberkomponenten werden durch die Aktivierung der Konfiguration und den Download des Projekts auf das Zielsystem heruntergeladen.

3.1 Systemvoraussetzungen

Technische Daten	TF3500 TwinCAT 3 Analytics Library
Zielsystem	Windows 10, WinCE, TwinCAT/BSD PC (x86, x64 und ARM)
Min. TwinCAT-Version	3.1.4022.29
Min. TwinCAT Level	TC1200 TC3 | PLC

3.2 Installation

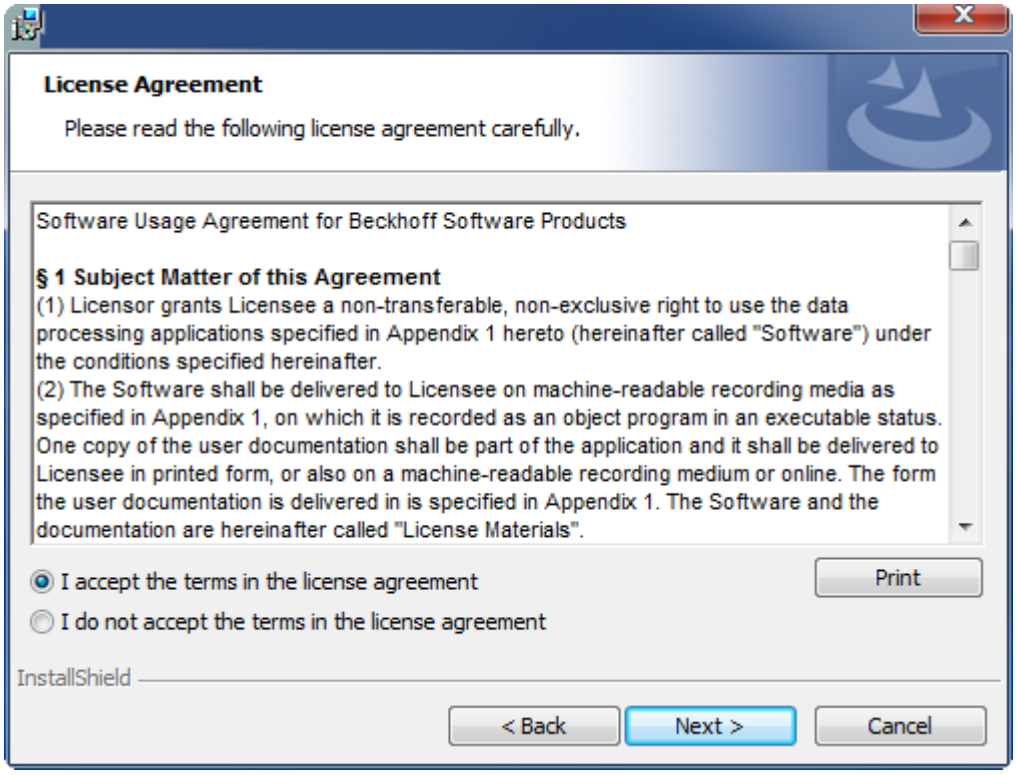
Nachfolgend wird beschrieben, wie die TwinCAT 3 Function für Windows-basierte Betriebssysteme installiert wird.

✓ Die Setup-Datei der TwinCAT 3 Function wurde von der Beckhoff-Homepage heruntergeladen.

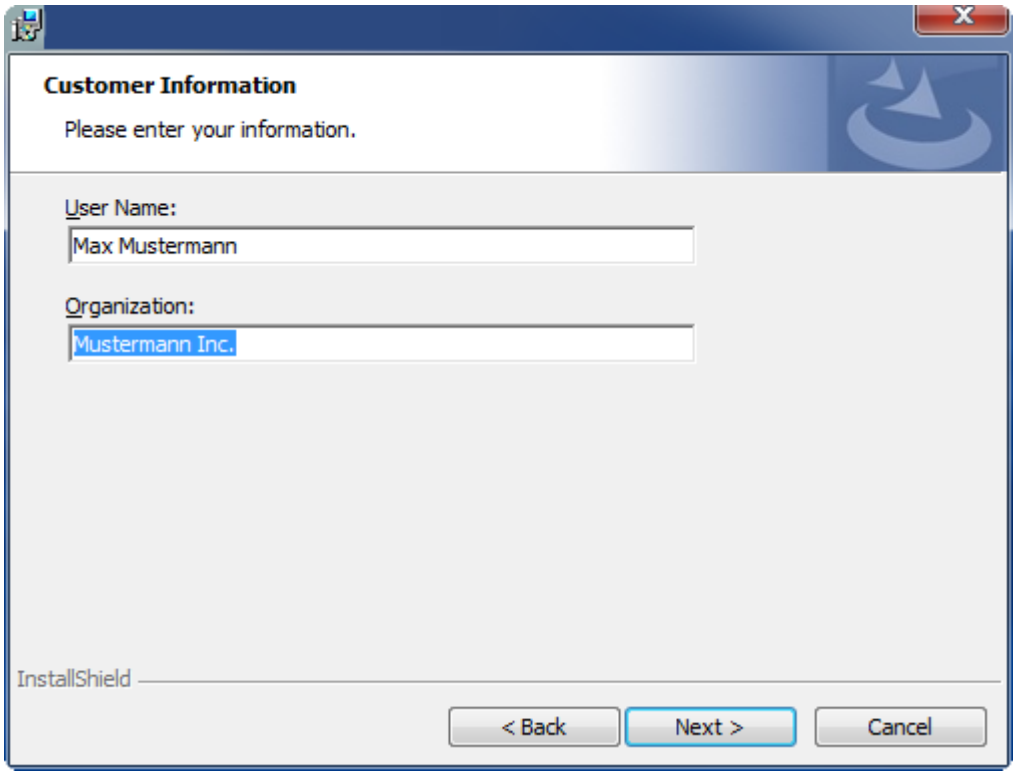
1. Führen Sie die Setup-Datei als Administrator aus. Wählen Sie dazu im Kontextmenü der Datei den Befehl **Als Administrator ausführen**.

   ⇨ Der Installationsdialog öffnet sich.

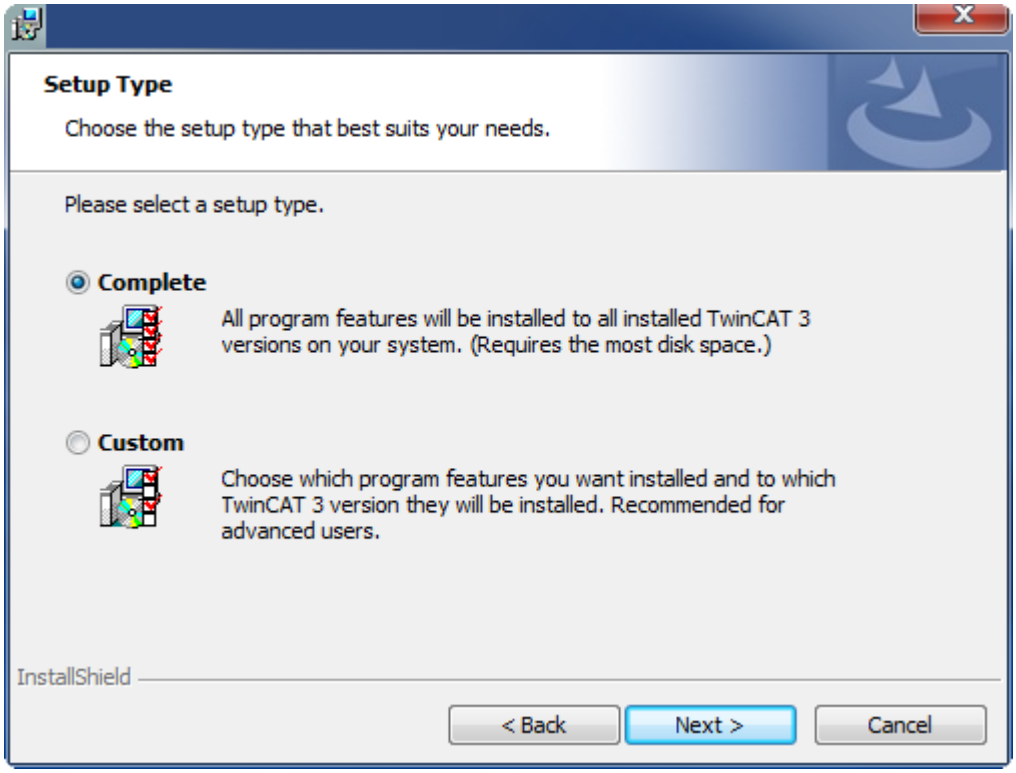
2. Akzeptieren Sie die Endbenutzerbedingungen und klicken Sie auf **Next**.



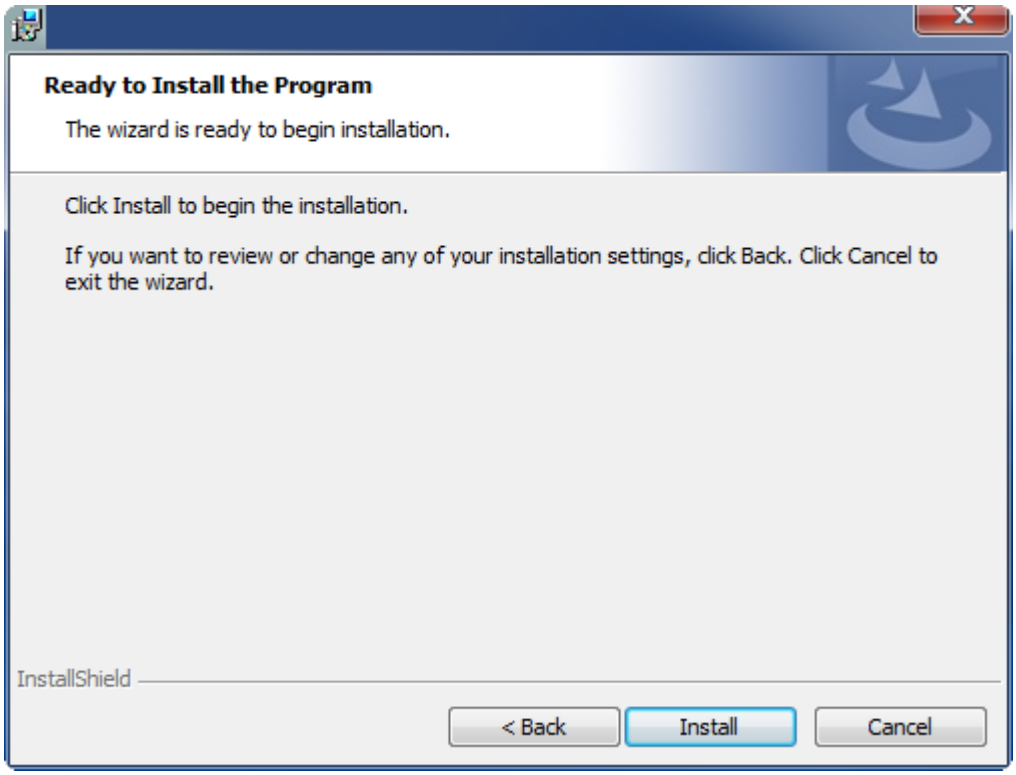
3. Geben Sie Ihre Benutzerdaten ein.



4. Wenn Sie die TwinCAT 3 Function vollständig installieren möchten, wählen Sie **Complete** als Installationstyp. Wenn Sie die Komponenten der TwinCAT 3 Function separat installieren möchten, wählen Sie **Custom**.

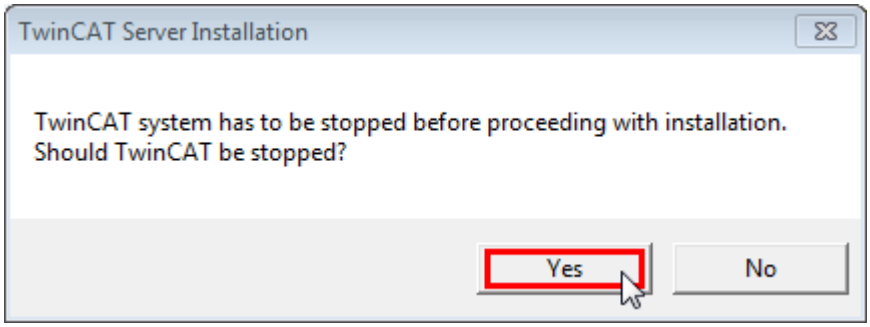


5. Wählen Sie **Next** und anschließend **Install**, um die Installation zu beginnen.

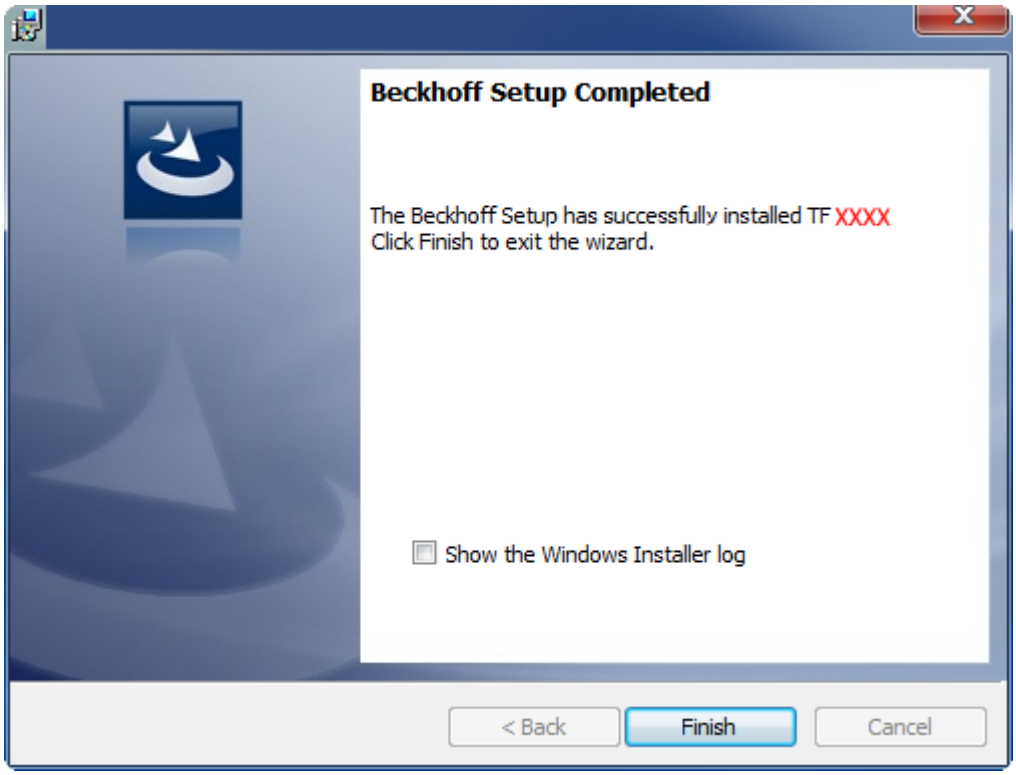


⇨ Ein Dialog weist Sie darauf hin, dass das TwinCAT-System für die weitere Installation gestoppt werden muss.

6. Bestätigen Sie den Dialog mit **Yes.**



7. Wählen Sie **Finish**, um das Setup zu beenden.



⇨ Die TwinCAT 3 Function wurde erfolgreich installiert und kann lizenziert werden (siehe Lizenzierung [▶ 15]).

3.3 Lizenzierung

Die TwinCAT 3 Function ist als Vollversion oder als 7-Tage-Testversion freischaltbar. Beide Lizenztypen sind über die TwinCAT-3-Entwicklungsumgebung (XAE) aktivierbar.

Lizenzierung der Vollversion einer TwinCAT 3 Function

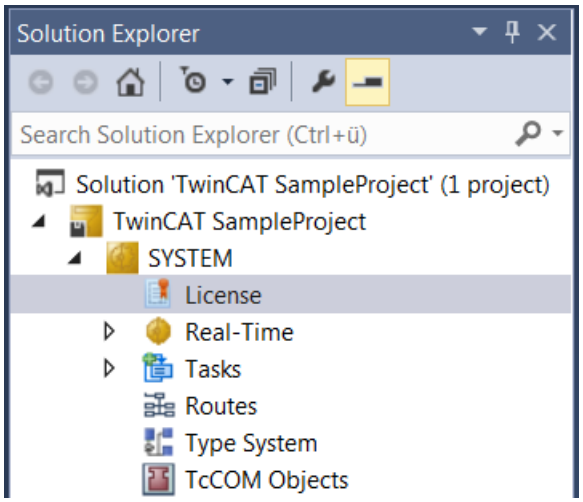
Die Beschreibung der Lizenzierung einer Vollversion finden Sie im Beckhoff Information System in der Dokumentation „TwinCAT 3 Lizenzierung“.

Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function

Eine 7-Tage-Testversion kann nicht für einen TwinCAT-3-Lizenz-Dongle freigeschaltet werden.

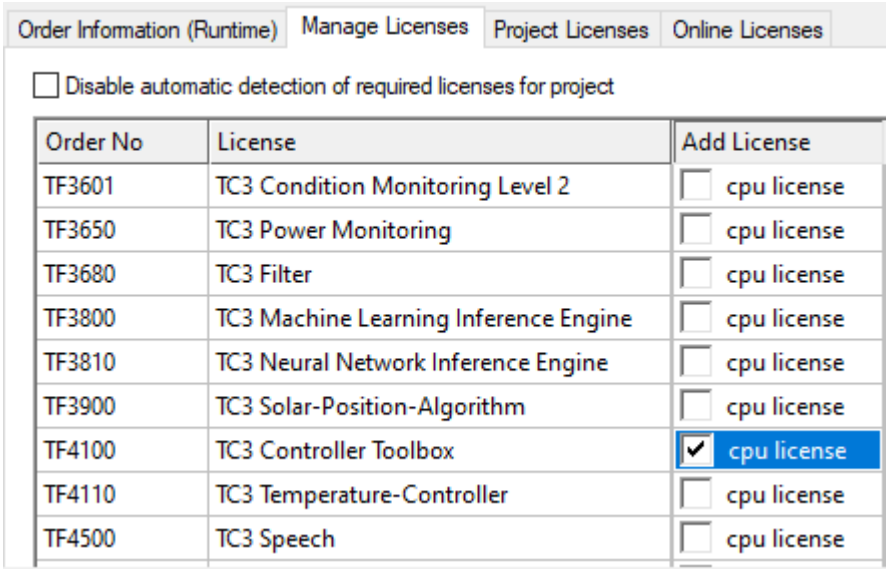
1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).
2. Öffnen Sie ein bestehendes TwinCAT-3-Projekt oder legen Sie ein neues Projekt an.
3. Wenn Sie die Lizenz für ein Remote-Gerät aktivieren wollen, stellen Sie das gewünschte Zielsystem ein. Wählen Sie dazu in der Symbolleiste in der Drop-down-Liste **Choose Target System** das Zielsystem aus.
   ⇨ Die Lizenzierungseinstellungen beziehen sich immer auf das eingestellte Zielsystem. Mit der Aktivierung des Projekts auf dem Zielsystem werden automatisch auch die zugehörigen TwinCAT-3-Lizenzen auf dieses System kopiert.

4. Klicken Sie im **Solution Explorer** im Teilbaum **SYSTEM** doppelt auf **License.**



⇨ Der TwinCAT-3-Lizenzmanager öffnet sich.

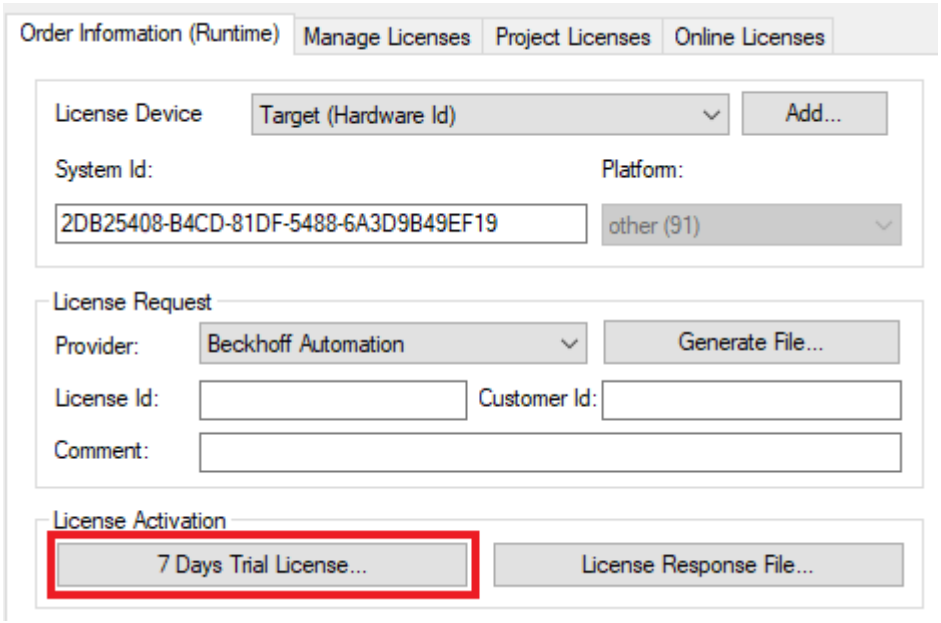
5. Öffnen Sie die Registerkarte **Manage Licenses.** Aktivieren Sie in der Spalte **Add License** das Auswahlkästchen für die Lizenz, die Sie Ihrem Projekt hinzufügen möchten (z. B. „TF4100 TC3 Controller Toolbox“).



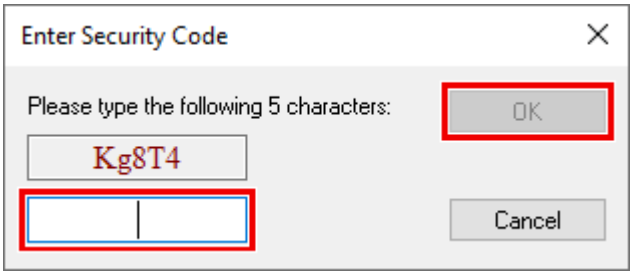
6. Öffnen Sie die Registerkarte **Order Information (Runtime).**

⇨ In der tabellarischen Übersicht der Lizenzen wird die zuvor ausgewählte Lizenz mit dem Status „missing“ angezeigt.

7. Klicken Sie auf **7 Days Trial License...**, um die 7-Tage-Testlizenz zu aktivieren.



⇨ Es öffnet sich ein Dialog, der Sie auffordert, den im Dialog angezeigten Sicherheitscode einzugeben.



8. Geben Sie den Code genauso ein, wie er angezeigt wird, und bestätigen Sie ihn.
9. Bestätigen Sie den nachfolgenden Dialog, der Sie auf die erfolgreiche Aktivierung hinweist.
   ⇨ In der tabellarischen Übersicht der Lizenzen gibt der Lizenzstatus nun das Ablaufdatum der Lizenz an.
10. Starten Sie das TwinCAT-System neu.

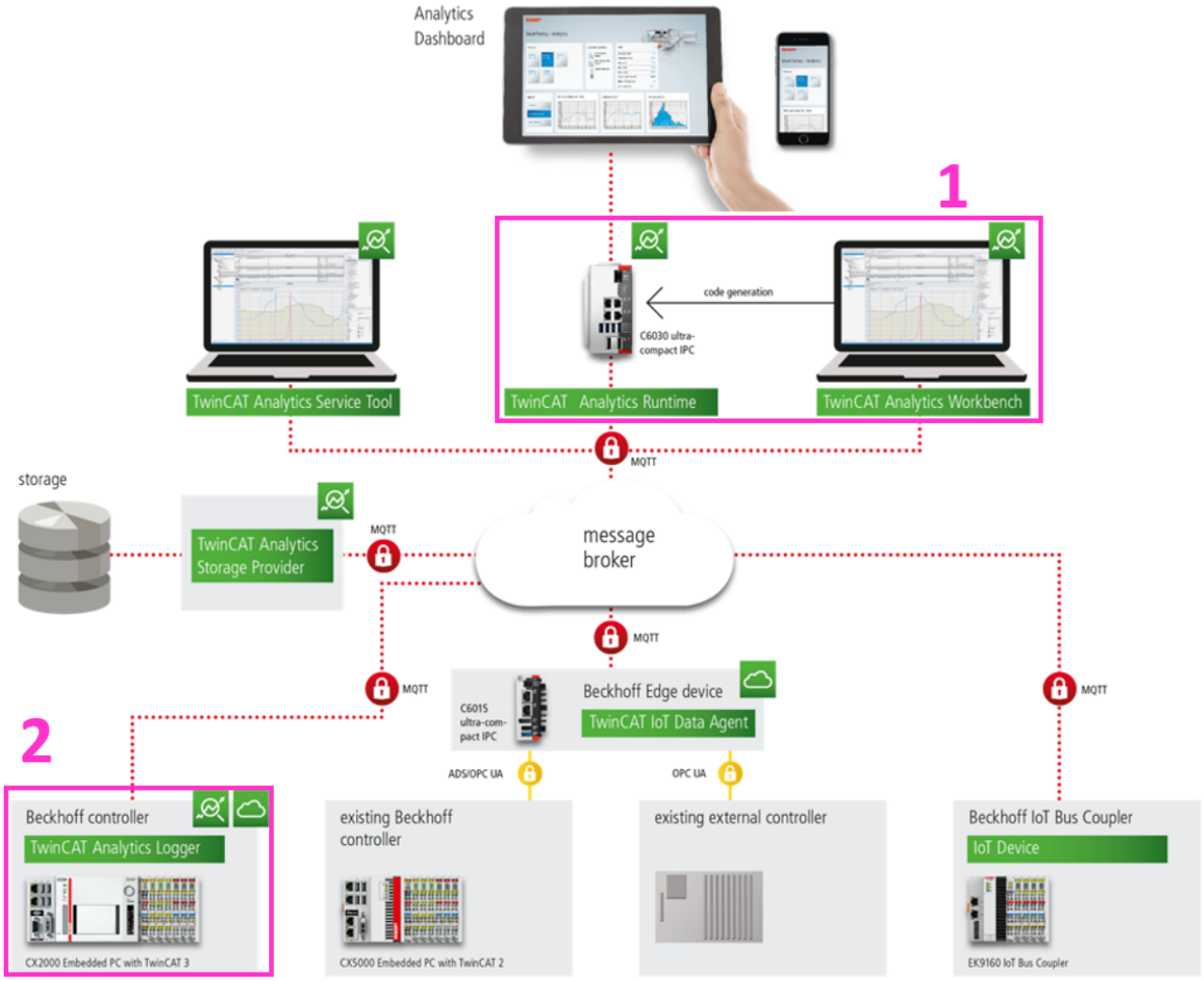
⇨ Die 7-Tage-Testversion ist freigeschaltet.

4 Technische Einführung

Es gibt zwei Hauptanwendungsfälle für die TwinCAT Analytics Library.

1. Die Bibliothek wird bei der automatischen SPS-Codegenerierung der TwinCAT Analytics Workbench verwendet. Alle Basisalgorithmen der Toolbox im Analytics-Konfigurator sind auch in der TwinCAT Analytics Library vorhanden. Der generierte Code kann in die Analytics Runtime heruntergeladen werden, wo er 24/7 parallel zu den Anwendungen im Feld läuft.
2. Die Bibliothek kann auch als Standard-SPS-Bibliothek auf dem Zielgerät im Feld verwendet werden. Tatsächlich ist es nicht notwendig, die Bibliothek immer zusammen mit MQTT-Kommunikation zu verwenden. Sie können auch einfach lokale Variablen nehmen und sie dem Algorithmus der Bibliothek zuweisen. Dies können Sie im Beispielcode [▶ 359] dieses Dokuments sehen.

Wo sie im Workflow von TwinCAT Analytics zum Einsatz kommt, ist im Bild unten dargestellt.



Prinzip und Handhabung

Die Funktionsbausteine des Algorithmus bieten verschiedene Methoden. Mit Ausnahme einiger Algorithmen hat jeder Funktionsbaustein eine Konfigurationsmethode. Diese Methode wird als Erstes aufgerufen, um den Algorithmus z. B. mit Schwellen oder Operatoren usw. zu konfigurieren. Eine weitere „SetChannelValue“-Methode liefert dem Algorithmus die spezifische Eingangsvariable. Der Eingangswert wird erst verwendet, wenn die „Call“-Methode aufgerufen worden ist. Daher hat jeder Funktionsbaustein eine „Call“-Methode, die schließlich die Berechnung startet. Die Ergebnisse sind Ausgänge am Body des Funktionsbausteins.

Zeitstempel

Die Zeitstempel und Zeitspannen in einem TwinCAT Analytics-System basieren auf der in EtherCAT verwendeten DC-Zeit. Zeitstempel werden in Nanosekunden seit dem 01.01.2000 (UTC) ausgedrückt. Sie werden als vorzeichenlose 64-Bit-Integerwerte dargestellt. Zeitspannen werden als vorzeichenbehaftete 64-Bit-Integerwerte dargestellt. Für eine bessere Visualisierung stellen die Algorithmen Funktionsbausteine in ihren Ausgängen zum Speichern der Zeitstempel oder Zeitspannen bereit. Für Zeitstempel wird der Funktionsbaustein FB_ALY_DateTime [▶ 320] verwendet. Zeitspannen werden im Funktionsbaustein FB_ALY_Timespan [▶ 324] gespeichert. Mit Hilfe dieser Funktionsbausteine ist es einfach, mit ihnen Berechnungen anzustellen oder sie als Strings für eine HMI-Anwendung zur Verfügung zu stellen.

5 PLC API

5.1 Funktionsbausteine

5.1.1 Algorithmen

5.1.1.1 Base

5.1.1.1.1 FB_ALY_BatchNShift_1Ch

Der *BatchNShift 1Ch* puffert die Werte des Eingangssignals entsprechend der Puffergröße und des Sample Modus. Die Anzahl der Ausgangskanäle, in denen die gepufferten Eingangswerte gespeichert werden, entspricht der Puffergröße. Mithilfe des Sample Modus kann zwischen zwei verschiedenen Betriebsarten des Algorithmus unterschieden werden. Wird der Sample Modus *Flow* gewählt, so wird ein Ringpuffer beziehungsweise Schieberegister realisiert (Shift). Die Werte werden jeweils nacheinander in den Puffer geschrieben und in jedem Zyklus um eine Stelle des Puffers verschoben. Ist der Puffer voll, fällt der letzte Wert heraus. Im Modus *Wait* wird der Puffer stattdessen immer dann, wenn er vollständig gefüllt ist, komplett geleert und mit neuen Werten befüllt, sodass die Werte in Form von Batches verarbeitet werden (Batch). Zu Beginn einer Analyse wird dabei ebenfalls zunächst gewartet, bis der Puffer vollständig komplett gefüllt ist, erst dann werden die Werte in den Puffer geschrieben. Daher liefert der Baustein erst ab dem Zyklus (*BufferSize + 1)* gültige Werte.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_BatchNShift_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Ausgangskanäle (Größe des internen Puffers).
GetChannelOutputValue()	Local	Methode für das Abholen von einzelnen Ausgangswerten aus dem Ausgangs-Array.
GetOutputArray()	Local	Methode für das Abholen des gesamten Ausgangs-Arrays.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbBatchNShift : FB_ALY_BatchNShift_1Ch(nBufferSize := 100);
    eSampleMode : E_ALY_SampleMode := E_ALY_SampleMode.Flow;
    bConfigure : BOOL := TRUE;
    nInput : INT;
    aBuffer : ARRAY[1..100] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbBatchNShift.Configure(eSampleMode);
END_IF

// Call algorithm
fbBatchNShift.SetChannelValue(nInput);
fbBatchNShift.Call();
fbBatchNShift.GetOutputArray(ADR(aBuffer), SIZEOF(aBuffer));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.1.1 FB_init

Initialisierung der Puffergröße.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nBufferSize: UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nBufferSize	UDINT	Gibt die Größe des Puffers und damit die Anzahl der Werte, die gespeichert werden, an. Die Anzahl der Ausgangskanäle entspricht der Puffergröße.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.1.1.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.1.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
    eSampleMode : E_ALY_SampleMode;
VAR_INPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eSampleMode	E_ALY_SampleMode	Die Werte aus dem Puffer können auf zwei verschiedene Arten an die Ausgangskanäle übergeben werden: *Flow*: Der Puffer wird wie ein Ringpuffer gefüllt. Bei Beginn der Analyse werden alle Ausgangswerte auf null gesetzt. Jede Änderung des Ringpuffers wird sofort an die Ausgangskanäle übertragen. Das Flag New Result wird auf TRUE gesetzt, sobald allen Ausgangskanälen ein Wert zugewiesen wurde, und ist immer TRUE, wenn ein neuer Wert im Puffer gespeichert wird. *Wait*: Bei Beginn der Analyse oder nach einem Reset werden alle Ausgangskanäle auf null gesetzt. Erst, wenn der interne Puffer voll ist, werden diese Werte an die Ausgangskanäle übertragen und das Flag New Result wird auf TRUE gesetzt. Diese Werte bleiben als Ausgangswerte gültig, bis alle Werte im internen Puffer erneuert sind. Erst dann werden sie an die Ausgangskanäle übertragen.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.1.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.1.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.1.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.1.7 GetOutputArray

Abholen des gesamten Ausgangs-Arrays. Die Anzahl an Elementen entspricht der konfigurierten Puffergröße.

Syntax

Definition:

```
METHOD GetOutputArray : BOOL
VAR_INPUT
    pArrayOut : POITNER TO LREAL;
    nArrayOutSize : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pArrayOut	POINTER TO LREAL	Zeiger auf das Ausgangs-Array: Dim: nBufferSize
nArrayOutSize	UDINT	Größe des Ausgangs-Arrays

Rückgabewert

Name	Typ	Beschreibung
GetOutputArray	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.2 FB_ALY_ContinuousPieceCounter_1Ch

Der *Continuous Piece Counter 1Ch* zählt die Anzahl der Teile innerhalb des konfigurierten Intervalls. Der Zähler zählt hoch, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert. Die Berechnung wird neu gestartet, wenn die Zeit des Intervalls abgelaufen ist. Der Algorithmus liefert die Anzahl der Teile, die minimale und maximale Anzahl der Teile sowie die Zeitwerte von Minimum und Maximum.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ContinuousPieceCounter_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nNumIntervals: ULINT;
    nCountLastInterval: ULINT;
    nCountCurrentInterval: ULINT;
    nCountMin: ULINT;
    nCountMax: ULINT;
    fbTimeCountMin: FB_ALY_DateTime;
    fbTimeCountMax: FB_ALY_DateTime;
    fbTimeCurrentInterval: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
nNumIntervals	ULINT	Anzahl der verstrichenen Intervalle.
nCountLastInterval	ULINT	Anzahl der Ereignisse im letzten Intervall.
nCountCurrentInterval	ULINT	Anzahl der Ereignisse im aktuellen Intervall.
nCountMin	ULINT	Minimale Anzahl von Ereignissen im aktuellen Intervall.
nCountMax	ULINT	Maximale Anzahl von Ereignissen im aktuellen Intervall.
fbTimeCountMin	FB_ALY_DateTime	Zeitstempel von nCountMin.
fbTimeCountMax	FB_ALY_DateTime	Zeitstempel von nCountMax.
fbTimeCurrentInterval	FB_ALY_Timespan	Verstrichene Zeit im aktuellen Intervall.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbContinuousPieceCounter : FB_ALY_ContinuousPieceCounter_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
```

```
    stThresholdEdge : ST_ALY_Threshold;
    tInterval : LTIME := LTIME#5M;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbContinuousPieceCounter.ConfigureChannel(stThresholdEdge);
    fbContinuousPieceCounter.Configure(tInterval);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbContinuousPieceCounter.SetChannelValue(nInput);
fbContinuousPieceCounter.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.2.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tInterval : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tInterval	LTIME	Intervallzeit, in der die Ereignisse berücksichtigt werden sollen

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.2.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.2.4 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.2.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.2.6 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.3 FB_ALY_Downsampling_1Ch

Der *Downsampling 1Ch* verarbeitet die Werte des Eingangskanals mit einem konfigurierbaren Downsampling-Faktor. Dadurch wird ein Downsampling erzielt, sodass das Ausgangssignal einer Repräsentation des Eingangssignals mit einer geringeren Abtastrate entspricht. Dies kann beispielsweise sinnvoll sein, um Trends besser zu erkennen oder aber auch um eine nachträgliche Komprimierung hochabgetasteter Signale vorzunehmen, wenn innerhalb der Analyse lediglich geringere Abtastraten benötigt werden. So lässt sich die Performance der Analyse auf einfache Weise steigern.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Downsampling_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fOut: LREAL;
    fbTimeLastSample: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fOut	LREAL	Ausgangssignal mit der um den Downsamplingfaktor geringeren Abtastrate.
fbTimeLastSample	FB_ALY_DateTime	Speichert den Zeitstempel des zuletzt an `fOut` ausgegebenen Datenpunktes.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbDownsampling : FB_ALY_Downsampling_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nDownsamplingFactor : UDINT := 10;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbDownsampling.Configure(nDownsamplingFactor);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbDownsampling.SetChannelValue(nInput);
fbDownsampling.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.3.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
    nDownsamplingFactor : UDINT;
VAR_INPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nDownsamplingFactor	UDINT	Der Faktor, mit dem das Downsampling vorgenommen wird. Ist der Downsampling Factor beispielsweise 100, so wird lediglich jeder 100. Wert gespeichert. Die Abtastzeit (Sample Time) ist damit 100 mal so groß wie die ursprüngliche Zykluszeit, die zwischen zwei abgetasteten Datenpunkten liegt. Wenn der Downsampling-Faktor auf eins gesetzt wird, werden alle Werte gespeichert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.3.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.3.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.4 FB_ALY_EdgeCounter_1Ch

Der *Edge Counter 1Ch* zählt die Anzahl der ausgelösten Ereignisse. Ein Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_EdgeCounter_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bEdge: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bEdge	BOOL	TRUE, wenn eine Flanke erkannt wurde.
nCount	ULINT	Zählt die Anzahl der erkannten Flanken.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt erkannten Flanke.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbEdgeCounter : FB_ALY_EdgeCounter_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_Threshold;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbEdgeCounter.ConfigureChannel(stThresholdEdge);
    fbEdgeCounter.Configure();
END_IF

// Get current system time
fbSystemTime.Call();
```

```
// Call algorithm
fbEdgeCounter.SetChannelValue(nInput);
fbEdgeCounter.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.4.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.4.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.4.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.4.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.4.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.4.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCount : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCount	ULINT	Initialisieren der Anzahl der erkannten Flanken.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.5 FB_ALY_EdgeCounterOnOff_1Ch

Der *Edge Counter On Off 1Ch* zählt die Anzahl der ausgelösten On- und Off-Ereignisse. Ein On-Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten On-Schwelle passiert, und ein Off-Ereignis wird ausgelöst, wenn die Off-Schwelle von demselben Signal passiert wird.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_EdgeCounterOnOff_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    bEdgeOn: BOOL;
    bEdgeOff: BOOL;
    nCountOn: ULINT;
    nCountOff: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	TRUE innerhalb der Zeitspanne zwischen dem On-Ereignis und dem Off-Ergebnis, ansonsten FALSE.
bEdgeOn	BOOL	TRUE, wenn ein On-Ereignis erkannt wurde.
bEdgeOff	BOOL	TRUE, wenn ein Off-Ereignis erkannt wurde.
nCountOn	ULINT	Zählt die Anzahl der ausgelösten On-Ereignisse.
nCountOff	ULINT	Zählt die Anzahl der ausgelösten Off-Ereignisse.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Ereignisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbEdgeCounterOnOff : FB_ALY_EdgeCounterOnOff_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_ThresholdOnOff;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.On.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.On.fThreshold := 1;
    stThresholdEdge.Off.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.Off.fThreshold := 0;
```

```
    fbEdgeCounterOnOff.ConfigureChannel(stThresholdEdge);
    fbEdgeCounterOnOff.Configure();
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbEdgeCounterOnOff.SetChannelValue(nInput);
fbEdgeCounterOnOff.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.5.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.5.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.5.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_ThresholdOnOff;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_ThresholdOnOff [▶ 352]	Kombination von Vergleichsoperator und Schwelle für On-Zustand und Off-Zustand

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.5.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.5.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.5.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCountOn : ULINT;
    nCountOff : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCountOn	ULINT	Initialisieren der Anzahl der erkannten On-Ereignisse.
nCountOff	ULINT	Initialisieren der Anzahl der erkannten Off-Ereignisse.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.6 FB_ALY_EdgeCounterOnOff_2Ch

Der *Edge Counter On Off 2Ch* zählt die Anzahl der ausgelösten On- und Off-Ereignisse. Ein On-Ereignis wird ausgelöst, wenn das Signal des ersten Eingangskanals die konfigurierte Flanke bei einer bestimmten On-Schwelle passiert, und ein Off-Ereignis wird ausgelöst, wenn die Off-Schwelle vom Signal des zweiten Kanals passiert wird.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_EdgeCounterOnOff_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    bEdgeOn: BOOL;
    bEdgeOff: BOOL;
    nCountOn: ULINT;
    nCountOff: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	TRUE innerhalb der Zeitspanne zwischen dem On-Ereignis und dem Off-Ergebnis, ansonsten FALSE.
bEdgeOn	BOOL	TRUE, wenn ein On-Ereignis erkannt wurde.
bEdgeOff	BOOL	TRUE, wenn ein Off-Ereignis erkannt wurde.
nCountOn	ULINT	Zählt die Anzahl der ausgelösten On-Ereignisse.
nCountOff	ULINT	Zählt die Anzahl der ausgelösten Off-Ereignisse.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Ereignisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbEdgeCounterOnOff : FB_ALY_EdgeCounterOnOff_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_ThresholdOnOff;
    bResetOnMultipleOn : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInput : INT;
    fInput : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.On.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.On.fThreshold := 1;
    stThresholdEdge.Off.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
```

```
    stThresholdEdge.Off.fThreshold := 0;

    fbEdgeCounterOnOff.ConfigureChannel(1, stThresholdEdge.On);
    fbEdgeCounterOnOff.ConfigureChannel(2, stThresholdEdge.Off);
    fbEdgeCounterOnOff.Configure(bResetOnMultipleOn);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbEdgeCounterOnOff.SetChannelValue(1, nInput);
fbEdgeCounterOnOff.SetChannelValue(2, fInput);

fbEdgeCounterOnOff.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.6.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.6.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bResetOnMultipleOn : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetOnMultipleOn	BOOL	Zurücksetzen bei mehrfachem On-Ereignis

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.6.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Ein* 2: *Aus*
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.6.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.6.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Ein* 2: *Aus*
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.6.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCountOn : ULINT;
    nCountOff : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCountOn	ULINT	Initialisieren der Anzahl der erkannten On-Ereignisse.
nCountOff	ULINT	Initialisieren der Anzahl der erkannten Off-Ereignisse.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.7 FB_ALY_EventTimingAnalysis_1Ch

Der *Event Timing Analysis 1Ch* misst Zeitdifferenzen zwischen On- und Off-Ereignissen und zählt die Anzahl der ausgelösten Ereignisse. Ein On-Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten On-Schwelle passiert, und ein Off-Ereignis wird ausgelöst, wenn die Off-Schwelle von demselben Signal passiert wird.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_EventTimingAnalysis_1Ch
VAR_INPUT
    bPersistent: BOOL;
    bInitWithThresholdLevel: BOOL
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    nSwitchedOn: ULINT;
```

```
    fbTimeCurrentInterval: FB_ALY_Timespan;
    fbTimeOnMin: FB_ALY_Timespan;
    fbTimeOnMax: FB_ALY_Timespan;
    fbTimeOnAvg: FB_ALY_Timespan;
    fbTimeOnTotal: FB_ALY_Timespan;
    fbTimeOffMin: FB_ALY_Timespan;
    fbTimeOffMax: FB_ALY_Timespan;
    fbTimeOffAvg: FB_ALY_Timespan;
    fbTimeOffTotal: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.
bInitWithThresholdLevel	BOOL	Ist der Wert TRUE, verwendet der Algorithmus zur Initialisierung des internen Zustandes einen Schwellwert, anstatt auf eine Flanke zu warten.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	TRUE innerhalb der Zeitspanne zwischen dem On-Ereignis und dem Off-Ergebnis, ansonsten FALSE.
nSwitchedOn	ULINT	Zählt die Anzahl der ausgelösten On-Ereignisse.
fbTimeCurrentInterval	FB_ALY_Timespan	Zeitspanne des aktuellen Intervalls.
fbTimeOnMin	FB_ALY_Timespan	Minimale Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnMax	FB_ALY_Timespan	Maximale Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnAvg	FB_ALY_Timespan	Durchschnittliche Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnTotal	FB_ALY_Timespan	Gesamtzeit zwischen On-Ereignissen und Off-Ereignissen.
fbTimeOffMin	FB_ALY_Timespan	Minimale Zeit zwischen Off-Ereignis und On-Ereignis
fbTimeOffMax	FB_ALY_Timespan	Maximale Zeit zwischen Off-Ereignis und On-Ereignis
fbTimeOffAvg	FB_ALY_Timespan	Durchschnittliche Zeit zwischen Off-Ereignis und On-Ereignis
fbTimeOffTotal	FB_ALY_Timespan	Gesamtzeit zwischen Off-Ereignissen und On-Ereignissen.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbEventTimingAnalysis : FB_ALY_EventTimingAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_ThresholdOnOff;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.On.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.On.fThreshold := 1;
    stThresholdEdge.Off.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.Off.fThreshold := 0;

    fbEventTimingAnalysis.ConfigureChannel(stThresholdEdge);
    fbEventTimingAnalysis.Configure();
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbEventTimingAnalysis.SetChannelValue(nInput);
fbEventTimingAnalysis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.7.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.7.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.7.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_ThresholdOnOff;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_ThresholdOnOff [▶ 352]	Kombination von Vergleichsoperator und Schwelle für On-Zustand und Off-Zustand. Es besteht die Möglichkeit, nur die Bedingung für das On-Ereignis zu konfigurieren. In diesem Fall wird nur dieses Ereignis bei der Ausführung des Algorithmus berücksichtigt.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.7.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.7.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.8 FB_ALY_EventTimingAnalysis_2Ch

Der *Event Timing Analysis 2Ch* misst Zeitdifferenzen zwischen On- und Off-Ereignissen und zählt die Anzahl der ausgelösten Ereignisse. Ein On-Ereignis wird ausgelöst, wenn das Signal des ersten Eingangskanals die konfigurierte Flanke bei einer bestimmten On-Schwelle passiert, und ein Off-Ereignis wird ausgelöst, wenn die Off-Schwelle vom Signal des zweiten Kanals passiert wird.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_EventTimingAnalysis_2Ch
VAR_INPUT
    bPersistent: BOOL;
    bInitWithThresholdLevel: BOOL
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    nSwitchedOn: ULINT;
```

```
    fbTimeCurrentInterval: FB_ALY_Timespan;
    fbTimeOnMin: FB_ALY_Timespan;
    fbTimeOnMax: FB_ALY_Timespan;
    fbTimeOnAvg: FB_ALY_Timespan;
    fbTimeOnTotal: FB_ALY_Timespan;
    fbTimeOffMin: FB_ALY_Timespan;
    fbTimeOffMax: FB_ALY_Timespan;
    fbTimeOffAvg: FB_ALY_Timespan;
    fbTimeOffTotal: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.
bInitWithThresholdLevel	BOOL	Ist der Wert TRUE, verwendet der Algorithmus zur Initialisierung des internen Zustandes einen Schwellwert, anstatt auf eine Flanke zu warten.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	TRUE innerhalb der Zeitspanne zwischen dem On-Ereignis und dem Off-Ergebnis, ansonsten FALSE.
nSwitchedOn	ULINT	Zählt die Anzahl der ausgelösten On-Ereignisse.
fbTimeCurrentInterval	FB_ALY_Timespan	Zeitspanne des aktuellen Intervalls.
fbTimeOnMin	FB_ALY_Timespan	Minimale Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnMax	FB_ALY_Timespan	Maximale Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnAvg	FB_ALY_Timespan	Durchschnittliche Zeit zwischen On-Ereignis und Off-Ereignis.
fbTimeOnTotal	FB_ALY_Timespan	Gesamtzeit zwischen On-Ereignissen und Off-Ereignissen.
fbTimeOffMin	FB_ALY_Timespan	Minimale Zeit zwischen Off-Ereignis und On-Ereignis.
fbTimeOffMax	FB_ALY_Timespan	Maximale Zeit zwischen Off-Ereignis und On-Ereignis.
fbTimeOffAvg	FB_ALY_Timespan	Durchschnittliche Zeit zwischen Off-Ereignis und On-Ereignis.
fbTimeOffTotal	FB_ALY_Timespan	Gesamtzeit zwischen Off-Ereignissen und On-Ereignissen.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbEventTimingAnalysis : FB_ALY_EventTimingAnalysis_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_ThresholdOnOff;
    bResetOnMultipleOn : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInput : INT;
    fInput : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.On.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.On.fThreshold := 1;
    stThresholdEdge.Off.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.Off.fThreshold := 0;

    fbEventTimingAnalysis.ConfigureChannel(1, stThresholdEdge.On);
    fbEventTimingAnalysis.ConfigureChannel(2, stThresholdEdge.Off);
    fbEventTimingAnalysis.Configure(bResetOnMultipleOn);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbEventTimingAnalysis.SetChannelValue(1, nInput);
fbEventTimingAnalysis.SetChannelValue(2, fInput);
fbEventTimingAnalysis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.8.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.8.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
    bResetOnMultipleOn : BOOL;
VAR_INPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetOnMultipleOn	BOOL	Zurücksetzen bei mehrfachem On-Ereignis

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.8.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Ein* 2: *Aus*
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.8.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.8.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Ein* 2: *Aus*
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.9 FB_ALY_FlipFlop_2Ch

Der *Flip Flop 2Ch* realisiert eine bistabile Kippstufe. Die Dominanz für das Setzen (RS) oder das Rücksetzens (SR) des Ausgangswertes kann konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_FlipFlop_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOut: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bOut	BOOL	Ergebnis der bistabilen Kippstufe.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert TRUE ist. Das Verhalten ist abhängig von dem Konfigurationsparameter eCountMode.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Ereignisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbFlipFlop : FB_ALY_FlipFlop_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;

    stConfigS : ST_ALY_Threshold :=(eComparisonOperator:=E_ALY_ComparisonOperator.GreaterThan, fThre
shold:= 0.5);
    stConfigR : ST_ALY_Threshold :=(eComparisonOperator:=E_ALY_ComparisonOperator.GreaterThan, fThre
shold:= 0.5);
    bSetIsDominant : BOOL := TRUE;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.Cyclic;
    bConfigure : BOOL := TRUE;

    nInS : INT;
    bInR : BOOL;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbFlipFlop.ConfigureChannel(1, stConfigS);
    fbFlipFlop.ConfigureChannel(2, stConfigR);

    fbFlipFlop.Configure(bSetIsDominant, eCountMode);
END_IF

// Call algorithm
fbFlipFlop.SetChannelValue(1, nInS);
fbFlipFlop.SetChannelValue(2, bInR);
fbFlipFlop.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.9.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.9.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bSetIsDominant : BOOL;
    eCountMode : E_ALY_CountMode;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bSetIsDominant	BOOL	Die Domianz des Setzens (RS) oder des Rücksetzens (SR) kann konfiguriert werden.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf TRUE ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung TRUE ist.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.9.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    stThresholdLevel : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Set (S) 2: Reset (R)
stThresholdLevel	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Schwellwertbetrachtung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.9.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.9.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Set (S) 2: Reset (R)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.9.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCount : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCount	ULINT	Initialisieren des Zählwertes.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.10 FB_ALY_IntervalPieceCounter_1Ch

Der *Interval Piece Counter 1Ch* zählt die Anzahl der ausgelösten Ereignisse innerhalb eines konfigurierten Intervalls, das beginnt, wenn der Wert des Flags Start Intervall *TRUE ist.* Ein Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert. Die Berechnung wird neu gestartet, wenn die Zeit des Intervalls abgelaufen ist und der Wert des Flags Start Intervall wieder *True* ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IntervalPieceCounter_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bExecutingInterval: BOOL;
    nNumIntervals: ULINT;
    nCountLastInterval: ULINT;
    nCountCurrentInterval: ULINT;
    nCountMin: ULINT;
    nCountMax: ULINT;
    fbTimeCountMin: FB_ALY_DateTime;
    fbTimeCountMax: FB_ALY_DateTime;
    fbTimeCurrentInterval: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bExecutingInterval	BOOL	`TRUE`, wenn das Intervall ausgeführt wird.
nNumIntervals	ULINT	Anzahl der verstrichenen Intervalle.
nCountLastInterval	ULINT	Anzahl der Ereignisse im letzten Intervall.
nCountCurrentInterval	ULINT	Anzahl der Ereignisse im aktuellen Intervall.
nCountMin	ULINT	Minimale Anzahl von Ereignissen im aktuellen Intervall.
nCountMax	ULINT	Maximale Anzahl von Ereignissen im aktuellen Intervall.
fbTimeCountMin	FB_ALY_DateTime	Zeitstempel von `nCountMin`.
fbTimeCountMax	FB_ALY_DateTime	Zeitstempel von `nCountMax`.
fbTimeCurrentInterval	FB_ALY_Timespan	Verstrichene Zeit im aktuellen Intervall.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Beispiel

```
VAR
    fbIntervalPieceCounter : FB_ALY_IntervalPieceCounter_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_Threshold;
    bResetOnMultipleStart : BOOL := FALSE;
    tInterval : LTIME := LTIME#5M;
    bConfigure : BOOL := TRUE;
```

```
    bStartInterval : BOOL := FALSE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbIntervalPieceCounter.ConfigureChannel(stThresholdEdge);
    fbIntervalPieceCounter.Configure(bResetOnMultipleStart, tInterval);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbIntervalPieceCounter.SetChannelValue(nInput);
fbIntervalPieceCounter.Call(fbSystemTime.tSystemTime, bStartInterval)
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.10.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    bStartInterval : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel.
bStartInterval	BOOL	Start der Verarbeitung des konfigurierten Intervalls.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.10.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bResetOnMultipleStart : BOOL;
    tInterval : LTIME
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetOnMultipleStart	BOOL	Zurücksetzen bei mehrfachem „Start“-Ereignis
tInterval	LTIME	Intervallzeit, in der die Ereignisse berücksichtigt werden sollen

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.10.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.10.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.10.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.10.6 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.11 FB_ALY_LatchingSwitch_1Ch

Der *Latching Switch 1Ch* realisiert einen virtuellen Stromstoßschalter. Der Ausgang wechselt zwischen TRUE und FALSE bei jeder erkannten Flanke am Eingang.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_FlipFlop_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOut: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bOut	BOOL	Ergebnis des virtuellen Stromstoßschalters.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert wechselt.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Ereignisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbLatchingSwitch : FB_ALY_LatchingSwitch_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;

    stConfig: ST_ALY_Threshold :=(eComparisonOperator:=E_ALY_ComparisonOperator.GreaterThan, fThresh
old:= 0.5);
    bConfigure : BOOL := TRUE;

    nInput : INT;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbLatchingSwitch.ConfigureChannel(stConfig);
    fbLatchingSwitch.Configure();
END_IF

// Call algorithm
fbLatchingSwitch.SetChannelValue(nInput);
fbLatchingSwitch.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.11.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.11.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.11.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.11.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.11.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.11.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCount : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCount	ULINT	Initialisieren des Zählwertes.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.12 FB_ALY_LifecycleAnalysis_1Ch

Der *Lifecycle Analysis 1Ch* berechnet die abgelaufenen und die geschätzten verbleibenden Zyklen eines Geräts. Wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert, werden die abgelaufenen Zyklen erhöht und die verbleibenden Zyklen verringert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_LifecycleAnalysis_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nCyclesElapsed: ULINT;
    nCyclesRemaining: LINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
nCyclesElapsed	ULINT	Anzahl der abgelaufenen Zyklen.
nCyclesRemaining	LINT	Verbleibende Zyklen.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbLifecycleAnalysis : FB_ALY_LifecycleAnalysis_1Ch;
    stThresholdEdge : ST_ALY_Threshold;
    nCyclesEstimated : ULINT := 1_000_000;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbLifecycleAnalysis.ConfigureChannel(stThresholdEdge);
    fbLifecycleAnalysis.Configure(nCyclesEstimated);
END_IF

// Call algorithm
fbLifecycleAnalysis.SetChannelValue(nInput);
fbLifecycleAnalysis.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.12.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.12.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nCyclesEstimated : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCyclesEstimated	ULINT	Geschätzte Komponentenzyklen

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.12.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.12.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.12.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.12.6 SetInitial

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCyclesElapsed : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCyclesElapsed	ULINT	Initialisieren der Anzahl abgelaufener Zyklen.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.13 FB_ALY_LifetimeAnalysis_1Ch

Der *Lifetime Analysis 1Ch* berechnet die abgelaufene und die geschätzte verbleibende Lebensdauer eines Geräts. Wenn der Eingangswert die konfigurierte Bedingung erfüllt, wird die Lebensdauer verringert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_LifetimeAnalysis_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbLifetimeElapsed: FB_ALY_Timespan;
    fbLifetimeRemaining: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbLifetimeElapsed	FB_ALY_Timespan	Zeit, die von der Lebensdauer abgelaufen ist.
fbLifetimeRemaining	FB_ALY_Timespan	Erwartete verbleibende Lebensdauer.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Ausgangswerten für die Algorithmen, z. B. bereits abgelaufene Lebensdauer.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbLifetimeAnalysis : FB_ALY_LifetimeAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdLevel : ST_ALY_Threshold;
    tLifetimeEstimated : LTIME := LTIME#10000H;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdLevel.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdLevel.fThreshold := 1;

    fbLifetimeAnalysis.ConfigureChannel(stThresholdLevel);
    fbLifetimeAnalysis.Configure(tLifetimeEstimated);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbLifetimeAnalysis.SetChannelValue(nInput);
fbLifetimeAnalysis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.13.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.13.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tLifetimeEstimated : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tLifetimeEstimated	LTIME	Geschätzte Komponentenlebensdauer

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.13.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdLevel : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdLevel	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und verwendete Schwelle für On-Zustand.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.13.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.13.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.13.6 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    tLifetimeElapsed : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCount	ULINT	Initialisieren der bereits abgelaufenen Lebensdauer.

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.14 FB_ALY_MinMaxAvg_1Ch

Der *Min Max Avg 1Ch* berechnet das Minimum, Maximum und den Durchschnitt der Eingangswerte vom Beginn der Analyse bis zum gegenwärtigen Zeitpunkt. Des Weiteren werden die Zeitwerte des Minimums und Maximums angezeigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MinMaxAvg_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fMin: LREAL;
    fMax: LREAL;
    fAvg: LREAL;
    fbTimeMin: FB_ALY_DateTime;
    fbTimeMax: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fMin	LREAL	Minimum der Eingangswerte.
fMax	LREAL	Maximum der Eingangswerte.
fAvg	LREAL	Durchschnitt der Eingangswerte.
fbTimeMin	FB_ALY_DateTime	Zeitstempel von fMin.
fbTimeMax	FB_ALY_DateTime	Zeitstempel von fMax.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbMinMaxAvg : FB_ALY_MinMaxAvg_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nInput : INT;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbMinMaxAvg.SetChannelValue(nInput);
fbMinMaxAvg.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.14.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.14.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.14.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.15 FB_ALY_MinMaxAvgInterval_1Ch

Der *Min Max Avg Interval 1Ch* berechnet das Minimum, Maximum und den Durchschnitt der Eingangswerte für den Zeitraum des konfigurierten Intervalls. Des Weiteren werden die Zeitwerte des Minimums und Maximums angezeigt. Zu beachten ist, dass alle Werte aus dem relativen letzten Intervall stammen und dass sie erst aktualisiert werden, wenn das Intervall vorüber ist. Die Berechnung wird neu gestartet, wenn die Zeit des Intervalls abgelaufen ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MinMaxAvgInterval_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fMin: LREAL;
    fMax: LREAL;
    fAvg: LREAL;
    fbTimeMin: FB_ALY_DateTime;
    fbTimeMax: FB_ALY_DateTime;
    fbTimeCurrentInterval: FB_ALY_Timespan;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fMin	LREAL	Minimum der Eingangswerte im aktuellen Zeitintervall.
fMax	LREAL	Maximum der Eingangswerte im aktuellen Zeitintervall.
fAvg	LREAL	Durchschnitt der Eingangswerte im aktuellen Zeitintervall.
fbTimeMin	FB_ALY_DateTime	Zeitstempel von `fMin`.
fbTimeMax	FB_ALY_DateTime	Zeitstempel von `fMax`.
fbTimeCurrentInterval	FB_ALY_Timespan	Verstrichene Zeitspanne des aktuellen Intervalls.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Beispiel

```
VAR
    fbMinMaxAvgInterval : FB_ALY_MinMaxAvgInterval_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    tInterval : LTIME := LTIME#20S;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbMinMaxAvgInterval.Configure(tInterval);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbMinMaxAvgInterval.SetChannelValue(nInput);
fbMinMaxAvgInterval.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.15.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.15.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tInterval : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tInterval	LTIME	Intervallzeit

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.15.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.15.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.15.5 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.16 FB_ALY_MovingAvg_1Ch

Der *Moving Average 1Ch* berechnet den gleitenden Durchschnitt, das Minimum und das Maximum der neuesten Eingangswerte in einem Intervall einer bestimmten Länge. Des Weiteren werden die Zeitwerte des Minimums und Maximums angezeigt. Die Berechnung des gleitenden Durchschnitts hängt von der Konfiguration der Parameter *Num Values* und *Startup Behaviour* ab.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MovingAvg_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fMovingAvg: LREAL;
    fMovingMin: LREAL;
    fMovingMax: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fMovingAvg	LREAL	Gleitender-Mittelwert des Eingangswerts.
fMovingMin	LREAL	Gleitendes Minimum des Eingangswerts.
fMovingMax	LREAL	Gleitendes Maximum des Eingangswerts.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbMovingAvg : FB_ALY_MovingAvg_1Ch;
    nNumValues : UDINT := 100;
    eStartupBehaviour : E_ALY_MovingAvgStartupBehaviour := E_ALY_MovingAvgStartupBehaviour.AvgOverEx
isting;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbMovingAvg.Configure(nNumValues, eStartupBehaviour);
END_IF

// Call algorithm
fbMovingAvg.SetChannelValue(nInput);
fbMovingAvg.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.16.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.16.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumValues : UDINT;
    eStartupBehaviour : E_ALY_MovingAvgStartupBehaviour;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumValues	ULINT	Anzahl der Werte, die in die Berechnung des gleitenden Mittelwerts einfließen.
eStartupBehaviour	E_ALY_MovingAvgStartupBehaviour [▶ 355]	Berechnungsverhalten am Anfang der Analyse bei weniger Werten, als in Num Values konfiguriert sind. *ZeroPadding*: Die fehlenden Werte werden mit Nullen gefüllt. *UseFirstValue:* Der erste Wert wird verwendet, bis die Anzahl der Werte Num Values entspricht. *WaitUntilFilled:* Das erste Ergebnis wird berechnet, wenn die Anzahl der Werte Num Values entspricht. *AvgOverExisting:* Der Durchschnitt wird mit den bereits vorhandenen Werten berechnet, bis die Anzahl der Werte Num Values entspricht.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.16.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.16.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.17 FB_ALY_MovingIntervalCounter_1Ch

Der *Moving Interval Counter 1Ch* zählt die Anzahl der ausgelösten Ereignisse innerhalb eines konfigurierten Intervalls. Ein Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert. Die Berechnung wird neu gestartet, wenn die Zeit des Intervalls abgelaufen ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MovingIntervalCounter_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bEdge: BOOL;
    bLimited: BOOL;
    nCountsInInterval: ULINT;
    fbTimeFirstCount: FB_ALY_DateTime;
    fbTimeLastCount: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bEdge	BOOL	TRUE zu dem Zeitpunkt, zu dem das Ereignis ausgelöst wird, ansonsten FALSE.
bLimited	BOOL	TRUE, wenn die Anzahl der Flanken im aktuellen Intervall das eingestellte Count Limit überschreitet.
nCountsInInterval	ULINT	Anzahl der ausgelösten Ereignisse im aktuellen Intervall.
fbTimeFirstCount	FB_ALY_DateTime	Zeitstempel des ersten Ereignisses im aktuellen Intervall.
fbTimeLastCount	FB_ALY_DateTime	Zeitstempel des letzten Ereignisses im aktuellen Intervall.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbMovingIntervalCounter : FB_ALY_MovingIntervalCounter_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_Threshold;
    tInterval : LTIME := LTIME#20S;
    nCountLimit : UDINT := 20;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbMovingIntervalCounter.ConfigureChannel(stThresholdEdge);
    fbMovingIntervalCounter.Configure(tInterval, nCountLimit);
END_IF
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbMovingIntervalCounter.SetChannelValue(nInput);
fbMovingIntervalCounter.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.17.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.17.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tInterval : LTIME;
    nCountLimit : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tInterval	LTIME	Zeitintervall, in dem die Werte berechnet werden sollen.
nCountLimit	UDINT	Grenze der Zählungen im konfigurierten Intervall

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.17.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.17.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.17.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.18 FB_ALY_OverallEquipmentEffectiveness

Der *Overall Equipment Effectiveness (OEE)* berechnet Kennzahlen, die es ermöglichen, den aktuellen Zustand des Herstellungsprozesses mit seinem maximalen Potential zu vergleichen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_OverallEquipmentEffectiveness
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fOEE: LREAL;
    eOeeClass: E_ALY_Classification_3Cls;
    fbTimeOeeEventWarning: FB_ALY_DateTime;
    fbTimeOeeEventAlarm: FB_ALY_DateTime;
    fAvailability: LREAL;
    nPerformance: LREAL;
    fQuality: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fOEE	LREAL	Gesamtanlageneffektivität in Prozent. Sie berechnet sich aus der Multiplikation des Verfügbarkeitsfaktors, des Leistungsfaktors sowie des Qualitätsfaktors.
eOeeClass	E_ALY_Classification_3Cls [▶ 352]	Ergebnis der Einstufung des OEE.
fbTimeOeeEventWarning	FB_ALY_DateTime	Zeitstempel der letzten Einstufung des OEE als Warnung.
fbTimeOeeEventAlarm	FB_ALY_DateTime	Zeitstempel der letzten Einstufung des OEE als Alarm.
fAvailability	LREAL	Verfügbarkeitsfaktor in Prozent. Sie berechnet sich aus dem Verhältnis zwischen der Laufzeit und der Betriebszeit.
fPerformance	LREAL	Leistungsfaktor in Prozent. Er berechnet sich aus dem Verhältnis tatsächlich produzierter Einheiten und der Anzahl produzierter Einheiten im Idealfall.
fQuality	LREAL	Qualitätsfaktor in Prozent. Er berechnet sich aus dem Verhältnis intakter produzierter Einheiten im Verhältnis zu den produzierten Einheiten.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbOEE : FB_ALY_OverallEquipmentEffectiveness;
    fbSystemTime : FB_ALY_GetSystemTime;

    tIdealCycleTime : LTIME := LTIME#1M30S;
    fThresholdLevelOkWarning : LREAL := 90.0;
    fThresholdLevelWarningAlarm : LREAL := 75.0;
    bConfigure : BOOL := TRUE;

    tScheduledTime : LTIME;
    tOperatingTime : LTIME;
    nUnitsProduced : ULINT;
    nDefectiveUnits : ULINT;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbOEE.Configure(tIdealCycleTime, fThresholdLevelOkWarning, fThresholdLevelWarningAlarm);
END_IF

// Call algorithm
fbOEE.SetChannelValue(1, tScheduledTime);
fbOEE.SetChannelValue(2, tOperatingTime);
fbOEE.SetChannelValue(3, nUnitsProduced);
fbOEE.SetChannelValue(4, nDefectiveUnits);
fbOEE.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.18.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.18.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tIdealCycleTime : LTIME;
    fThresholdLevelOkWarning : LREAL;
    fThresholdLevelWarningAlarm : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tIdealCycleTime	BOOL	Ideale Zykluszeit für die Herstellung eine Einheit.
fThresholdLevelOkWarning	LREAL	Die Gesamtanlageneffektivität, die größter als die konfigurierte Schwelle ist, wird in die Klasse *OK* eingestuft. Ist die Gesamtanlageneffektivität kleiner oder gleich der konfigurierten Schwelle, wird sie in die Klasse *Warning* eingestuft.
fThresholdLevelWarningAlarm	LREAL	Die Gesamtanlageneffektivität, die größer als die konfigurierte Schwelle ist, wird in die Klasse *Warning* eingestuft. Ist die Gesamtanlageneffektivität kleiner oder gleich der konfigurierten Schwelle, wird sie in die Klasse *Alarm* eingestuft.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.18.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.18.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

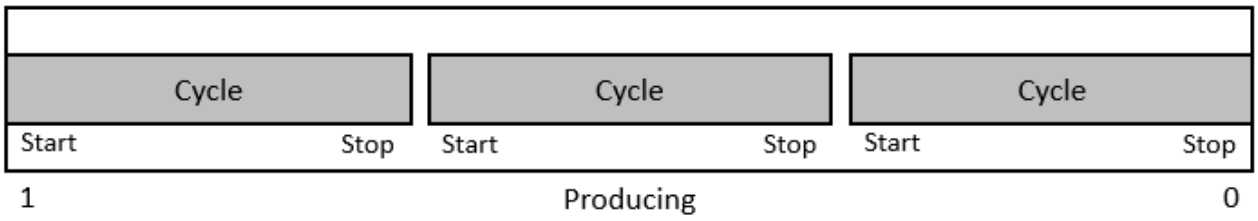
Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Scheduled Time:* Die Betriebszeit errechnet sich aus der Kalenderzeit abzüglich der geplanten Nichtproduktion. 2: *Operating Time*: Die Laufzeit errechnet sich aus der Betriebszeit abzüglich der Stillstandszeiten. 3: *Units Produced*: Entspricht der Anzahl Produzierter Einheiten inklusive defekter Einheiten. 4: *Defective Units*: Entspricht der Anzahl defekter Einheiten.
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.19 FB_ALY_ProductivityDiagnosis_3Ch

Der Algorithmus *Productivity Diagnosis 3Ch* berechnet die Produktivität des Prozesses während eines Produktionsintervalls. Die folgende Abbildung illustriert schematisch den Zusammenhang zwischen dem Produktionsprozess und den einzelnen Produktionszyklen.



Das Produktionsintervall kann durch den Eingang *Is Producing* gestartet und gestoppt werden. Während der Ausführung des Produktionsintervalls werden die Produktionszyklen gezählt. Jeder Produktionszyklus entspricht einem produzierten Teil. Ein Produktionszyklus wird mit einer Flanke an *Start Cycle* gestartet und mit einer Flanke an *Stop Cycle* gestoppt. Die Produktivität über gesamte Produktionsintervall (*Productivity*) wird nach dem Stoppen des Intervalls, wenn das Signal *Is Producing* die Bedingung für *Threshold Level Producing* nicht mehr erfüllt, berechnet. Hierbei werden die vollständigen Produktionszyklen und demnach alle fertig gestellten Teile berücksichtigt. Die Produktivität ergibt sich als das Verhältnis der tatsächlich produzierten Teile pro Zeit und des Sollwerts der in einer bestimmten Zeit zu produzierenden Teile. Der Ausgang *Productivity Last Cycle* errechnet sich aus der für den letzten Produktionszyklus benötigten Zeit bezogen auf die konfigurierte Zeit für ein Teil. Eventuelle Pausenzeiten zwischen den Zyklen werden dabei nicht berücksichtigt. Der Ausgang *Expected Productivity* schätzt die Gesamtproduktivität während des Produktionsintervalls. Dafür erfolgt eine Hochrechnung der bisherigen Produktionszeit auf die Gesamtproduktivität für den Sollwert der zu produzierenden Teile. Der Algorithmus kann mit dem Sollwert der produzierten Teile *(Produced Pieces)* innerhalb eines konfigurierten Intervalls *(Production Time)* konfiguriert werden, z. B. 1 Teil in 30 Sekunden oder 50 Teile pro Stunde.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ProductivityDiagnosis_3Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bProducing: BOOL;
    bCycleFinished: BOOL;
    fProductivity: LREAL;
    fProductivityLastCycle: LREAL;
    fExpectedProductivity: LREAL;
    fbElapsedTime: FB_ALY_Timespan;
    nProductionCycles: ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bProducing	BOOL	Gibt an, dass das Produktionsintervall aktiv ist.
bCycleFinished	BOOL	TRUE, wenn ein Produktionszyklus abgeschlossen worden ist, ansonsten FALSE.
fProductivity	LREAL	Produktivität des gesamten Produktionsintervalls in Prozent.
fProductivityLastCycle	LREAL	Produktivität des letzten Produktionszyklus in Prozent.
fExpectedProductivity	LREAL	Schätzt die Produktivität des Produktionsintervalls. Angabe in Prozent.
fbElapsedTime	FB_ALY_Timespan	Zeitspanne seit dem Start des Produktionsintervalls.
nProductionCycles	ULINT	Anzahl der vollständigen Produktionszyklen im aktuellen Produktionsintervall.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbProductivityDiagnosis : FB_ALY_ProductivityDiagnosis_3Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdIsProducing : ST_ALY_Threshold;
    stThresholdStartCycle : ST_ALY_Threshold;
    stThresholdStopCycle : ST_ALY_Threshold;
    nNumProducedPieces : ULINT := 40;
    tProductionTime : LTIME := LTIME#1H;
    bConfigure : BOOL := TRUE;
    bIsProducing : BOOL;
    bStartProductionCycle : BOOL;
    bStopProductionCycle : BOOL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdIsProducing.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdIsProducing.fThreshold := 1;
    stThresholdStartCycle.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdStartCycle.fThreshold := 1;
    stThresholdStopCycle.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdStopCycle.fThreshold := 1;

    fbProductivityDiagnosis.ConfigureChannel(1, stThresholdIsProducing);
    fbProductivityDiagnosis.ConfigureChannel(2, stThresholdStartCycle);
    fbProductivityDiagnosis.ConfigureChannel(3, stThresholdStopCycle);
    fbProductivityDiagnosis.Configure(nNumProducedPieces, tProductionTime);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbProductivityDiagnosis.SetChannelValue(1, bIsProducing);
fbProductivityDiagnosis.SetChannelValue(2, bStartProductionCycle);
fbProductivityDiagnosis.SetChannelValue(3, bStopProductionCycle);
fbProductivityDiagnosis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.19.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.19.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumProducedPieces : ULINT;
    tProductionTime : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumProducedPieces	ULINT	Sollwert der produzierten Teile während des konfigurierten Zeitintervalls (*Production Time*).
tProductionTime	LTIME	Zeitintervall, in dem nNumProducedPieces produziert werden.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.19.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    stThreshold : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Is producing* (Ebene) 2: *Start production cycle* (Flanke) 2: Stop production cycle (Ebene)
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle. Abhängig von nChannel.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.19.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.19.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Is producing (Ebene) 2: Start production cycle (Flanke) 2: Stop production cycle (Ebene)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.20 FB_ALY_ProductivityInterval_1Ch

Der Algorithmus *Productivity Interval 1Ch* berechnet die Produktivität des Prozesses während eines vorgegebenen Intervalls. Das Intervall kann durch die Eingänge *tTimeStart* und *tTimeStop* definiert werden. Bei der Ausführung werden die produzierten Teile berücksichtigt. Ein produziertes Element wird gezählt, wenn eine Flanke am Eingang anliegt. Als Ausgangswerte werden die geschätzte Produktivität des aktuellen Intervalls sowie die Produktivität des letzten vollständigen Intervalls bereitgestellt. Der Algorithmus kann mit dem Sollwert der produzierten Teile innerhalb eines vorgegebenen Intervalls konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ProductivityInterval_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bWithinInterval: BOOL;
    fbCurrentTimestamp: FB_ALY_DateTime;
    fbIntervalLength: FB_ALY_Timespan;
    fbElapsedTime: FB_ALY_Timespan;
```

```
    fbRemainingTime: FB_ALY_Timespan;
    nProducedInInterval: ULINT;
    nRemainingInInterval: ULINT;
    fCurrentProductivity: LREAL;
    fExpectedProductivity: LREAL;
    fLastFullPeriodProductivity: LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bWithinInterval	BOOL	Gibt an, ob sich die aktuelle Uhrzeit innerhalb des Intervalls befindet.
fbCurrentTimestamp	FB_ALY_DateTime	Aktueller Zeitstempel.
fbIntervalLength	FB_ALY_Timespan	Länge des Intervalls.
fbElapsedTime	FB_ALY_Timespan	Verstrichene Zeit innerhalb des Intervalls.
fbRemainingTime	FB_ALY_Timespan	Verbleibende Zeit innerhalb des Intervalls.
nProducedInInterval	ULINT	Produzierte Teile innerhalb des Intervalls.
nRemainingInInterval	ULINT	Verbleibende Teile innerhalb des Intervalls.
fCurrentProductivity	LREAL	Aktuelle Produktivität des Intervalls in Prozent. Berücksichtigt die Länge des Intervalls, die bereits verstrichene Zeit, die zu produzierenden Teile sowie die bereits produzierten Teile. Die Ausgabe erfolgt in Prozent.
fExpectedProductivity	LREAL	Geschätzte Produktivität des Intervalls in Prozent. Zur Ermittlung der möglichen produzierbaren Teile innerhalb der verbleibenden Zeit wird die Produktionszeit des letzten Teils verwendet.
fLastFullPeriodProductivity	LREAL	Produktivität des letzten komplett durchlaufenden Intervalls in Prozent. Wird nur berechnet, wenn das Intervall vollständig durchlaufen wurde.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbProductivityInterval : FB_ALY_ProductivityInterval_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThreshold : ST_ALY_Threshold;
    nExpectedPieces : ULINT := 40;
    bConfigure : BOOL := TRUE;
    bPieceProduced : BOOL;
    tTimeStart : LTIME := LTIME#8H;
    tTimeStop : LTIME := LTIME#16H;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThreshold.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThreshold.fThreshold := 1;

    fbProductivityInterval.ConfigureChannel(stThreshold);
    fbProductivityInterval.Configure(nExpectedPieces);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbProductivityInterval.SetChannelValue(bPieceProduced);
fbProductivityInterval.Call(tTimeStart, tTimeStop, fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.20.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimeStart : LTIME;
    tTimeStop : LTIME;
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimeStart	LTIME	Startzeitpunkt des vorgegebenen Intervalls. Ermöglicht die Vorgabe des Intervalls aus Prozessdaten.
tTimeStop	LTIME	Endzeitpunkt des vorgegebenen Intervalls. Ermöglicht die Vorgabe des Intervalls aus Prozessdaten.
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.20.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nExpectedPieces: ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nExpectedPieces	ULINT	Vorgabe der zu produzierenden Teile innerhalb der definierten Zeitspanne.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.20.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThreshold : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle. Durch diese Einstellung wird festgelegt, wann ein produziertes Teil gezählt wird.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.20.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.20.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.21 FB_ALY_SignalGenerator_1Ch

Der *Signal Generator 1Ch* kann zur Erzeugung verschiedener Signalverläufe genutzt werden. Die Signalform, die Frequenz, die Amplitude sowie der Offset können individuell eingestellt werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_SignalGenerator_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
```

```
    bNewResult: BOOL;
    bConfigured: BOOL;
    fSignal: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fSignal	LREAL	Ausgabe des konfigurierten Signals

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.

Beispiel

```
VAR
    fbSignalGenerator : FB_ALY_SignalGenerator_1Ch;
    eFunctionType : E_ALY_FunctionType := E_ALY_FunctionType.Sine;
    fSampleRate : LREAL := 1000.0;
    fFrequency : LREAL := 50.0;
    fAmplitude : LREAL := 100.0;
    fOffset : LREAL := 0.0;
    bConfigure : BOOL := TRUE;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbSignalGenerator.Configure(eFunctionType, fSampleRate, fFrequency, fAmplitude, fOffset);
END_IF

// Call algorithm
fbSignalGenerator.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.21.1 Call

Aufrufen des Algorithmus.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.21.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eFunctionType : E_ALY_FunctionType;
    fSampleRate : LREAL;
    fFrequency : LREAL;
    fAmplitude : LREAL;
    fOffset : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eFunctionType	E_ALY_FunctionType [▶ 357]	Funktionstyp des generierten Signals. *Const:* Konstant *Rectangle:* Rechteckfunktion *Sawtooth:* Sägezahnfunktion *Sine:* Sinusfunktion *Triangle:* Dreiecksfunktion
fSampleRate	LREAL	Sample Rate des zu analysierenden Systems.
fFrequency	LREAL	Frequenz des generierten Signals.
fAmplitude	LREAL	Konfiguration der Signalamplitude.
fOffset	LREAL	Konstanter Versatz des generierten Signals.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.21.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.22 FB_ALY_TimeClock_1Ch

Der *Time Clock 1Ch* führt eine Zeitschaltung aus, die mit Einschaltzeit, Ausschaltzeit und den Tagen der Woche, an denen die Zeitschaltung aktiv sein soll, konfiguriert werden kann. Als Referenzwert ist ein Zeitstempel erforderlich, da der Algorithmus einen Zeitkontext benötigt, in dem er arbeiten soll.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_TimeClock_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    fbTimeUntilNextSwitch: FB_ALY_Timespan
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	`TRUE`, wenn die aktuelle Zeit innerhalb der konfigurierten On-Zeit liegt.
fbTimeUntilNextSwitch	FB_ALY_Timespan	Zeitspanne bis zur nächsten Schaltung.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.

Beispiel

```
VAR
    fbTimeClock : FB_ALY_TimeClock_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    tTimeOn : LTIME := LTIME#8H;
    tTimeOff : LTIME := LTIME#16H;
    nDayOfWeekMask : WORD := E_ALY_DayOfWeekMask.MondayToFriday;
    bConfigure : BOOL := TRUE;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbTimeClock.Configure(tTimeOn, tTimeOff, nDayOfWeekMask);
END_IF
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbTimeClock.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.22.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.22.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
    tTimeOn : LTIME;
    tTimeOff : LTIME;
    nDayOfWeekMask : WORD;
VAR_INPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimeOn	LTIME	Zeit, zu der sich die Zeituhr einschalten soll
tTimeOff	LTIME	Zeit, zu der sich die Zeituhr ausschalten soll
nDayOfWeekMask	WORD	Bitmaske zur Auswahl des Wochentags, an dem der Zeitschaltuhrprogramm ausgeführt werden soll. 1: *Sonntag* 2: *Montag* 4: *Dienstag* 8: *Mittwoch* 16: *Donnerstag* 32: *Freitag* 64: *Samstag* Die Aufzählung E_ALY_DayOfWeekMask [▶ 354] kann verwendet werden. Mehrere Tage können mit dem Operator OR kombiniert werden.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.22.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.23 FB_ALY_Timer_1Ch

Der *Timer 1Ch* startet einen Timer, der nach Timermodus und Intervall konfiguriert werden kann. Gemäß dem spezifischen Timermodus wird der Timer gestartet, wenn die konfigurierte Bedingung *TRUE* (TON, TP) oder *FALSE* (TOF) wird.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Timer_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOut: BOOL;
    fbTimeElapsed: FB_ALY_Timespan;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bOut	BOOL	Ausgabewert, der von dem konfigurierten Timer betroffen ist.
fbTimeElapsed	FB_ALY_Timespan	Abgelaufene Zeit.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbTimer : FB_ALY_Timer_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    eTimerMode : E_ALY_TimerMode := E_ALY_TimerMode.TON;
    tInterval : LTIME := LTIME#20S;
    bConfigure : BOOL := TRUE;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbTimer.Configure(eTimerMode, tInterval);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbTimer.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.23.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.23.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eTimerMOde : E_ALY_TimerMode;
    tInterval : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eTimerMode	E_ALY_TimerMode [▶ 356]	Modus des Timers: *TON:* Der TON-Timer ist ein Einschaltverzögerungstimer, der den Ausgang aktiviert, nachdem die Schwellenbedingung *TRUE* wird und die im Intervall festgelegte Zeitspanne abgelaufen ist. *TOF:* Der TOF-Timer ist ein Ausschaltverzögerungstimer, der den Ausgang deaktiviert, nachdem die Schwellenbedingung *FALSE* wird und die im Intervall festgelegte Zeitspanne abgelaufen ist. *TP:* Der TP-Timer ist ein Impulsgeber, der den Ausgang für die im Intervall festgelegte Zeit aktiviert, nachdem die Schwellenbedingung *TRUE* wird.
tInterval	LTIME	Intervallzeit, in der die Ereignisse berücksichtigt werden sollen

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.23.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdLevel : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdLevel	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Eingangsbedingung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.23.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.23.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.23.6 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.24 FB_ALY_TimingAnalysis_1Ch

Der *Timing Analysis 1Ch* misst Zeitdifferenzen zwischen On- und Off-Perioden und zählt die Anzahl der On-Perioden. Die On-Periode beginnt, wenn die Bedingung von Operator und Schwelle erfüllt ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_TimingAnalysis_1Ch
VAR_INPUT
    bPersistent: BOOL;
```

```
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bIsOn: BOOL;
    nSwitchedOn: ULINT;
    fbTimeCurrentInterval: FB_ALY_Timespan;
    fbTimeOnTotal: FB_ALY_Timespan;
    fbTimeOffTotal: FB_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bIsOn	BOOL	TRUE innerhalb der Zeitspanne zwischen dem On-Ereignis und dem Off-Ergebnis, andernfalls FALSE.
nSwitchedOn	ULINT	Zählt die Anzahl der ausgelösten On-Ereignisse.
fbTimeCurrentInterval	FB_ALY_Timespan	Zeitspanne des aktuellen Intervalls.
fbTimeOnTotal	FB_ALY_Timespan	Gesamtzeit, für die bIsOn=TRUE ist.
fbTimeOffTotal	FB_ALY_Timespan	Gesamtzeit, für die bIsOn=FALSE ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbTimingAnalysis : FB_ALY_TimingAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stThresholdEdge : ST_ALY_Threshold;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbTimingAnalysis.ConfigureChannel(stThresholdEdge);
    fbTimingAnalysis.Configure();
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbTimingAnalysis.SetChannelValue(nInput);
fbTimingAnalysis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.1.24.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.1.24.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.24.3 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    stThresholdLevel : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stThresholdLevel	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und verwendete Schwelle für On-Zustand.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.1.24.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.1.24.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2 Classification

5.1.1.2.1 FB_ALY_BandwidthClassificator_1Ch

Der *Bandwidth Classifier 1Ch* bestimmt, ob das Eingangssignal innerhalb der konfigurierten Grenzen liegt oder kleiner oder größer als die Grenzen ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_BandwidthClassificator_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    eClass: E_ALY_Classification_Bounds;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
eClass	E_ALY_Classification_Bounds [▶ 352]	Klasse, zu der die Eingangswerte gehören (*WithinBounds/ Smaller/Bigger*).
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel der letzten Änderung des Klassifikationsergebnisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbBandwidthClassificator : FB_ALY_BandwidthClassificator_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fLowerBound : LREAL := 10.0;
    fUpperBound : LREAL := 20.0;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbBandwidthClassificator.Configure(fLowerBound, fUpperBound);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbBandwidthClassificator.SetChannelValue(nInput);
fbBandwidthClassificator.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.1.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fLowerBound : LREAL;
    fUpperBound : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fLowerBound	LREAL	Untere Grenze für den Vergleich.
fUpperBound	LREAL	Obere Grenze für den Vergleich.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.2 FB_ALY_BandwidthClassificator_3Ch

Der *Bandwidth Classifier 3Ch* bestimmt, ob das Eingangssignal innerhalb der Grenzen liegt oder kleiner oder größer als die Grenzen ist. Die Grenzen können mit Eingangssignalen konfiguriert werden, sodass es möglich ist, Kurvenverläufe als unteres und oberes Band zu verwenden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_BandwidthClassificator_3Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    eClass: E_ALY_Classification_Bounds;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
eClass	E_ALY_Classification_Bounds [▶ 352]	Klasse, zu der die Eingangswerte gehören (*WithinBounds/ Smaller/Bigger*).
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel der letzten Änderung des Klassifikationsergebnisses.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbBandwidthClassificator : FB_ALY_BandwidthClassificator_3Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fInputLowerBound : LREAL;
    fInputUpperBound : LREAL;
    nInput : INT;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbBandwidthClassificator.SetChannelValue(1, nInput);
fbBandwidthClassificator.SetChannelValue(2, fInputLowerBound);
fbBandwidthClassificator.SetChannelValue(3, fInputUpperBound);
fbBandwidthClassificator.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.2.2 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Data in* 2: Lower bound 3: *Upper bound*
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.3 FB_ALY_CurveSketcher_1Ch

Curve Sketcher 1Ch identifiziert Umkehrungen (*Peaks* und *Täler*) in einem Eingangsdatenstrom. Des Weiteren können lokale Maxima der absoluten Differenz zwischen zwei aufeinanderfolgenden Werten (bezeichnet als *Delta*) identifiziert werden. Analog zu einer kontinuierlichen Kurve entsprechen die identifizierten Peaks und Täler lokalen Maxima und Minima. Das Delta entspricht der Steigung, sodass ein Maximum der absoluten Werte des Deltas mit einem Wendepunkt verbunden werden kann.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_CurveSketcher_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fLastPeak: LREAL;
    fbTimeLastPeak: FB_ALY_DateTime;
    nCountPeaks: ULINT;
    fLastValley: LREAL;
    fbTimeLastValley: FB_ALY_DateTime;
    nCountValleys: ULINT;
    fValueAtMaxDelta: LREAL;
    fMaxDelta: LREAL;
    fbTimeMaxDelta: FB_ALY_DateTime;
    nCountMaxDelta: ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fLastPeak	LREAL	Y-Wert des zuletzt identifizierten Peaks.
fbTimeLastPeak	FB_ALY_DateTime	Zeitstempel von `fLastPeak`.
nCountPeaks	ULINT	Gesamtanzahl der gezählten Peaks.
fLastValley	LREAL	Y-Wert des zuletzt identifizierten Tals.
fbTimeLastValley	FB_ALY_DateTime	Zeitstempel von `fLastValley`.
nCountValleys	ULINT	Anzahl der erkannten Täler.
fValueAtMaxDelta	LREAL	Eingangsvariable, die von dem zuletzt erkannten Maximum von Delta geleitet wird. Der Wert Delta ist die Differenz zwischen diesem Wert und dem Eingangswert von einem Zyklus zuvor.
fMaxDelta	LREAL	Das zuletzt erkannte lokale Maximum der absoluten Differenz zwischen zwei aufeinanderfolgenden Werten im Eingangsstrom.
fbTimeMaxDelta	FB_ALY_DateTime	Zeitstempel von `fValueAtMaxDelta`.
nCountMaxDelta	ULINT	Gesamtanzahl der gezählten lokalen Maxima von Delta.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbCurveSketcher : FB_ALY_CurveSketcher_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fThresholdReversal : LREAL := 0.0;
    bCalcInflection : BOOL := FALSE;
    fThresholdDelta : LREAL := 10.0;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbCurveSketcher.Configure(fThresholdReversal, bCalcInflection, fThresholdDelta);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbCurveSketcher.SetChannelValue(nInput);
fbCurveSketcher.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.3.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.3.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

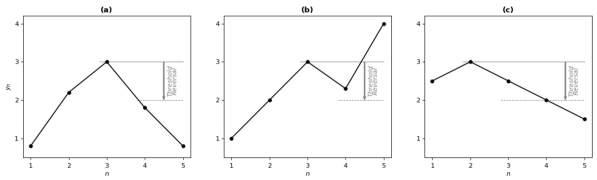
Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.3.4 Configure

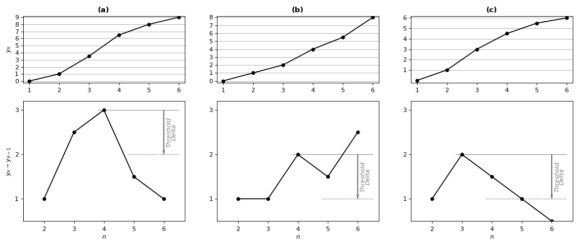
Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Konfigurationsoptionen

- **Calculate Inflection:** Boolsches Flag. Maxima der Änderungsrate werden nur identifiziert, wenn dieses Flag *True* ist. Anderenfalls werden die Werte für *Count Max Delta, Max Delta*, *Time Max Delta* und *Value at Max Delta* nicht berechnet.
- **Threshold Reversal:** Schwellenwert für die Identifizierung von Umkehrungen. Umkehrungen werden nur erkannt, wenn ihre Differenz zur nächsten Umkehrung den Wert von Threshold Reversal überschreitet.
  Nachfolgend sind drei Beispiele für die Identifizierung von Peaks mit dem Parameter *Threshold Reversal* aufgeführt.
  (a) Der Wert y_3 wird direkt nach der Verarbeitung des Werts y_4 als Peak identifiziert, da die Differenz zwischen y_3 und y_4 größer ist als *Threshold Reversal*.
  (b) Der Wert y_3 wird nicht als Peak identifiziert, da die Differenz zwischen y_3 und y_4 kleiner ist als *Threshold Reversal* und die Kurve nach y_4 wieder ansteigt.
  (c) Der Wert y_2 wird nach der Verarbeitung des Werts y_5 als Peak identifiziert, da die Differenz zwischen y_2 und y_5 *Threshold Reversal* überschreitet. Der Wert y_2 kann vorher nicht als Peak identifiziert werden, da die Differenz zwischen y_2 und y_3 (y_4) kleiner als/gleich *Threshold Reversal* ist und nicht bekannt ist, ob die Werte weiter sinken.



- **Threshold Delta:** Schwellenwert für die Identifizierung der Delta-Maxima. Maxima der absoluten Differenz von zwei aufeinanderfolgenden Werten (Delta) werden nur erkannt, wenn die Differenz zwischen aufeinanderfolgenden Deltas *Threshold Delta* überschreitet.
  Nachfolgend sind drei Beispiele für die Identifizierung der Delta-Maxima mit dem Parameter *Threshold Delta* aufgeführt. Die oberen Diagramme zeigen die ursprünglichen Eingangssignale, die unteren das zugehörige Delta.
  (a) Der Wert y_4 wird nach der Verarbeitung des Werts y_5 als Maximum identifiziert, da die Differenz der zwei Deltas *Threshold Delta* überschreitet.
  (b) Es wird kein Maximum identifiziert, da die Differenz zwischen den Deltas kleiner ist als *Threshold Delta*.
  (c) Der Wert y_3 wird nach der Verarbeitung des Werts y_6 als Maximum identifiziert.



Unabhängig von *Threshold Delta* wird mindestens ein Maximum des Deltas zwischen zwei Umkehrungen erkannt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fThresholdReversal : LREAL;
    bCalcInflection : BOOL;
    fThresholdDelta : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fThresholdReversal	LREAL	Schwelle für die Peakanalyse
bCalcInflection	BOOL	Auswertung von Max Delta
fThresholdDelta	LREAL	Schwelle für die Delta-Analyse

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.4 FB_ALY_Histogram_1Ch

Histogram 1Ch berechnet die Verteilung eines Einzelkanal-Eingangswerts zyklisch. Er kann mit minimaler Klasse, maximaler Klasse und Gesamtanzahl der Klassen konfiguriert werden. Die Dimension des Ausgangs-Arrays ist die Anzahl der Klassen + 2, da Werte, die kleiner als die minimale Klasse sind, im ersten Array-Element gespeichert werden und Werte, die größer als die maximale Klasse sind, im letzten Array-Element gespeichert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Histogram_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nNumValues: ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
nNumValues	ULINT	Anzahl der Werte, die im Histogramm aufgenommen wurden.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
GetResults()	Local	Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbHistogram : FB_ALY_Histogram_1Ch;
    eHistMode : E_ALY_HistMode := E_ALY_HistMode.Absolute;
    nBins : UDINT := 20;
    fMinBinned : LREAL := 1;
    fMaxBinned : LREAL:= 200;
    bConfigure : BOOL := TRUE;
    nInput : INT;
    aHistArrayOut : ARRAY[0..21] OF ULINT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbHistogram.Configure(eHistMode, nBins, fMinBinned, fMaxBinned);
END_IF

// Call algorithm
fbHistogram.SetChannelValue(nInput);
fbHistogram.Call(ADR(aHistArrayOut), SIZEOF(aHistArrayOut));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.4.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    pHistArrayOut : PVOID;
    nHistArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHistArrayOut	PVOID	Zeiger auf Histogramm-Array. Dim.: nBins + 2 Der Datentyp hängt vom konfigurierten Modus ab. *Abs*: ULINT *Rel*: LREAL
nHistArrayOutSize	UDINT	Größe des Histogramm-Arrays.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.4.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.4.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.4.4 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eHistMode : E_ALY_HistMode;
    nBins : UDINT;
    fMinBinned : LREAL;
    fMaxMinned : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eHistMode	E_ALY_HistMode [▶ 357]	Die Betriebsart des Histogramms. *Abs*: Absolute Werte *Rel*: Relative Werte, um die prozentuale Verteilung anzuzeigen.
nBins	UDINT	Gesamtanzahl der Klassen für das Histogramm, die berechnet werden soll.
fMinBinned	LREAL	Minimaler Wert, der analysiert werden soll.
fMaxBinned	LREAL	Maximaler Wert, der analysiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.4.5 GetResults

Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.

Syntax

Definition:

```
METHOD GetResults : BOOL
VAR_INPUT
    pHistArrayOut: POINTER TO ULINT;
    nHistArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHistArrayOut	POINTER TO ULINT	Zeiger auf Histogramm-Array. Dim.: nBins + 2
nHistArrayOutSize	UDINT	Größe des Histogramm-Arrays. Dim.: nBins + 2

Rückgabewert

Name	Typ	Beschreibung
GetResults	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.2.5 FB_ALY_SectionTimer_1Ch

Der *Section Timer 1Ch* berechnet die Zeitspanne, in der sich der Eingang im Bereich jedes konfigurierten Abschnitts befindet. Er kann mit der Anzahl der Abschnitte und den Grenzen jedes Abschnitts konfiguriert werden. Jeder Abschnitt wird mit einer unteren Grenze (größer als oder gleich) und einer oberen Grenze (kleiner als) definiert. Die untere Grenze des folgenden Abschnitts wird durch die vorherige obere Grenze gesetzt. Werte, die kleiner als die minimale Grenze sind, werden im ersten Array-Element gespeichert. Werte, die gleich oder größer als die maximale Grenze sind, werden im letzten Array-Element gespeichert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_SectionTimer_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nSection: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
nSection	UDINT	Aktuell klassifizierter Abschnitt des Eingangs.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
GetResults()	Local	Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
SetSections()	Local	Festlegen einzelner Abschnitte.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInitial()	Local	Methode zur Festlegung von Startabschnittszeiten.

Beispiel

```
VAR
    fbSectionTimer : FB_ALY_SectionTimer_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nNumSections : UDINT := 4;
    fFirstLowerBorder : LREAL := 10;
    aUpperBorders : ARRAY[1..4] OF LREAL := [20,30,40,50];
    bConfigure : BOOL := TRUE;
    nInput : INT;
    aTimespansOut : ARRAY [0..5] OF LINT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbSectionTimer.Configure(nNumSections);
    fbSectionTimer.SetSections(fFirstLowerBorder, ADR(aUpperBorders), SIZEOF(aUpperBorders));
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbSectionTimer.SetChannelValue(nInput);
fbSectionTimer.Call(fbSystemTime.tSystemTime, ADR(aTimespansOut), SIZEOF(aTimespansOut));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.5.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    pTimespanArrayOut : POINTER TO LINT;
    nTimespanArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel.
pTimespanArrayOut	POINTER TO LINT	Zeiger auf Zeitspannen-Array. Diese Werte werden: Dim.: nNumSections + 2
nTimespanArrayOutSize	UDINT	Größe des Zeitspannen-Arrays. Dim.: nNumSections + 2

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.5.2 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    pTimespanArrayIn : POINTER TO ULINT;
    nTimespanArrayInSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pTimespanArrayIn	POINTER TO ULINT	Zeiger auf Zeitspannen-Array, das als Ausgangswerte festgelegt werden soll: Dim.: nNumSections + 2
nTimespanArrayInSize	UDINT	Größe des Zeitspannen-Arrays, das als Ausgangswerte festgelegt werden soll. Dim.: nNumSections + 2

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.5.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumSections : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumSections	UDINT	Konfigurieren der Anzahl der Abschnitte.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.5.4 SetSections

Festlegen von Abschnitten, nachdem der FB konfiguriert worden ist. Jeder Abschnitt wird mit einer unteren Grenze (größer als oder gleich) und einer oberen Grenze (kleiner als) definiert. Die untere Grenze des folgenden Abschnitts wird durch die vorherige obere Grenze gesetzt. Die erste untere Grenze wird separat festgelegt.

Syntax

Definition:

```
METHOD SetSections : BOOL
VAR_INPUT
    fFirstLowerBorder : LREAL;
    pSectionConfigArray : POINTER TO LREAL;
    nSectionConfigArraySize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFirstLowerBorder	LREAL	Konfigurieren der ersten unteren Grenze.
pSectionConfigArray	POINTER TO LREAL	Zeiger auf Array der oberen Abschnittsgrenzen: Dim.: nNumSections
nSectionConfigArraySize	UDINT	Größe des Abschnittskonfigurations-Arrays. Dim.: nNumSections

Rückgabewert

Name	Typ	Beschreibung
SetSections	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.2.5.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.5.6 GetResults

Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.

Syntax

Definition:

```
METHOD GetResults : BOOL
VAR_INPUT
    pTimespanArrayOut : POINTER TO LINT;
    nTimespanArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pTimespanArrayOut	POINTER TO LINT	Zeiger auf Zeitspannen-Array. Diese Werte werden: Dim.: nNumSections + 2
nTimespanArrayOutSize	UDINT	Größe des Zeitspannen-Arrays. Dim.: nNumSections + 2

Rückgabewert

Name	Typ	Beschreibung
GetResults	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.2.5.7 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.6 FB_ALY_StateHistogram_1Ch

Der Algorithmus *State Histogram 1Ch* zählt, wie oft das Eingangssignal (INT) einen bestimmten Wert zwischen dem konfigurierten Minimum und Maximum einnimmt und zeigt die Verteilung in einem Histogramm. Der erste Balken stellt die Grenzwerte dar, die kleiner als das Minimum sind, und der letzte Balken stellt die Grenzwerte dar, die größer als das Maximum sind. Der *State Histogram 1Ch* eignet sich für Zustandsmaschinen, um zu zeigen, wie oft die verschiedenen Zustände ausgeführt werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StateHistogram_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nNumValues: ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
nNumValues	ULINT	Anzahl der Werte, die im Histogramm aufgenommen wurden.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
GetResults()	Local	Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInital()	Local	Methode zur Festlegung von Starthistogrammwerten.

Beispiel

```
VAR
    fbStateHistogram : FB_ALY_StateHistogram_1Ch;
    eStateHistMode : E_ALY_StateHistMode := E_ALY_StateHistMode.Absolute;
    nMin : LINT := 1;
    nMax : LINT:= 20;
    bConfigure : BOOL := TRUE;
    nInput : INT;
    aHistArrayOut : ARRAY[0..21] OF ULINT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbStateHistogram.Configure(eStateHistMode, nMin, nMax);
END_IF

// Call algorithm
fbStateHistogram.SetChannelValue(nInput);
fbStateHistogram.Call(ADR(aHistArrayOut), SIZEOF(aHistArrayOut));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.6.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    pHistArrayOut : PVOID;
    nHistArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHistArrayOut	PVOID	Zeiger auf Histogramm-Array. Der Datentyp hängt vom konfigurierten Modus ab. *Abs*: ULINT *Rel*: LREAL
nHistArrayOutSize	UDINT	Größe des Histogramm-Arrays. Dim.: nMax – nMin + 3

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.6.2 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.6.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eStateHistMode : E_ALY_StateHistMode;
    nMin : LINT;
    nMax : LINT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eStateHistMode	E_ALY_StateHistMode [▶ 355]	Die Betriebsart des Histogramms. *Abs*: Absolute Werte *Rel*: Relative Werte, um die prozentuale Verteilung anzuzeigen.
nMin	LINT	Minimaler Wert, der analysiert werden soll.
nMax	LINT	Maximaler Wert, der analysiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.6.4 SetInital

Initialisieren interner Werte des Algorithmus.

Syntax

Definition:

```
METHOD SetInitial : BOOL
VAR_INPUT
    nCountOn : ULINT;
    nCountOff : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eStateHistMode	E_ALY_StateHistMode [▶ 355]	Die Betriebsart des Histogramms. Abs: Absolute Werte Rel: Relative Werte, um die prozentuale Verteilung anzuzeigen.
nNumValues	ULINT	Anzahl der Werte, die im Histogramm gespeichert werden.
pHistArrayIn	PVOID	Zeiger auf Histogramm-Array, das als Start festgelegt werden soll. Der Datentyp hängt vom konfigurierten Modus ab. Abs: ULINT Rel: LREAL
nHistArrayInSize	UDINT	Größe des Histogramm-Arrays, das als Start festgelegt werden soll. Dim.: nMax – nMin + 3

Rückgabewert

Name	Typ	Beschreibung
SetInitial	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.6.5 GetResults

Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen.

Syntax

Definition:

```
METHOD GetResults : BOOL
VAR_INPUT
    pHistArrayOut: PVOID;
    nHistArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHistArrayOut	PVOID	Zeiger auf Histogramm-Array. Der Datentyp hängt vom konfigurierten Modus ab. Abs: ULINT Rel: LREAL
nHistArrayOutSize	UDINT	Größe des Histogramm-Arrays. Dim.: nMax – nMin + 3

Rückgabewert

Name	Typ	Beschreibung
GetResults	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.2.7 FB_ALY_ThresholdClassificator_1Ch

Der *Threshold Classifier 1Ch* stuft die Eingangswerte in drei verschiedene Klassen ein. *OK*, *Warning* und *Alarm* gemäß den konfigurierten Schwellen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ThresholdClassificator_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    eClass: E_ALY_Classification_3Cls;
    fbTimeLastEventWarning: FB_ALY_DateTime;
    fbTimeLastEventAlarm: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
eClass	E_ALY_Classification_3Cls [▶ 352]	Klassifikationsergebnis.
fbTimeLastEventWarning	FB_ALY_DateTime	Zeitstempel der letzten Klassifikation als Warnung.
fbTimeLastEventAlarm	FB_ALY_DateTime	Zeitstempel der letzten Klassifikation als Alarm.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbThresholdClassificator : FB_ALY_ThresholdClassificator_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fThresholdLevelOkWarning : LREAL := 10;
    fThresholdLevelWarningAlarm : LREAL:= 20;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbThresholdClassificator.Configure(fThresholdLevelOkWarning, fThresholdLevelWarningAlarm);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbThresholdClassificator.SetChannelValue(nInput);
fbThresholdClassificator.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.7.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.7.2 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.7.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fThresholdLevelOkWarning : LREAL;
    fThresholdLevelWarningAlarm : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fThresholdLevelOkWarning	LREAL	Die Eingangswerte, die kleiner als die konfigurierte Schwelle sind, werden in die Klasse *OK* eingestuft. Die Eingangswerte, die größer oder gleich als die konfigurierte Schwelle sind, werden in die Klasse *Warnung* eingestuft.
fThresholdLevelWarningAlarm	LREAL	Die Eingangswerte, die kleiner als die konfigurierte Schwelle sind, werden in die Klasse *Warnung* eingestuft. Die Eingangswerte, die größer oder gleich als die konfigurierte Schwelle sind, werden in die Klasse *Alarm* eingestuft.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.7.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.8 FB_ALY_ThresholdStringClassificator_1Ch

Der Algorithmus *Threshold String Classifier 1Ch* stuft die Eingangswerte in drei verschiedene Klassen gemäß den konfigurierten Schwellen ein. Die Klassennamen (Output String) können einzeln als *String 1*, *String 2* und *String 3* konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ThresholdStringClassificator_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    sResult: STRING(255) := '';
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
sResult	STRING(255)	Klassifikationsergebnis
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel der letzten Einstufungsänderung.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbThresholdStringClassificator : FB_ALY_ThresholdStringClassificator_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fThresholdLevel12 : LREAL := 10;
    fThresholdLevel23 : LREAL:= 20;
    sResult1 : STRING := 'This string is set at level 1 ( < 10)';
    sResult2 : STRING := 'This string is set at level 2 ( >= 10)';
    sResult3 : STRING := 'This string is set at level 3 ( >= 20)';
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbThresholdStringClassificator.Configure(fThresholdLevel12, fThresholdLevel23, sResult1, sResult
2, sResult3);
END_IF
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbThresholdStringClassificator.SetChannelValue(nInput);
fbThresholdStringClassificator.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.8.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.8.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.8.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.8.4 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fThresholdLevel12 : LREAL;
    fThresholdLevel23 : LREAL;
    sStringLevel1 : STRING(255);
    sStringLevel2 : STRING(255);
    sStringLevel3 : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fThresholdLevel12	LREAL	Die Eingangswerte, die kleiner als die konfigurierte Schwelle sind, werden in die *erste* Klasse eingestuft. Die Eingangswerte, die größer oder gleich als die konfigurierte Schwelle sind, werden in die *zweite* Klasse eingestuft.
fUpperBound	LREAL	Die Eingangswerte, die kleiner als die konfigurierte Schwelle sind, werden in die *zweite* Klasse eingestuft. Die Eingangswerte, die größer oder gleich als die konfigurierte Schwelle sind, werden in die *dritte* Klasse eingestuft.
sStringLevel1	STRING(255)	String, der in Klasse 1 einzuordnen ist
sStringLevel2	STRING(255)	String, der in Klasse 2 einzuordnen ist
sStringLevel3	STRING(255)	String, der in Klasse 3 einzuordnen ist

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.9 FB_ALY_TimeBasedEnvelope_1Ch

Der Algorithmus *Time Based Envelope 1Ch* vergleicht, ob die periodischen Eingangsdaten innerhalb einer konfigurierten Bandbreite von Werten liegen, die aus einer Datei gelesen werden. Dabei kann es sich beispielsweise um ein Referenzsignal handeln, welches zuvor mit dem Time Based Teach Path 1Ch erlernt wurde. Der Vergleich beginnt, wenn das Signal des Flags Start Period *TRUE* ist. Es wird empfohlen, *Time Based Envelope 1Ch* aufgrund des konkurrierenden Dateizugriffs nicht gleichzeitig mit Time Based Teach Path 1Ch zu verwenden. Stattdessen sollte zunächst ein Referenzsignal mit dem Time Based Teach Path 1Ch eingelernt werden und erst im Anschluss die Auswertung mithilfe des *Time Based Envelope 1Ch* erfolgen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_TimeBasedEnvelope_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bBusy: BOOL;
    eState: E_ALY_ReadState;
    bExecutingCompare: BOOL;
    nValueNumber: ULINT;
    fValueRead: LREAL;
    fBandLower: LREAL;
    fBandUpper: LREAL;
    bWithinBand: BOOL;
    eCompareResult: E_ALY_Classification_Bounds;
    nCurrentComparedCycles: ULINT;
    nCountWithinBand: ULINT;
    nCountSmaller: ULINT;
    nCountBigger: ULINT;
    stFileHeader: ST_ALY_FileHeader;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bBusy	BOOL	`TRUE`, wenn der FB aufgrund eines Dateizugriffs aktiv ist.
eState	E_ALY_ReadState [▶ 355]	Aktueller Status des FB. Notwendig aufgrund von asynchronem Dateizugriff.
bExecutingCompare	BOOL	`TRUE`, wenn der Algorithmus die Hüllkurve verarbeitet, ansonsten `FALSE`. Der Hüllkurvenprozess beginnt, wenn das Flag `bStartPeriod=TRUE` ist.
nValueNumber	ULINT	Wertzahl des Datenpunkts in der Datei, die gegenwärtig verglichen wird.
fValueRead	LREAL	Wert des aus einer Datei gelesenen Datenpunkts.
fBandLower	LREAL	Berechnete untere Grenze.
bWithinBand	BOOL	Berechnete obere Grenze.
eCompareResult	E_ALY_Classification_Bounds [▶ 352]	Klasse, zu der die Eingangswerte gehören (*WithinBounds/Smaller/Bigger*).
nCurrentComparedCycles	ULINT	Anzahl der verglichenen Zyklen.
nCountWithinBand	ULINT	Zählt, wie oft die Werte innerhalb des Bandes lagen.
nCountSmaller	ULINT	Zählt, wie oft die Werte kleiner als das Band waren.
nCountBigger	ULINT	Zählt, wie oft die Werte größer als das Band waren.
stFileHeader	ST_ALY_FileHeader	Header-Informationen der gelesenen Datei.

Methoden

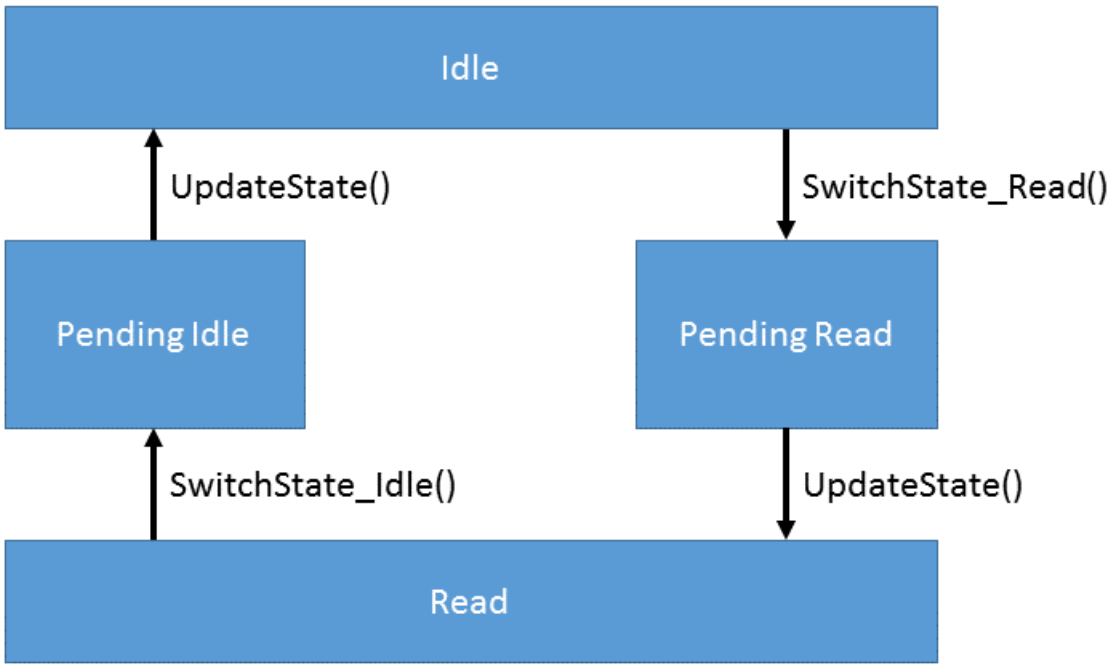
Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
GetBusyState()	Local	Diese Methode liefert den Zustand Busy des Funktionsbausteins.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SwitchState_Idle()	Local	Initiieren des Wechsels vom Zustand *Read* in den Zustand *Idle*. Siehe Zustandsdiagramm.
SwitchState_Read()	Local	Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Read*. Siehe Zustandsdiagramm.
UpdateState()	Local	Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Zustandsdiagramm

Aufgrund des asynchronen Dateizugriffs in Echtzeitanwendungen benötigt dieser Funktionsbaustein eine Zustandsmaschine, um den Dateizugriff vorzubereiten und abzuschließen.

Beim Start ist der Funktionsbaustein im Zustand *Idle*. Um die eingehenden Daten mit den Daten aus der Datei zu vergleichen, muss er in den Zustand *Read* wechseln. Daher muss die Methode *SwitchState_Read()* einmal aufgerufen werden, um den Funktionsbaustein in den Zustand *PendingRead* zu versetzen. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Read* befindet. In diesem Zustand können ein oder mehrere Vergleichszyklen durchlaufen werden. Wenn der Funktionsbaustein keine weiteren Zyklen vergleichen soll, kann er wieder in den Zustand *Idle* versetzt werden. Um den Zustandswechsel zu initiieren, muss die Methode *SwitchState_Idle()* aufgerufen werden. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Idle* befindet.

Zustandsdiagramm für den Lesevorgang der Daten:



Beispiel

```
VAR
    fbTimeBasedEnvelope : FB_ALY_TimeBasedEnvelope_1Ch;
    eBandMode : E_ALY_BandMode := E_ALY_BandMode.Absolute;
    fBand : LREAL := 2.0;
    nSegmentSize : UDINT := 200;
    tTimeout : TIME := T#5S;
    sFilePath : STRING := 'C:\TwinCAT\3.1\Boot\TimeBasedTeach.tas';
    bNegateStartPeriod : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    eState : E_ALY_ReadState := E_ALY_ReadState.Idle;
    bRead : BOOL;
    fInput : LREAL;
    bStartPeriod : BOOL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbTimeBasedEnvelope.Configure(eBandMode, fBand, nSegmentSize, tTimeout, sFilePath, bNegateStartP
eriod);
END_IF

// Call algorithm
eState := fbTimeBasedEnvelope.eState;
CASE eState OF
E_ALY_ReadState.Idle:
    IF bRead THEN
        fbTimeBasedEnvelope.SwitchState_Read();
        fbTimeBasedEnvelope.UpdateState();
    END_IF
E_ALY_ReadState.Read:
    fbTimeBasedEnvelope.SetChannelValue(fInput);
    fbTimeBasedEnvelope.Call(bStartPeriod:=bStartPeriod);

    IF NOT bRead THEN
        fbTimeBasedEnvelope.SwitchState_Idle();
        fbTimeBasedEnvelope.UpdateState();
    END_IF
E_ALY_ReadState.Pending,
E_ALY_ReadState.PendingIdle,
E_ALY_ReadState.PendingRead:
    fbTimeBasedEnvelope.UpdateState();
    eState := fbTimeBasedEnvelope.eState;
END_CASE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.9.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    bStartPeriod : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bStartPeriod	BOOL	Steigende Flanke startet Leseperiode. Der FB muss im Zustand Read sein.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.9.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.9.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.2.9.4 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eBandMode : E_ALY_BandMode;
    fBand : LREAL;
    nSegmentSize : UDINT;
    tTimeout : TIME;
    sFilePath : STRING(255);
    bNegateStartPeriod : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eBandMode	E_ALY_BandMode [▶ 353]	Konfigurieren des Bandmodus *Absolut*: Upper bound: Gelesener Wert + fBand Lower bound: Gelesener Wert – fBand *Relativ*: Upper bound: Gelesener Wert + (Gelesener Wert * fBand/100) Lower bound: Gelesener Wert - (Gelesener Wert * fBand/100)
fBand	LREAL	Konfigurieren des Bandes abhängig vom Bandmodus.
nSegmentSize	UDINT	Anzahl der gepufferten Elemente für Dateioperation.
tTimeout	TIME	Zeitüberschreitung für asynchrone Operationen.
sFilePath	STRING(255)	Pfad zur eingelernten Datei, z. B. *C:\TwinCAT\3.1\Boot\Teach.tas*
bNegateStartPeriod	BOOL	Negieren des Eingangsparameters bStartPeriod.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.9.5 SwitchState_Idle

Initiieren des Wechsels vom Zustand *Read* in den Zustand *Idle*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Idle : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Idle	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.2.9.6 SwitchState_Read

Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Read*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Read : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Read	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.2.9.7 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.2.9.8 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

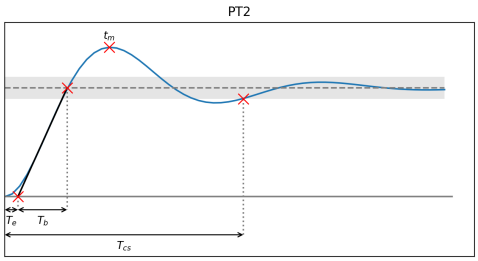
```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt TRUE zurück, wenn der FB aktiv ist.

5.1.1.2.10 FB_ALY_StepResponse_1Ch

Step Response 1Ch identifiziert Parameter der Sprungantwort einer PT2-Strecke. Dazu zählen die Verzugszeit T_e, die Ausgleichszeit T_b, die Ausregelzeit T_{cs} sowie der Zeitpunkt des Maximums t_m.



Um zu erkennen, ob die Strecke eingeschwungen ist, wird nach einem lokalen Minimum oder Maximum gesucht. Ist dieses innerhalb des Toleranzbandes (grau eingezeichnet), wird angenommen, dass die Strecke eingeschwungen ist. Erst dann wird die Ausregelzeit gesetzt.

Der Algorithmus startet, wenn ein neuer Sollwert außerhalb des zuvor gespeicherten Toleranzbandes liegt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StepResponse_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bExecuting: BOOL;
    fbEquivalentDeadTime: FB_ALY_Timespan;
    fbEquivalentTimeConstant: FB_ALY_Timespan;
    fbSettlingTime: FB_ALY_Timespan;
    fbTimeMax: FB_ALY_DateTime;
    fError: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bExecuting	BOOL	`TRUE`, wenn die Parameteridentifizierung aktiv ist.
fbEquivalentDeadTime	FB_ALY_Timespan	Verzugszeit der Strecke. Zeitspanne zwischen dem Start der Sprungantwort und dem Schnittpunt der Wendetangenten mit dem Startwert.
fbEquivalentTimeConstant	FB_ALY_Timespan	Ausgleichszeit der Strecke. Zeitspanne zwischen dem Schnittpunt der Wendetangenten mit dem Startwert und dem Schnittpunt der Wendetangenten mit dem Sollwert.
fbSettlingTime	FB_ALY_Timespan	Ausregelzeit der Strecke.
fbTimeMax	FB_ALY_DateTime	Zeitpunkt des maximalen Überschwingers.
fError	LREAL	Differenz zwischen Eingangswert und Sollwert.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbStepResponse : FB_ALY_StepResponse_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;

    fThresholdReversal : LREAL;
    bUseRelativeTolerance : BOOL;
    fErrorTolerance : LREAL;
    bConfigure : BOOL := TRUE;
```

```
    fInput : LREAL;
    fSetpoint : LREAL;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbStepResponse.Configure(fThresholdReversal, bUseRelativeTolerance, fErrorTolerance);
END_IF

// Call algorithm
fbStepResponse.SetChannelValue(1, fInput);
fbStepResponse.SetChannelValue(2, fSetpoint);
fbStepResponse.Call(fbSystemTime.tSystemTime)
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.2.10.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.10.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.2.10.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Data In 2: Setpoint
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

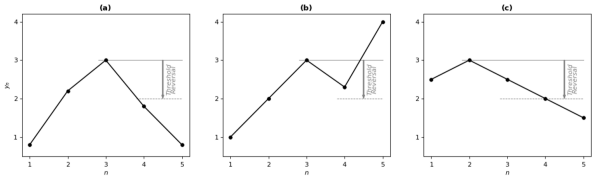
Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.2.10.4 Configure

Konfigurieren des Algorithmus.

Konfigurationsoptionen

- **Threshold Reversal:** Schwellenwert für die Identifizierung von Umkehrungen. Umkehrungen werden nur erkannt, wenn ihre Differenz zur nächsten Umkehrung den Wert von Threshold Reversal überschreitet.
  Nachfolgend sind drei Beispiele für die Identifizierung von Peaks mit dem Parameter *Threshold Reversal* aufgeführt.
  (a) Der Wert y_3 wird direkt nach der Verarbeitung des Werts y_4 als Peak identifiziert, da die Differenz zwischen y_3 und y_4 größer ist als *Threshold Reversal*.
  (b) Der Wert y_3 wird nicht als Peak identifiziert, da die Differenz zwischen y_3 und y_4 kleiner ist als *Threshold Reversal* und die Kurve nach y_4 wieder ansteigt.
  (c) Der Wert y_2 wird nach der Verarbeitung des Werts y_5 als Peak identifiziert, da die Differenz zwischen y_2 und y_5 *Threshold Reversal* überschreitet. Der Wert y_2 kann vorher nicht als Peak identifiziert werden, da die Differenz zwischen y_2 und y_3 (y_4) kleiner als/gleich *Threshold Reversal* ist und nicht bekannt ist, ob die Werte weiter sinken.



Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fThresholdReversal : LREAL;
    bUseRelativeTolerance : BOOL;
    fErrorTolerance : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fThresholdReversal	LREAL	Schwelle für die Peakanalyse
bUseRelativeTolerance	BOOL	Boolsches Flag. Ist dieses Flag *True*, so bezieht sich der Parameter *Error Tolerance* prozentual auf den Sollwert am Eingang. Andernfalls wird ein absolutes Toleranzband berücksichtigt.
fErrorTolerance	LREAL	Gibt die Größe des Toleranzbandes in Abhängigkeit zum Parameter *Relative Tolerance* an. Da Toleranzband wird bei einem Neustart der Parameteridentifizierung aktualisiert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.3 Clustering

5.1.1.3.1 FB_ALY_DenStream

DenStream ist eine Implementierung des gleichnamigen unüberwachten, dichtebasierten Clustering-Algorithmus [1]. Er basiert auf dem bekannten Clustering-Algorithmus DBSCAN [2, 3] und eignet sich insbesondere für Datenströme, deren Strukturen sich im Zeitverlauf verändern.

Die Anzahl der Eingangskanäle (im Folgenden als *n* bezeichnet) für diesen Algorithmus kann vom Benutzer gewählt werden. Diese Eingänge bilden den n-dimensionalen Merkmalsraum, in dem Cluster zu finden sind. In jedem Analysezyklus liefert der Datenstrom dem Algorithmus einen neuen Merkmalsvektor, der als Datenpunkt in diesem Merkmalsraum interpretiert werden kann. Cluster sind separierbare Bereiche mit einer hohen Dichte von Datenpunkten im Merkmalsraum.

In der ersten Phase des Algorithmus werden die eingehenden Datenpunkte sogenannten Mikro-Clustern (*MCs*) zugeordnet. Diese Mikro-Cluster haben Eigenschaften (wie Mittelpunkt, Gewicht und Varianz), die von den enthaltenen Datenpunkten abhängen. Nur Mikro-Cluster, deren Gewicht eine bestimmte Schwelle überschreitet, gelangen in die zweite Phase und werden vom DBSCAN-Algorithmus geclustert. Somit ist es nicht erforderlich, die Informationen über jeden einzelnen Datenpunkt zu behalten. Dadurch reduziert sich der Speicherbedarf, da es mit der Zeit viel weniger Mikro-Cluster als Datenpunkte gibt. Und auch der Rechenaufwand für den DBSCAN-Algorithmus ist viel geringer, da er über den reduzierten Satz von Mikro-Clustern und nicht über alle Datenpunkte läuft. Außerdem ist es möglich, auf die Gewichte der Mikro-Cluster eine Fading-Funktion anzuwenden. Auf diese Weise verlieren alte Datenpunkte mit der Zeit ihre Bedeutung für den Clustering-Prozess. Dadurch können Änderungen (wie die Verschiebung von Clustern oder ihr Verschwinden/Erscheinen mit der Zeit) vom Algorithmus erfasst werden.

Der DenStream-Algorithmus hat weitere Vorteile gegenüber anderen Clustering-Algorithmen. Der Benutzer muss die Anzahl der Mikro-Cluster nicht im Voraus kennen, da der DenStream-Algorithmus diese Anzahl automatisch bestimmt. Zudem ist der Algorithmus in der Lage, Ausreißer in den Daten zu erkennen, die zu keinem Cluster gehören. Da es sich um einen dichtebasierten Algorithmus handelt, ist es sogar möglich, separate Cluster beliebiger Form zu erkennen (auch wenn sie ineinander verschlungen sind).

Parametereinstellung

Hier geben wir eine kurze Einführung in die Funktionsweise des Algorithmus, hauptsächlich um dem Leser einen schnellen Einstieg in die Parametereinstellung zu ermöglichen. Für ein tiefgreifendes Verständnis des Algorithmus und seiner Parameter weisen wir den Leser auf die genannten Veröffentlichungen hin. Die meisten der hier verwendeten Begriffe und Parameternamen stammen direkt aus diesen Veröffentlichungen.

Die Parameter des DenStream-Algorithmus beeinflussen vor allem folgende Eigenschaften des Algorithmus:

- Die Grobheit der Mikro-Cluster,

- Die maximale Entfernung zwischen Datenpunkten/Mikro-Clustern, damit sie demselben Cluster zugeordnet werden,
- Die Mindestdichte, damit Datenpunkte als Cluster und nicht als Ausreißer erkannt werden,
- Die Fading-Rate, mit der ältere Datenpunkte ihre Bedeutung verlieren.

Die Parameter *Epsilon, Lambda* und *Mu x Beta* gehören zur ersten Phase des Algorithmus, der Bildung von Mikro-Clustern.

Nach Möglichkeit wird ein Datenpunkt in das Mikro-Cluster eingefügt, dessen Mittelpunkt dem Datenpunkt am nächsten liegt. Dazu werden die euklidischen Abstände zwischen dem Datenpunkt und den Mittelpunkten aller Mikro-Cluster verglichen und das Mikro-Cluster mit dem geringsten Abstand ausgewählt. Der Datenpunkt kann nur in das Mikro-Cluster eingefügt werden, wenn der Radius des Mikro-Clusters nach dem Einfügen nicht die Schwelle *Epsilon* überschreitet. Der Radius ist analog zur Varianz aller Datenpunkte, die im Mikro-Cluster enthalten sind. D.h. es können auch Datenpunkte in ein Mikro-Cluster integriert werden, deren euklidische Distanz zum Mittelpunkt des Clusters größer ist als *Epsilon*, solange genügend andere Punkte im Mikro-Cluster einen geringeren Abstand haben.

Wenn der Datenpunkt nicht in das nächste Mikro-Cluster eingefügt werden kann, wird mit diesem Datenpunkt ein neues Mikro-Cluster erstellt. Das Gewicht des jeweiligen Mikro-Clusters wird mit dem Einfügen eines Datenpunktes um eins erhöht.

Im linken Plot der Abbildung ist die Zuordnung der Datenpunkte zu den Mikro-Clustern exemplarisch für zwei Eingangskanäle skizziert. Abgebildet sind 20 Datenpunkte, die vier verschiedenen Mikro-Clustern zugeordnet werden. Das erste Mikro-Cluster enthält sechs Datenpunkte (#1, rot markiert), das zweite (#2, grün markiert) auch sechs, das dritte (#3, blau markiert) enthält sieben und das vierte (#1, grau markiert) nur einen Datenpunkte. Farbig hinterlegt ist jeweils der Bereich um den Mittelpunkt der Mikro-Cluster in dem sich ein neuer Datenpunkt befinden müsste, um bei gegebenem Epsilon (markiert durch gestrichelte Linie) in das jeweilige Mikro-Cluster aufgenommen zu werden. Dieser Einflussbereich ist größer, wenn bereits mehrere Datenpunkte im Mikro-Cluster enthalten sind und diese eine geringere Streuung aufweisen (vergleiche z.B. Mikro-Cluster #1 und #2). Außerdem können sich die Einflussbereiche von mehreren Mikro-Clustern überschneiden, und sich durch ihre Existenz gegenseitig beeinflussen, siehe Mikro-Cluster #2 (grün) und #3 (blau). Die Datenpunkte werden stets dem näheren Mikro-Cluster zugewiesen, weshalb die Einflussbereiche durch eine gerade Linie abgetrennt sind. Im Plot ist ein Datenpunkt zu sehen, der Mikro-Cluster #2 zugeordnet ist, wenn dieses jedoch nicht existieren würde, dann wäre er Mikro-Cluster #3 zugeordnet.

Wie auch in der Originalarbeit [1] werden Mikro-Cluster abhängig von ihrem Gewicht in potenzielle und Ausreißer-Mikro-Cluster aufgeteilt. Nur potenzielle Mikro-Cluster werden anschließend vom DBSCAN-Algorithmus geclustert. Die Datenpunkte in den Ausreißer-Mikro-Clustern werden als Ausreißer gekennzeichnet. Aber auch Ausreißer-Mikro-Cluster werden gespeichert und durch neue Datenpunkte aktualisiert, da sie sich noch zu potenziellen Mikro-Clustern entwickeln können. Das Gewicht eines Mikro-Clusters muss die Schwelle *Beta x Mu* überschreiten, um als potenzielles Mikro-Cluster gezählt zu werden. In der linken Skizze der Abbildung enthält das Mikro-Cluster #4 (grau) zum Beispiel nur einen Datenpunkt, hat damit ein Gewicht kleiner oder gleich eins und würde für *Beta x Mu* = 1 als Ausreißer-Mikro-Cluster gezählt werden.

Wenn eine Fading-Funktion angewendet wird, nimmt das Gewicht der Mikro-Cluster mit der Zeit ab. Diese Fading-Rate wird durch den Parameter *Lambda* festgelegt. Wenn der Wert auf null gesetzt wird, wird keine Fading-Funktion angewendet, anderenfalls nehmen die Gewichte jede Sekunde um einen Faktor von 2^(-*Lambda*) ab. Wenn das Gewicht eines Ausreißer-Mikro-Clusters unter eine interne Schwelle fällt (abhängig von *Mu x Beta* und *Lambda*), wird er aus dem Speicher gelöscht.

Die Parameter *Epsilon (DBSCAN)* und *Min Weight (DBSCAN)* betreffen die zweite Phase. Diese Parameter wurden vom DBSCAN-Algorithmus [3] übernommen.

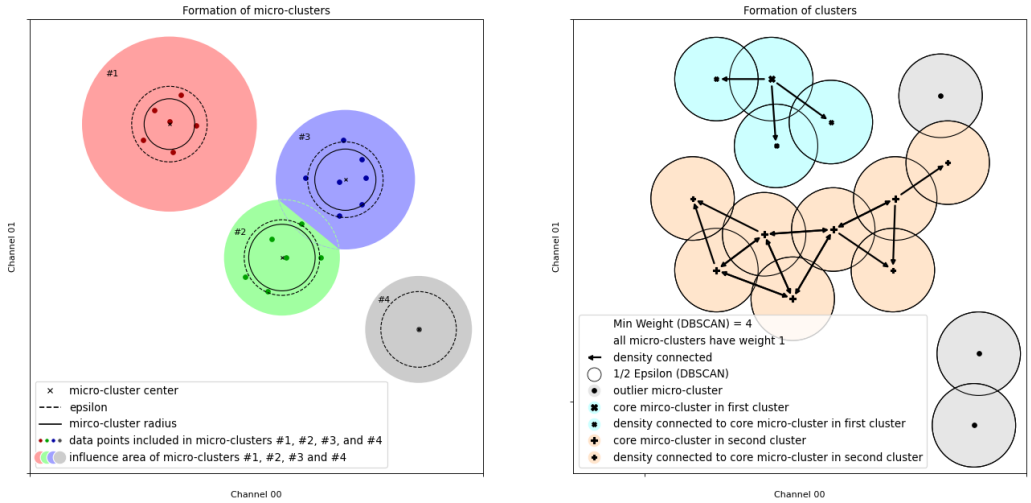
Der DBSCAN-Algorithmus läuft über den Satz potenzieller Mikro-Cluster und weist ihnen Cluster-Bezeichnungen zu. Dies kann entweder der Index des Clusters, zu dem sie gehören, oder die Bezeichnung Ausreißer sein. Dem gegenwärtig verarbeiteten Datenpunkt wird dann die Bezeichnung des Mikro-Clusters, zu dem er gehört, zugeordnet.

Wie clustert DBSCAN die Mikro-Cluster? Der Algorithmus arbeitet nach dem Konzept der Dichte-Erreichbarkeit. Objekte (in diesem Fall Mikro-Cluster) gehören zu demselben Cluster, wenn sie dichte-verbunden sind. Das bedeutet, dass es eine Kette von Mikro-Clustern mit einem maximalen Abstand *Epsilon (DBSCAN)* geben muss. Alle Mikro-Cluster, die diese Kette bilden, müssen eine zweite Bedingung erfüllen. Die Summe der Gewichte aller Mikro-Cluster innerhalb des Abstands *Epsilon (DBSCAN)* um jedes einzelne

Mikro-Cluster in dieser Kette muss die Schwelle *Min Weight (DBSCAN)* überschreiten. Mikro-Cluster, die nicht mit mindestens einem Mikro-Cluster dichte-verbunden sind, welches diese zweite Bedingung erfüllt, werden als Ausreißer gekennzeichnet.

Dies ist in der rechten Skizze der Abbildung dargestellt. Zur Vereinfachung wird hier angenommen, dass die Gewichtung aller Mikro-Cluster gleich 1 ist. Dies entspricht dem Fall, dass in jedem Mikro-Cluster genau ein Datenpunkt enthalten ist und keine Fading-Funktion angewendet wurde. Die beiden Cluster (markiert mit einem „x“ (türkis) und einem „+“ (orange)) mit den zwei Ausreißer-Mikro-Clustern ergeben sich, wenn der Parameter *Min Weight (DBSCAN)* auf vier gesetzt ist. Die mit einem großen „x“ bzw. „+“ markierten Mikro-Cluster sind Core-Mikro-Cluster. Das heißt mindestens drei weitere Mikro-Cluster (plus das betrachtete Mikro-Cluster = 4) haben eine maximale Distanz von *Epsilon (DBSCAN)* zu diesen Mikro-Clustern. Die mit einem kleinen „x“ bzw. „+“ markierten Mikro-Cluster sind keine Core-Mikro-Cluster, aber befinden sich in der *Epsilon (DBSCAN)* – Nachbarschaft eines Core-Mikro-Clusters und gehören somit zum selben Cluster. Das mit einem kleinen Punkt markierte Mikro-Cluster in der rechten, oberen Ecke ist ein Ausreißer-Mikro-Cluster. Es befindet sich zwar in der *Epsilon (DBSCAN)* – Nachbarschaft eines Mikro-Clusters, welches zu einem Cluster gezählt wird, dieses ist jedoch kein Core-Mikro-Cluster.

Genauso sind die beiden Mikro-Cluster rechts unten Ausreißer. Diese befinden sich zwar in unmittelbarer *Epsilon (DBSCAN)* – Nachbarschaft, jedoch sind sie nur zu zweit. Die Schwelle *Min Weight (DBSCAN)* der Gewichte wird nicht überschritten.



Die Parameter *outMCs Buffer Size* und *potMCs Buffer Size* sind spezifisch für diese Implementierung des Algorithmus und erforderlich, weil der Speicher für Ausreißer und potenzielle Mikro-Cluster vor der Ausführung zugewiesen werden muss. Somit begrenzen *outMCs Buffer Size* und *potMCs Buffer Size* die mögliche Anzahl der Ausreißer und potenziellen Mikro-Cluster während der Laufzeit. Der Benutzer muss solche Werte für diese Parameter finden, dass diese Grenze nicht überschritten wird.

Die maximale Anzahl der Ausreißer und potenziellen Mikro-Cluster während der Ausführung des Algorithmus hängt von der Verteilung der Eingangsdaten, aber auch von der Einstellung der anderen Parameter ab. Es gibt weniger Mikro-Cluster bei höheren Werten von *Epsilon*, da dies zu gröberen Mikro-Clustern führt, die Datenpunkte aus einem größeren Bereich enthalten können. Im Allgemeinen steigt die Anzahl der Ausreißer-Mikro-Cluster am Anfang der Analyse, sinkt jedoch wieder, wenn Ausreißer-Mikro-Cluster in potenzielle Mikro-Cluster übergehen. Wenn sich die Muster im Datenstrom mit der Zeit nicht verändern, pendelt sich die Anzahl der Mikro-Cluster nach einer Anfangsphase ein.

Je mehr Mikro-Cluster vorhanden sind, desto höher sind die Rechenanforderungen. Für alle Ausreißer und potenzielle Mikro-Cluster wir die Distanz zu einem Datenpunkt verglichen und anschließend müssen alle potenziellen Mikro-Cluster in die Berechnung des DBSCAN-Algorithmus einbezogen werden. Es muss also ein Kompromiss zwischen Rechengeschwindigkeit und der Grobheit der Mikro-Cluster eingegangen werden.

Was geschieht, wenn die Werte von *outMCs Buffer Size* und *potMCs Buffer Size* zu niedrig gesetzt sind und zu irgendeinem Zeitpunkt während der Analyse mehr Mikro-Cluster erforderlich sind, um die Eingangsdatenpunkte zu erfassen? Der Algorithmus ordnet in diesem Fall weiterhin die Datenpunkte den

vorhandenen Mikro-Clustern zu und kennzeichnet die Datenpunkte entsprechend, aber die vorhandenen Mikro-Cluster werden nicht mehr aktualisiert, um einen Überlauf des Puffers zu verhindern. Dies bedeutet, dass das Clustern der Datenpunkte fortgesetzt wird, aber mit einem insgesamt stagnierten Merkmalsraum (älterer Satz von Mikro-Clustern). Änderungen im Muster des Datenstroms könnten nicht mehr erkannt werden.

[1] F. Cao, M. Ester, W. Qian, A. Zhou. Density-Based Clustering over an Evolving Data Stream with Noise. In Proceedings of the 2006 SIAM International Conference on Data Mining, S. 326-337. SIAM.

[2] M. Ester, H.-P. Kriegel, J. Sander und X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of KDD*, 1996.

[3] J. Sander, M. Ester, H.-P. Kriegel, X. Xu. Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications. Data Mining and Knowlage Discovery 2, 169-194 (1998)

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DenStream
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nClusterIdx: DINT;
    nNumClusters: DINT;
    fbTimeLastEvent: FB_ALY_DateTime;
    fbTimeLastSwitch: FB_ALY_DateTime;
    bOverflow: BOOL;
    nNumPotMCs: UDINT;
    nNumOutMCs: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
nClusterIdx	DINT	Gibt den Cluster-Index an, den der DBSCAN-Algorithmus für den Datenpunkt des aktuellen Zyklus ausgibt.
nNumClusters	DINT	Gibt die Gesamtzahl der vom DBSCAN-Algorithmus erkannten Cluster an.
fbTimeLastEvent	FB_ALY_DateTime	Dies ist der Zeitstempel des letzten Zyklus mit einer Änderung des Cluster-Index
fbTimeLastSwitch	FB_ALY_DateTime	Dies ist der Zeitstempel des letzten Zyklus mit einem Wechsel zwischen Aktualisierung und Nichtaktualisierung von Mikro-Clustern (entweder durch Setzen des Eingangs `bUpdateMicroCluster` auf `TRUE` oder durch internes Verhindern eines Überlaufs von `nPotMCsBufferSize` oder `nOutMCBufferSize`)
bOverflow	BOOL	`TRUE`, wenn die Aktualisierung der Mikro-Cluster gestoppt wird, um ein Überlaufen von `nPotMCsBufferSize` oder `nOutMCsBufferSize` zu verhindern.
nNumPotMCs	UDINT	Gibt die Anzahl der gegenwärtig vorhandenen potenziellen Mikro-Cluster an.
nNumOutMCs	UDINT	Gibt die Anzahl der gegenwärtig vorhandenen Ausreißer-Mikro-Cluster an.

Beispiel

```
VAR
    fbDenStream : FB_ALY_DenStream(nNumChannels := 2);
    fbSystemTime : FB_ALY_GetSystemTime;

    fEps : LREAL := 0.15;
    fMuBeta : LREAL := 2;
    fLambda : LREAL := 0;
    fEps_DBSCAN : LREAL := 0.8;
    fMinWeight_DBSCAN : LREAL := 2;
    nPotMCsBufferSize : UDINT := 100;
    nOutMCsBufferSize : UDINT := 100;
    bConfigure : BOOL := TRUE;

    nInputCh1 : UDINT;
    fInputCh2 : LREAL;
    bUpdateMCs : BOOL := TRUE;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbDenStream.Configure(
        fEps := fEps,
        fMuBeta := fMuBeta,
        fLambda := fLambda,
        fEpsDBSCAN := fEps_DBSCAN,
        fMinWeightDBSCAN:= fMinWeight_DBSCAN,
        nPotMCsBufferSize := nPotMCsBufferSize,
        nOutMCsBufferSize := nOutMCsBufferSize);
END_IF
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbDenStream.SetChannelValue(1, nInputCh1);
fbDenStream.SetChannelValue(2, fInputCh2);
fbDenStream.Call(tTimestamp := fbSystemTime.tSystemTime, bUpdateMCs := bUpdateMCs);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

5.1.1.3.1.1 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.3.1.2 FB_init

Initialisieren der Anzahl der Eingangskanäle, der Anzahl der Cluster-Mittelpunkte sowie die Größe des Aggregationspuffers.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
    nNumClusters : UDInT := 1;
    nAggBufferSize : UDINT := 10;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.
nNumClusters	UDINT	Definiert die Anzahl der Cluster.
nAggBufferSize	UDINT	Gibt die Anzahl an Zyklen an, nach denen die Cluster-Mittelpunkte aktualisiert werden. Die Eingangswerte über diese Anzahl von Zyklen werden intern (im Aggregation Buffer) zwischengespeichert.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.3.1.3 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    bUpdateMCs : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.
bUpdateMCs	BOOL	TRUE: Die Mikro-Cluster werden durch die eingehenden Daten aktualisiert. FALSE: Die vorhandenen Mikro-Cluster bleiben unverändert und werden nur verwendet, um den Cluster-Index der eingehenden Datenpunkte zu bestimmen.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.3.1.4 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fEps : LREAL;
    fMuBeta : LREAL;
    fLambda : LREAL;
    fEpsDBSCAN : LREAL;
    fMinWeightDBSCAN : LREAL;
    nPotMCsBufferSize : UDINT;
    nOutMCsBufferSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nEps	LREAL	Schwelle für den maximalen Radius von Mikro-Clustern.
fMuBeta	LREAL	Schwelle für das Gewicht eines Mikro-Clusters zwischen Ausreißer und potenziellem Mikro-Cluster.
fLambda	LREAL	Gibt die Vergessensrate des Algorithmus an. Das Gewicht jedes Datenpunkts nimmt jede Sekunde um einen Faktor von 2^(-f*Lambda*) ab.
fEpsDBSCAN	LREAL	Gibt den Parameter Epsilon des DBSCAN-Algorithmus an.
fMinWeightDBSCAN	LREAL	Schwelle für die Summe der Gewichte in der Epsilon-Nachbarschaft eines Mikro-Clusters für den DBSCAN-Algorithmus.
nPotMCsBufferSize	UDINT	Die maximale Anzahl potenzieller Mikro-Cluster. Der Speicher wird *potMCs Buffer Size* Mikro-Clustern zugeordnet.
nOutMCsBufferSize	UDINT	Die maximale Anzahl von Ausreißer-Mikro-Clustern. Der Speicher wird *outMCs Buffer Size* Mikro-Clustern zugeordnet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.3.1.5 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.3.2 FB_ALY_SequentialKMeans

Der Algorithmus *Sequential k-Means* ist eine Implementierung des gleichnamigen unüberwachten Clustering-Algorithmus und stellt eine sequentielle Variante des weit verbreiteten Clustering-Algorithmus k-Means für Streamingdaten dar. Das Ziel des Algorithmus ist es, anhand der Struktur der Daten Cluster zu finden, die jeweils ähnliche Datenpunkte beinhalten und unterschiedliche Datenpunkte voneinander abgrenzen.

Die Anzahl der Eingangskanäle (im Folgenden als *n* bezeichnet) für diesen Algorithmus kann vom Benutzer frei gewählt werden. Diese Eingänge spannen den n-dimensionalen Merkmalsraum auf, in dem die Cluster zu finden sind. In jedem Analysezyklus liefert der Datenstrom dem Algorithmus einen neuen

Merkmalsvektor, der als Datenpunkt in diesem Merkmalsraum interpretiert werden kann. Datenpunkte, die in diesem Merkmalsraum nahe beieinanderliegen, werden dem gleichen Cluster zugewiesen. Die Anzahl der vorhandenen Cluster muss vor Beginn der Analyse vom Anwender gesetzt werden und bleibt fest.

Im Gegensatz zum k-Means Algorithmus für die klassische Batch-Analyse liegen die Daten für den *Sequential k-Means* zum Zeitpunkt der Analyse nicht vollständig vor. Stattdessen gehen die Datenpunkte in Form von Streamingdaten nach und nach ein und werden hier daher sequentiell verarbeitet und dem entsprechenden am nächsten liegenden Cluster zugeordnet. Aus diesem Vorgehen ergeben sich einige Unterschiede, wobei zwei davon für die Benutzung des Algorithmus sowie die Parametereinstellungen besonders relevant sind.

Zum einen liegen zu Beginn einer Batch-Analyse bereits alle Datenpunkte und damit auch die Wertebereiche der einzelnen Merkmale vor, bei der sequentiellen Analyse ist dies nicht der Fall, sodass die Wertebereiche vorher nicht zwangsläufig feststehen. Es ist jedoch hilfreich, die Wertebereiche der Eingangskanäle bereits vorher zu kennen, auch wenn die Werte erst im Laufe der Analyse eintreffen. Dies spielt insbesondere für die Initialisierung der Cluster-Mittelpunkte eine Rolle. Für die Initialisierung können drei verschiedene Vorgehensweisen gewählt werden. Die Mittelpunkte können mit konkreten Werten über ein Parameter-Array vorgegeben werden *(Values)*. Die Mittelpunkte können aber auch zufällig *(Random)* oder äquidistant *(Equidistant)* in einem definierten Wertebereich gesetzt werden. Für die Initialisierungsmodi *Random* und *Equidistant* werden die Wertebereiche benötigt und müssen über die Parameter *Lower Bounds* und *Upper Bounds* für die einzelnen Eingangskanäle eingestellt werden.

Zum anderen werden in einer Batch-Analyse üblicherweise alle Datenpunkte mehrfach durchlaufen, um die Cluster-Mittelpunkte zu aktualisieren, bis diese sich nur noch minimal verändern. Das ist in der Form im Rahmen der sequentiellen Analyse nicht möglich. Um die Cluster-Mittelpunkte jedoch trotzdem anpassen zu können und Datenpunkte mehrfach zu durchlaufen, verfügt der Algorithmus *Sequential k-Means* über einen Zwischenspeichermechanismus, den *Aggregation Buffer*. Mit diesem wird es möglich, eine begrenzte Anzahl von Werten zwischenzuspeichern. Beim Füllen des Buffers, werden alle eintreffenden Datenpunkte dem am nächsten liegenden Cluster zugeordnet. Die Distanz zwischen einem Datenpunkt und den Cluster-Mittelpunkten wird durch die Euklidische Norm bestimmt. Erst wenn der Buffer gefüllt ist, werden die Cluster-Mittelpunkte anhand der neu zugewiesenen Datenpunkte im Buffer aktualisiert. Dabei entspricht der neue Cluster-Mittelpunkt dem Mittelwert aller im Cluster enthaltener Datenpunkte. Dieser kann inkrementell berechnet werden, sodass die alten Datenpunkte nicht für die Berechnung benötigt werden. Die Größe des Buffers wird durch den Parameter *Aggregation Buffer Size* gesetzt, der Default-Wert ist 10. Durch den Parameter *Max Iterations* kann zusätzlich angegeben werden, wie oft durch den Buffer iteriert wird. Hier ist der Default-Wert gleich eins. Wird der Wert beispielsweise auf zwei gesetzt, werden nach dem ersten Anpassen der Cluster-Mittelpunkte, die Datenpunkte im Buffer den Clustern neu zugewiesen und anschließend die Cluster-Mittelpunkte erneut angepasst. Aufgrund der Verschiebung der Cluster-Mittelpunkte ist es möglich, dass einzelne Datenpunkte von einer Iteration zur nächsten unterschiedlichen Clustern zugewiesen werden können. Aufgrund der begrenzten Rechenkapazität für die Datenverarbeitung zwischen zwei Zyklen sollten für die Parameter *Aggregation Buffer Size* und *Max Iterations* keine zu hohen Werte vergeben werden, da die Aktualisierung der Cluster-Mittelpunkte andernfalls eventuell nicht gewährleistet werden kann. Werden die Cluster-Mittelpunkte bei großen Werten für diese Parameter nicht aktualisiert, bei kleineren Parameterwerten hingegen schon, ist dies ein Indiz dafür, dass die Rechenkapazität für die eingestellten Parameterwerte nicht ausreicht und kleinere Werte gewählt werden sollten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_SequentialKMeans
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nClusterIdx: DINT;
    fDistance: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
nClusterIdx	DINT	Gibt den Cluster-Index an, den der DBSCAN-Algorithmus für den Datenpunkt des aktuellen Zyklus ausgibt.
fDistance	LREAL	Gibt die Gesamtzahl der vom DBSCAN-Algorithmus erkannten Cluster an.

Beispiel

```
VAR
    fbSequentialKMeans : FB_ALY_SequentialKMeans(nNumChannels := 2, nNumClusters := 3, nAggBufferSiz
e := 10);

    nMaxIterations : UDINT :=1;
    eInitMode : E_ALY_KMeansInitMode := E_ALY_KMeansInitMode.Values;
    aInitialClusterCenters : ARRAY[1..3] OF ARRAY[1..2] OF LREAL := [[-30, 0], [10, 2], [30, 4]];

    bConfigure : BOOL := TRUE;

    nInputCh1 : UDINT;
    fInputCh2 : LREAL;
    bUpdateClusterCenters : BOOL := TRUE;

    aClusterCenters : ARRAY[1..3] OF ARRAY[1..2] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbSequentialKMeans.Configure(nMaxIterations := nMaxIterations, eInitMode := eInitMode);
    fbSequentialKMeans.SetInitialClusterCenters(ADR(aInitialClusterCenters), SIZEOF(aInitialClusterC
enters));
END_IF

// Call algorithm
fbSequentialKMeans.SetChannelValue(1, nInputCh1);
fbSequentialKMeans.SetChannelValue(2, fInputCh2);
fbSequentialKMeans.Call(bUpdateClusterCenters, ADR(aClusterCenters), SIZEOF(aClusterCenters));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

Methoden

Name	Definitions-ort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
GetResults()	Local	Abrufen der Ergebnis-Matrix, ohne neue Werte hinzuzufügen
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetInitialBounds()	Local	Setzen der initialen Grenzen. Abhängig des konfigurierten Initialisierungsmodus.
SetInitialClusterCenters()	Local	Setzen der initialen Cluster-Mittelpunkte. Abhängig des konfigurierten Initialisierungsmodus.

5.1.1.3.2.1 FB_init

Initialisieren der Anzahl der Eingangskanäle, der Anzahl der Cluster-Mittelpunkte sowie die Größe des Aggregationspuffers.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
    nNumClusters : UDInT := 1;
    nAggBufferSize : UDINT := 10;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.
nNumClusters	UDINT	Definiert die Anzahl der Cluster.
nAggBufferSize	UDINT	Gibt die Anzahl an Zyklen an, nach denen die Cluster-Mittelpunkte aktualisiert werden. Die Eingangswerte über diese Anzahl von Zyklen werden intern (im Aggregation Buffer) zwischengespeichert.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.3.2.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    bUpdateMCs : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.
bUpdateMCs	BOOL	TRUE: Die Mikro-Cluster werden durch die eingehenden Daten aktualisiert. FALSE: Die vorhandenen Mikro-Cluster bleiben unverändert und werden nur verwendet, um den Cluster-Index der eingehenden Datenpunkte zu bestimmen.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.3.2.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nMaxIterations : UDINT;
    eInitMode : E_ALYKMeansInitMode;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nMaxIterations	UDINT	Gibt an, wie oft über die Werte im Aggregation Buffer iteriert wird. Default ist 1.
eInitMode	E_ALY_KMeansInitMode [▶ 357]	Gibt an, mit welcher Methode die Initial-Werte für die Cluster-Mittelpunkte gesetzt werden: *Random*: Die Cluster-Mittelpunkte werden zufällig gesetzt in den Grenzen, die durch Lower Bounds und Upper Bounds gesetzt sind. Die Grenzen können nach der Konfiguration mit der Methode SetInitialBounds [▶ 156] gesetzt werden. *Euqidistant*: Die Cluster-Mittelpunkte werden äquidistant in dem Wertebereich verteilt, welcher durch die Grenzen Lower Bounds und Upper Bounds definiert ist. Die Grenzen können nach der Konfiguration mit der Methode SetInitialBounds [▶ 156] gesetzt werden. *Values*: Die Cluster-Mittelpunkte werden mit den Werten initialisiert, die durch initialen Cluster-Mittelpunkte gesetzt werden. Die Cluster-Mittelpunkte können nach der Konfiguration mit der Methode SetInitialClusterCenters [▶ 157] gesetzt werden.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.3.2.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.3.2.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.3.2.6 SetInitialBounds

Setzen der initialen Grenzen. Abhängig des konfigurierten Initialisierungsmodus.

Syntax

Definition:

```
METHOD SetInitialBounds : BOOL
VAR_INPUT
    pLowerBoundsConfigArray : POINTER TO LREAL;
    nLowerBoundsConfigArraySize : UDINT;
    pUpperBoundsConfigArray : POINTER TO LREAL;
    nUpperBoundsConfigArraySize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pLowerBoundsConfigArray	POINTER TO LREAL	Zeiger auf ein Array mit den unteren Werten der zu initialisierenden Grenzen. Dim.: nNumChannels
nLowerBoundsConfigArraySize	UDINT	Größe des Arrays mit den unteren Werten der zu initialisierenden Grenzen.
pUpperBoundsConfigArray	POINTER TO LREAL	Zeiger auf ein Array mit den oberen Werten der zu initialisierenden Grenzen. Dim.: nNumChannels
nUpperBoundsConfigArraySize	UDINT	Größe des Arrays mit den oberen Werten der zu initialisierenden Grenzen.

Rückgabewert

Name	Typ	Beschreibung
SetInitialBounds	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.3.2.7 SetInitialClusterCenters

Setzen der initialen Cluster-Mittelpunkte. Abhängig des konfigurierten Initialisierungsmodus.

Syntax

Definition:

```
METHOD SetInitialClusterCenters : BOOL
VAR_INPUT
    pInitialClusterCenterMatrix : POINTER TO LREAL;
    nInitialClusterCenterMatrixSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInitialClusterCenterMatrix	POINTER TO LREAL	Zeiger auf eine Matrix mit initialen Cluster-Mittelpunkten. Dim.: nNumChannels*nNumClusters
nInitialClusterCenterMatrixSize	UDINT	Größe der Matrix mit initialen Cluster-Mittelpunkten.

Rückgabewert

Name	Typ	Beschreibung
SetInitialClusterCenters	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.3.2.8 GetResults

Abrufen der Ergebnis-Matrix, ohne neue Werte hinzuzufügen.

Syntax

Definition:

```
METHOD GetResults : BOOL
VAR_INPUT
    pClusterCenterMatrixOut : POINTER TO LREAL;
    nClusterCenterMatrixOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pClusterCenterMatrixOut	POINTER TO LREAL	Zeiger auf Cluster-Mittelpunkt-Matrix. Dim.: nNumChannels*nNumClusters
nClusterCenterMatrixOut	UDINT	Größe der Cluster-Mittelpunkt-Matrix

Rückgabewert

Name	Typ	Beschreibung
GetResults	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4 Compare

5.1.1.4.1 FB_ALY_Demultiplexer

Der Demultiplexer wählt einen Ausgangskanal anhand des Eingangswertes aus. Dafür wird der Eingangswert als Integer interpretiert. Dieser Wert entspricht dem Ausgangskanal. Ist der Wert Außerhalb der konfigurierten Anzahl an Kanälen, so wird der Ausgangskanal 0 gesetzt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Demultiplexer
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSwitched: BOOL;
    nCurrentChannel: UDINT;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bSwitched	BOOL	TRUE, wenn ein Kanalwechsel erfolgte.
nCurrentChannel	UDINT	Gibt die Nummer des ausgewählten Kanals an. Der Wert ist 0, wenn der ausgewählte Kanal außerhalb der konfigurierten Kanäle ist.
nCount	ULINT	Beginnt mit 1 für den Kanal, der bei Beginn der Analyse ausgewählt wird, und zählt jedes Mal hoch, wenn ein anderer Kanal ausgewählt wird.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Kanalwechsels.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
GetResults()	Local	Abrufen des Ergebnis-Arrays, ohne neue Werte hinzuzufügen
FB_init()	Local	Initialisieren der Anzahl der Ausgangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe kanalspezifischer Werte an den Algorithmus.

Beispiel

```
VAR
    fbDemultiplexer : FB_ALY_Demultiplexer(nNumChannels := 3);
    fbSystemTime : FB_ALY_GetSystemTime;
    nInput : INT := 1;
    aResults : ARRAY[0..3] OF BOOL;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbDemultiplexer.SetChannelValue(nInput);
fbDemultiplexer.Call(fbSystemTime.tSystemTime, ADR(aResults), SIZEOF(aResults));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.1.1 FB_init

Initialisieren der Anzahl der Ausgangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.4.1.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    pDataOut : POINTER TO BOOL;
    nDataOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.
pDataOut	POINTER TO BOOL	Zeiger auf das Ergebnis-Array der Ausgangskanäle. Dim.: nNumChannels+1
nDataOutSize	UDINT	Größe des Ergebnis-Arrays.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.1.5 GetResults

Abrufen eines Ergebnis-Arrays, ohne neue Werte hinzuzufügen.

Syntax

Definition:

```
METHOD GetResults : BOOL
VAR_INPUT
    pDataOut : POINTER TO BOOL;
    nDataOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pDataOut	POINTER TO BOOL	Zeiger auf das Ergebnis-Array der Ausgangskanäle. Dim.: nNumChannels+1
nDataOutSize	UDINT	Größe des Ergebnis-Arrays.

Rückgabewert

Name	Typ	Beschreibung
GetResults	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.2 FB_ALY_DetectStringChange_1Ch

Der *Detect String Change 1Ch* erkennt und zählt Änderungen der Stringwerte. Dabei kann die Groß- und Kleinschreibung beachtet werden oder nicht.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DetectStringChange_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bStringChanged: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bStringChanged	BOOL	`TRUE`, wenn eine Änderung des Eingangsstrings erkannt wurde, ansonsten `FALSE`.
nCount	ULINT	Zählt jedes Mal hoch, `bStringChanged=TRUE` ist.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel der zuletzt erkannten Stringänderung.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbDetectStringChange : FB_ALY_DetectStringChange_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    bCaseSensitive : BOOL := TRUE;
    bConfigure : BOOL := TRUE;
    sInput : STRING := 'Modify';
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbDetectStringChange.Configure(bCaseSensitive);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbDetectStringChange.SetChannelValue(sInput);
fbDetectStringChange.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.2.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.2.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bCaseSensitive : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bCaseSensitive	BOOL	Wenn TRUE, wird die Groß- und Kleinschreibung beachtet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.2.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.3 FB_ALY_DynamicTimeWarping

Der Algorithmus *Dynamic Time Warping* vergleicht Eingangsdaten mit zuvor aufgenommenen Templates. Das Besondere an dem Algorithmus ist, dass auch Signale mit unterschiedlicher Geschwindigkeit oder aber auch verschobene Signale verglichen werden können. Als Ergebnis wird die Distanz zwischen dem Eingangssignal und dem jeweiligen Template ausgegeben. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Ist die Distanz 0, so sind beide Signale identisch. Die Höhe der Distanz ist abhängig von der Gleichheit aber auch von der Länge der Signale.

Der Vergleich beginnt, wenn das Signal des Flags Start Period *TRUE* ist. Ein Ergebnis wird ausgegeben, wenn das Signal des Flags Stop Period TRUE oder das Flag Start Period erneut TRUE ist.

Es wird empfohlen, *Dynamic time Warping* aufgrund des konkurrierenden Dateizugriffs nicht gleichzeitig mit Time Based Teach Path 1Ch zu verwenden. Stattdessen sollte zunächst ein Referenzsignal mit dem Time Based Teach Path 1Ch eingelernt werden und erst im Anschluss die Auswertung mithilfe des *Dynamic time Warping* erfolgen. Die Templates enthalten Referenzsignale, die zuvor mit dem Time Based Teach Path 1Ch aufgenommen wurden. In der Regel handelt es sich bei den Templates um wenige Hundert Stützpunkte. Daher ist eine Reduktion der Daten mit dem Baustein Downsampling 1Ch oft sinnvoll.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DynamicTimeWarping
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bBusy: BOOL;
    eState: E_ALY_ReadState;
    bExecutingCompare: BOOL;
    nBestMatchIdx: ULINT;
    fValueRead: LREAL;
    stFileHeader: ST_ALY_FileHeader;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bBusy	BOOL	`TRUE`, wenn der FB aufgrund eines Dateizugriffs aktiv ist.
eState	E_ALY_ReadState [▶ 355]	Aktueller Status des FB aufgrund von asynchronen Dateizugriffen.
bExecutingCompare	BOOL	`TRUE`, wenn der Algorithmus die Hüllkurve verarbeitet, ansonsten `FALSE`. Der Hüllkurvenprozess beginnt, wenn das Flag `bStartPeriod=TRUE` ist.
nBestMatchIdx	UDINT	Gibt den Index des Templates mit der geringsten Distanz zum Eingangskanal aus.
stFileHeader	ST_ALY_FileHeader	Header-Informationen der zuletzt gelesenen Datei.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Konfiguriert die Dateipfade der Templates.
FB_init()	Local	Initialisieren der Anzahl der Templates.
GetBusyState()	Local	Diese Methode liefert den Zustand Busy des Funktionsbausteins.
GetChannelOutputVaue()	Local	Methode für das Abholen von einzelnen Ausgangswerten aus dem Ausgangs-Array
GetChannelOutputArray()	Local	Methode für das Abholen des gesamten Ausgangs-Arrays.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SwitchState_Idle()	Local	Initiieren des Wechsels vom Zustand *Read* in den Zustand *Idle*. Siehe Zustandsdiagramm.
SwitchState_Read()	Local	Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Read*. Siehe Zustandsdiagramm.
UpdateState()	Local	Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

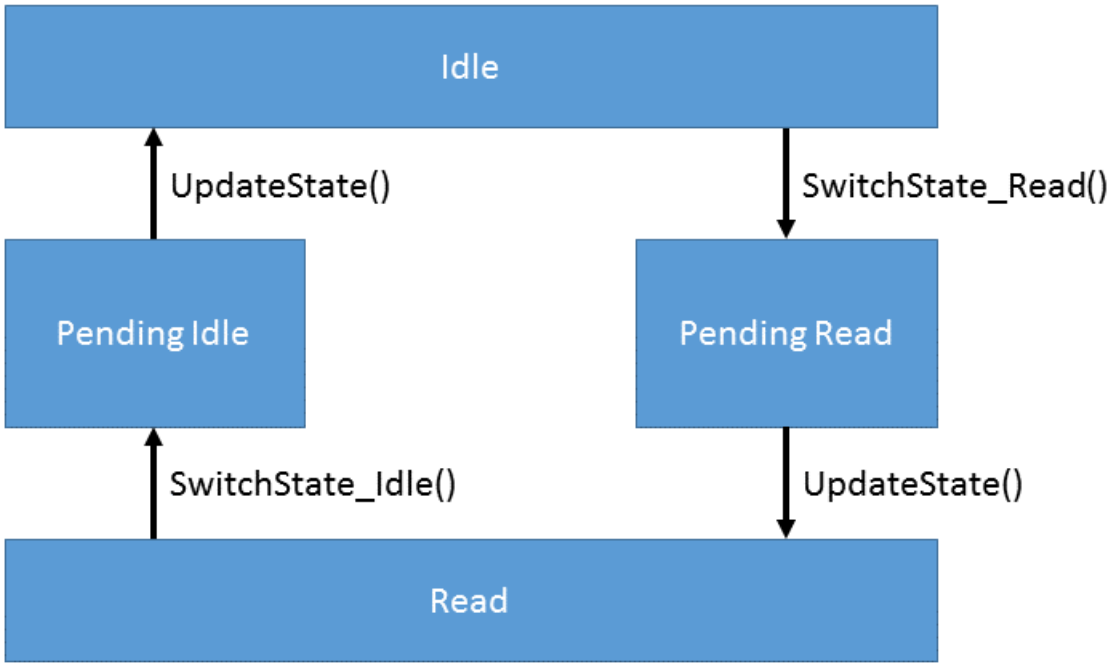
Zustandsdiagramm

Aufgrund des asynchronen Dateizugriffs in Echtzeitanwendungen benötigt dieser Funktionsbaustein eine Zustandsmaschine, um den Dateizugriff vorzubereiten und abzuschließen.

Beim Start ist der Funktionsbaustein im Zustand *Idle*. Um die eingehenden Daten mit den Daten aus der Datei zu vergleichen, muss er in den Zustand *Read* wechseln. Daher muss die Methode *SwitchState_Read()* einmal aufgerufen werden, um den Funktionsbaustein in den Zustand *PendingRead* zu versetzen.

Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Read* befindet. In diesem Zustand können ein oder mehrere Vergleichszyklen durchlaufen werden. Wenn der Funktionsbaustein keine weiteren Zyklen vergleichen soll, kann er wieder in den Zustand *Idle* versetzt werden. Um den Zustandswechsel zu initiieren, muss die Methode *SwitchState_Idle()* aufgerufen werden. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Idle* befindet.

Zustandsdiagramm für den Lesevorgang der Daten:



Beispiel

```
VAR
    fbDynamicTimeWarping : FB_ALY_DynamicTimeWarping(nNumChannels := 3);
    tTimeout : TIME := T#5S;
    sFilePath1 : STRING := 'C:\TwinCAT\3.1\Boot\Template1.tas';
    sFilePath2 : STRING := 'C:\TwinCAT\3.1\Boot\Template2.tas';
    sFilePath3 : STRING := 'C:\TwinCAT\3.1\Boot\Template3.tas';

    bConfigure : BOOL := TRUE;
    eState : E_ALY_ReadState := E_ALY_ReadState.Idle;
    bRead : BOOL;
    fInput : LREAL;
    bStartPeriod : BOOL;
    bStopPeriod : BOOL;

    aDistances : ARRAY[1..3] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbDynamicTimeWarping.ConfigureChannel(1, sFilePath1);
    fbDynamicTimeWarping.ConfigureChannel(2, sFilePath2);
    fbDynamicTimeWarping.ConfigureChannel(3, sFilePath3);
    fbDynamicTimeWarping.Configure(tTimeout);
END_IF

// Call algorithm
eState := fbDynamicTimeWarping.eState;
CASE eState OF
E_ALY_ReadState.Idle:
    IF bRead THEN
        fbDynamicTimeWarping.SwitchState_Read();
        fbDynamicTimeWarping.UpdateState();
    END_IF
E_ALY_ReadState.Read:
    fbDynamicTimeWarping.SetChannelValue(fInput);
    fbDynamicTimeWarping.Call(bStartPeriod:=bStartPeriod, bStopPeriod:=bStopPeriod);
```

```
    IF NOT bRead THEN
        fbDynamicTimeWarping.SwitchState_Idle();
        fbDynamicTimeWarping.UpdateState();
    END_IF
E_ALY_ReadState.Pending,
E_ALY_ReadState.PendingIdle,
E_ALY_ReadState.PendingRead:
    fbDynamicTimeWarping.UpdateState();
    eState := fbDynamicTimeWarping.eState;
END_CASE

// Get results
fbDynamicTimeWarping.GetOutputArray(pArrayOut:=ADR(aDistances), nArrayOutSize:=SIZEOF(aDistances));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.3.1 FB_init

Initialisieren der Anzahl der Vergleichskanäle (Templates).

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Anzahl der Vergleichskanäle (Templates)

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.4.3.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    bStartPeriod : BOOL;
    bStopPeriod : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bStartPeriod	BOOL	Steigende Flanke startet die Berechnung. Der FB muss im Zustand Read sein.
bStopPeriod	BOOL	Steigende Flanke stoppt die Berechnung. Die Ergebnisse werden ausgegeben. Der FB muss im Zustand Read sein.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.3.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    tTimeout : TIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimeout	TIME	Zeitüberschreitung für asynchrone Operationen.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.3.4 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    sFilePath : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
sFilePath	STRING(255)	Pfad zur eingelernten Datei, z. B. *C:\TwinCAT\3.1\Boot\Teach.tas*

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.3.5 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.3.6 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt TRUE zurück, wenn der FB aktiv ist.

5.1.1.4.3.7 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert entspricht der Distanz zwischen dem Eingangssignal und dem jeweiligen Template. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Der Wert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.3.8 GetOutputArray

Abholen des gesamten Ausgangs-Arrays. Die Array-Elemente entsprechen der Distanz zwischen dem Eingangssignal und dem jeweiligen Template. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Die Anzahl an Elementen entspricht der konfigurierten Anzahl an Vergleichskanälen (Templates). Die Werte werden nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetOutputArray : BOOL
VAR_INPUT
    pArrayOut : POITNER TO LREAL;
    nArrayOutSize : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pArrayOut	POINTER TO LREAL	Zeiger auf das Ausgangs-Array: Dim: nBufferSize
nArrayOutSize	UDINT	Größe des Ausgangs-Arrays

Rückgabewert

Name	Typ	Beschreibung
GetOutputArray	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.3.9 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.3.10 SwitchState_Idle

Initiieren des Wechsels vom Zustand *Read* in den Zustand *Idle*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Idle : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Idle	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.4.3.11 SwitchState_Read

Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Read*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Read : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwithcState_Read	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.4.3.12 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.4.4 FB_ALY_DynamicTimeWarpingInterval

Der Algorithmus *Dynamic Time Warping Interval* vergleicht mehrere Eingangsdaten miteinander. Das Besondere an dem Algorithmus ist, dass auch Signale mit unterschiedlicher Geschwindigkeit oder aber auch verschobene Signale verglichen werden können. Für den Vergleich wird nur das Signalintervall eines konfigurierten Fensters berücksichtigt. Neue Ergebnisse werden nach dem Ablauf des Fensters ausgegeben. Als Ergebnis wird die Distanz zwischen dem Referenzsignal und dem jeweiligen Eingangssignal ausgegeben. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Ist die Distanz 0, so sind beide Signale identisch. Die Höhe der Distanz ist abhängig von der Gleichheit aber auch von der Länge der Signale.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DynamicTimeWarpingInterval
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
```

```
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    nBestMatchIdx: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
nBestMatchIdx	UDINT	Gibt den Index des Eingangskanals mit der geringsten Distanz zum Referenzkanal aus.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Konfiguration des Algorithmus.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
GetChannelOutputVaue()	Local	Methode für das Abholen von einzelnen Ausgangswerten aus dem Ausgangs-Array
GetChannelOutputArray()	Local	Methode für das Abholen des gesamten Ausgangs-Arrays.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbDynamicTimeWarpingInterval : FB_ALY_DynamicTimeWarpingInterval(nNumChannels := 3);

    nWindowSize : UDINT := 100;
    bConfigure : BOOL := TRUE;

    fReference : LREAL;
    fInput1 : LREAL;
    fInput2 : LREAL;
    fInput3 : LREAL;

    aDistances : ARRAY[1..3] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbDynamicTimeWarpingInterval.Configure(nWindowSize);
END_IF

// Call algorithm
fbDynamicTimeWarpingInterval.SetChannelValue(0, fReference);
fbDynamicTimeWarpingInterval.SetChannelValue(1, fInput1);
fbDynamicTimeWarpingInterval.SetChannelValue(2, fInput2);
fbDynamicTimeWarpingInterval.SetChannelValue(3, fInput3);
fbDynamicTimeWarpingInterval.Call();
```

```
// Get results
fbDynamicTimeWarpingInterval.GetOutputArray(pArrayOut:=ADR(aDistances), nArrayOutSize:=SIZEOF(aDista
nces));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.4.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.4.4.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.4.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nWindowSize : UDINT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nWindowSize	UDINT	Gibt die Anzahl an Zyklen an, über die eine Berechnung erfolgt. Der Speicherbedarf des Algorithmus ist proportional zu diesem Parameter.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.4.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.4.5 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert entspricht der Distanz zwischen dem Referenzkanal und den Eingangskanal. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Der Wert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.4.6 GetOutputArray

Abholen des gesamten Ausgangs-Arrays. Die Array-Elemente entsprechen der Distanz zwischen dem Referenzkanal und den Eingangskanälen. Je geringer die Distanz, desto gleicher sind die verglichenen Signale. Die Anzahl an Elementen entspricht der konfigurierten Anzahl an Eingangskanälen. Die Werte werden nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetOutputArray : BOOL
VAR_INPUT
    pArrayOut : POITNER TO LREAL;
    nArrayOutSize : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pArrayOut	POINTER TO LREAL	Zeiger auf das Ausgangs-Array: Dim: nBufferSize
nArrayOutSize	UDINT	Größe des Ausgangs-Arrays

Rückgabewert

Name	Typ	Beschreibung
GetOutputArray	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.4.7 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 0: Referenzkanal 1 bis nChannels: Vergleichskanäle)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.5 FB_ALY_LogicOperationCounter

Der *Logic Operation Counter* führt eine logische Operation an den Werten von zwei oder mehr Kanälen aus und liefert das Ergebnis dieser logischen Operation. Dazu kann jeder Eingangswert mit einer Schwelle und einem Operator kombiniert werden. Des Weiteren können der logische Operator und der Zählmodus einzeln konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_LogicOperationCounter
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOperationOut: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bOperationOut	BOOL	Ergebnis der logischen Operation.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert bOperationOut=TRUE ist. Das Verhalten ist abhängig von dem Konfigurationsparameter eCountMode.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Wechsels von bOperationOut=TRUE.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbLogicOperationCounter : FB_ALY_LogicOperationCounter(nNumChannels := 3);
    fbSystemTime : FB_ALY_GetSystemTime;
    eLogicOperator : E_ALY_LogicOperator := E_ALY_LogicOperator.AND_;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.OnChange;
    stThresholdLevel : ARRAY[1..3] OF ST_ALY_Threshold :=
        [(eComparisonOperator := E_ALY_ComparisonOperator.GreaterThan, fThreshold := 10),
         (eComparisonOperator := E_ALY_ComparisonOperator.LessThan, fThreshold := 2),
         (eComparisonOperator := E_ALY_ComparisonOperator.Equals, fThreshold := 1)];
    bConfigure : BOOL := TRUE;
    nInputCh1 : INT := 11;
    fInputCh2 : LREAL := 1.5;
    bInputCh3 : BOOL := TRUE;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbLogicOperationCounter.ConfigureChannel(1, stThresholdLevel[1]);
    fbLogicOperationCounter.ConfigureChannel(2, stThresholdLevel[2]);
    fbLogicOperationCounter.ConfigureChannel(3, stThresholdLevel[3]);
    fbLogicOperationCounter.Configure(eLogicOperator, eCountMode);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbLogicOperationCounter.SetChannelValue(1, nInputCh1);
fbLogicOperationCounter.SetChannelValue(2, fInputCh2);
fbLogicOperationCounter.SetChannelValue(3, bInputCh3);

fbLogicOperationCounter.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.5.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.5.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eLogicOperator : E_ALY_LogicOperator;
    eCountMode : E_ALY_CountMode;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eLogicOperator	E_ALY_LogicOperator [▶ 354]	Konfigurierter logischer Operator.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf TRUE ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung TRUE ist.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.5.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.5.4 ConfigureChannel

Konfigurieren kanalspezifischer Parameter. Die Konfiguration wird erst verarbeitet, wenn die Methode Configure() aufgerufen wird.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT
    stThresholdLevel : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.5.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.5.6 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.4.6 FB_ALY_Multiplexer

Der Multiplexer wählt einen Kanal aus einem oder mehreren Eingangskanälen aus. Für jeden Eingangskanal muss zusätzlich ein boolscher Eingang bereitgestellt werden. Der Ausgang entspricht dem ersten Eingangskanal, wenn der bedingte Eingang TRUE ist. Die Priorität der konfigurierten Kanäle ist die Reihenfolge ihrer Konfiguration. Wenn die Bedingung für keinen der Kanäle erfüllt ist, wird der bereitgestellte Standardkanal zurückgegeben.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Multiplexer
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fResult: LREAL;
    nCurrentChannel: UDINT;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fResult	LREAL	Gibt das Signal des ausgewählten Eingangskanals aus.
nCurrentChannel	UDINT	Gibt die Nummer des ausgewählten Kanals an. Der Wert ist 0, wenn das Standardergebnis ausgewählt ist. Die Eingangskanäle werden in der Reihenfolge ihrer Konfiguration nummeriert.
nCount	ULINT	Beginnt mit 1 für den Kanal, der bei Beginn der Analyse ausgewählt wird, und zählt jedes Mal hoch, wenn ein anderer Kanal ausgewählt wird.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Kanalwechsels.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValues()	Local	Methode zur Übergabe kanalspezifischer Werte an den Algorithmus.
SetDefaultChannelValue()	Local	Methode zur Übergabe eines Werts an den Algorithmus, der als Standard ausgewählt wird, wenn kein Kanal ausgewählt wird.

Beispiel

```
VAR
    fbMultiplexer : FB_ALY_Multiplexer(nNumChannels := 3);
    fbSystemTime : FB_ALY_GetSystemTime;
    nInputCh1 : INT := 11;
    fInputCh2 : LREAL := 1.5;
    nInputCh3 : UDINT := 123;
    fDefault : LREAL := 3.1415;
    bConditionCh1 : BOOL;
    bConditionCh2 : BOOL;
    bConditionCh3 : BOOL;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Call algorithm
fbMultiplexer.SetChannelValues(1, bConditionCh1, nInputCh1);
fbMultiplexer.SetChannelValues(2, bConditionCh2, fInputCh2);
fbMultiplexer.SetChannelValues(3, bConditionCh3, nInputCh3);
fbMultiplexer.SetDefaultChannelValue(fDefault);

fbMultiplexer.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.6.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.4.6.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methoden SetChannelConditionAndValue () und SetChannelDefaultValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.6.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.6.4 SetChannelValues

Festlegen eines kanalspezifischen Eingangswerts und einer Bedingung. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValues: BOOL
VAR_INPUT
    nChannel : UDINT;
    bCondition : BOOL;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
bCondition	BOOL	Kanalbedingung.
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValues	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.4.6.5 SetDefaultChannelValue

Festlegen des Standardwerts, der gesetzt wird, wenn keine Kanalbedingung TRUE ist. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetDefaultChannelValue: BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetDefaultChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.4.7 FB_ALY_NumericalCompare_1Ch

Der *Numerical Compare 1Ch* vergleicht die Eingangswerte mit einem Referenzwert und liefert das Ergebnis dieser Vergleichsoperation. Der Operator, der Referenzwert und der Zählmodus können einzeln konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_NumericalCompare_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOperationOut: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bOperationOut	BOOL	Ergebnis der Vergleichsoperation.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert bOperationOut=TRUE ist. Das Verhalten ist abhängig von dem Konfigurationsparameter eCountMode.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Wechsels zu bOperationOut=TRUE.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbNumericalCompare : FB_ALY_NumericalCompare_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fReference : LREAL := 40;
    eComparisonOperator : E_ALY_ComparisonOperator := E_ALY_ComparisonOperator.GreaterThan;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.OnChange;
    bUseAbsValues : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbNumericalCompare.Configure(fReference, eComparisonOperator, eCountMode, bUseAbsValues);
```

```
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbNumericalCompare.SetChannelValue(nInput);
fbNumericalCompare.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.7.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.7.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fReference : LREAL;
    eComparisonOperator : E_ALY_ComparisonOperator;
    eCountMode : E_ALY_CountMode;
    bUseAbsValues : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fReference	LREAL	Referenzwert für die Vergleichsoperation.
eComparisonOperator	E_ALY_ComparisonOperator [▶ 353]	Gibt an, ob der Eingangswert größer, größer oder gleich, gleich, kleiner oder gleich, kleiner oder ungleich dem Referenzwert sein soll.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf `TRUE` ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung `TRUE` ist.
bUseAbsValues	BOOL	Wenn `TRUE`, werden die absoluten Werte der Eingangssignale und der Referenz verwendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.7.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.7.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.8 FB_ALY_NumericalCompare_2Ch

Numerical Compare 2Ch vergleicht die Eingangswerte des ersten Kanals mit den Eingangswerten des zweiten Kanals und liefert das Ergebnis dieser Vergleichsoperation. Der Operator und der Zählmodus können einzeln konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_NumericalCompare_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bOperationOut: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bOperationOut	BOOL	Ergebnis der Vergleichsoperation.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert von `bOperationOut=TRUE` ist. Das Verhalten ist abhängig von dem Konfigurationsparameter `eCountMode`.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Wechsels zu `bOperationOut=TRUE`.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbNumericalCompare : FB_ALY_NumericalCompare_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    eComparisonOperator : E_ALY_ComparisonOperator := E_ALY_ComparisonOperator.GreaterThan;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.OnChange;
    bUseAbsValues : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInputCh1 : INT;
    fInputCh2 : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbNumericalCompare.Configure(eComparisonOperator, eCountMode, bUseAbsValues);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbNumericalCompare.SetChannelValue(1, nInputCh1);
fbNumericalCompare.SetChannelValue(2, fInputCh2);
fbNumericalCompare.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.8.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.8.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eComparisonOperator : E_ALY_ComparisonOperator;
    eCountMode : E_ALY_CountMode;
    bUseAbsValues : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eComparisonOperator	E_ALY_ComparisonOperator [▶ 353]	Gibt an, ob der Eingangswert größer, größer oder gleich, gleich, kleiner oder gleich, kleiner oder ungleich dem Referenzwert sein soll.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf TRUE ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung TRUE ist.
bUseAbsValues	BOOL	Wenn TRUE, werden die absoluten Werte der Eingangssignale verwendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.8.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.8.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Kanal 1* 2: *Kanal 2*
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.4.9 FB_ALY_StringCompare_1Ch

Der *String Compare 1Ch* vergleicht den Eingangsstring mit einem Referenzstring und zählt die Stringübereinstimmungen. Dabei kann die Groß- und Kleinschreibung beachtet werden oder nicht und der Zählmodus geändert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StringCompare_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bStringMatch: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bStringMatch	BOOL	Ergebnis des Stringvergleichs.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert von `bStringMatch=TRUE` ist. Das Verhalten ist abhängig von dem Konfigurationsparameter `eCountMode`.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Wechsels zu `bStringMatch=TRUE`.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbStringCompare : FB_ALY_StringCompare_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    sReference : STRING := 'String to compare with';
    eStringCompareMode : E_ALY_StringCompareMode := E_ALY_StringCompareMode.Equals;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.OnChange;
    bCaseSensitive : BOOL := TRUE;
    bConfigure : BOOL := TRUE;
    sInput : STRING := 'Input string';
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbStringCompare.Configure(sReference, eStringCompareMode, eCountMode, bCaseSensitive);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbStringCompare.SetChannelValue(sInput);
fbStringCompare.Call(fbSystemTime.tSystemTime)
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.9.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.9.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sReference : STRING(255);
    eStringCompareMode : E_ALY_StringCompareMode;
    eCountMode : E_ALY_CountMode;
    bCaseSensitive : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sReference	STRING(255)	Referenzstring für den Vergleich.
eStringCompareMode	E_ALY_StringCompareMode [▶ 355]	String compare mode: *Equals*: Eingangsstring entspricht dem Referenzstring. *BeginsWith*: Eingangsstring beginnt mit dem Referenzstring. *Contains*: Eingangsstring enthält den Referenzstring.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf TRUE ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung TRUE ist.
bCaseSensitive	BOOL	Wenn TRUE, wird die Groß- und Kleinschreibung beachtet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.9.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.9.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY_STRING;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY_STRING	Eingangswert eines String-Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.10 FB_ALY_StringCompare_2Ch

Der *String Compare 2Ch* vergleicht die Werte des ersten Eingangsstrings mit den Werten des zweiten Strings und zählt die Stringübereinstimmungen. Dabei kann die Groß- und Kleinschreibung beachtet werden oder nicht und der Zählmodus geändert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StringCompare_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bStringMatch: BOOL;
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bStringMatch	BOOL	Ergebnis des Stringvergleichs.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert von bStringMatch=TRUE ist. Das Verhalten ist abhängig von dem Konfigurationsparameter eCountMode.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel des letzten Wechsels zu bStringMatch=TRUE.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbStringCompare : FB_ALY_StringCompare_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    eStringCompareMode : E_ALY_StringCompareMode := E_ALY_StringCompareMode.Equals;
    eCountMode : E_ALY_CountMode := E_ALY_CountMode.OnChange;
    bCaseSensitive : BOOL := TRUE;
    bConfigure : BOOL := TRUE;
    sInputCh1 : STRING := 'Input string';
    sInputCh2 : STRING := 'String to compare with';
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbStringCompare.Configure(eStringCompareMode, eCountMode, bCaseSensitive);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbStringCompare.SetChannelValue(1, sInputCh1);
fbStringCompare.SetChannelValue(2, sInputCh2);
fbStringCompare.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.10.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.10.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eStringCompareMode : E_ALY_StringCompareMode;
    eCountMode : E_ALY_CountMode;
    bCaseSensitive : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eStringCompareMode	E_ALY_StringCompareMode [▶ 355]	String compare mode: *Equals*: Eingangsstring entspricht dem Referenzstring. *BeginsWith*: Eingangsstring beginnt mit dem Referenzstring. *Contains*: Eingangsstring enthält den Referenzstring.
eCountMode	E_ALY_CountMode [▶ 354]	Modus des Ergebniszählers. *OnChange*: Der Zähler zählt jedes Mal, wenn sich das Ergebnis auf TRUE ändert. *Cyclic*: Der Zähler inkrementiert jeden Zyklus, wenn die Bedingung TRUE ist.
bCaseSensitive	BOOL	Wenn TRUE, wird die Groß- und Kleinschreibung beachtet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.10.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.10.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: *Kanal 1* 2: *Kanal 2*
Input	ANY_STRING	Eingangswert eines String-Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.11 FB_ALY_DetectValueChange_1Ch

Der *Detect Value Change 1Ch* erkennt und zählt Änderungen der numerischen Eingangswerte.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DetectValueChange_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bValueChanged: BOOL;
```

```
    nCount: ULINT;
    fbTimeLastEvent: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bValueChanged	BOOL	TRUE, wenn eine Wertänderung erkannt wurde, ansonsten FALSE.
nCount	ULINT	Wird inkrementiert, wenn der Ausgangswert von bValueChanged=TRUE ist.
fbTimeLastEvent	FB_ALY_DateTime	Zeitstempel der zuletzt erkannten Wertänderung.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbDetectValueChange : FB_ALY_DetectValueChange_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    fTolerance : LREAL := 0.1;
    bConfigure : BOOL := TRUE;
    fIn : LREAL;
END_VAR
```

```
// Get current system time
fbSystemTime.Call();

// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbDetectValueChange.Configure(fTolerance);
END_IF

// Call algorithm
fbDetectValueChange.SetChannelValue(fIn);
fbDetectValueChange.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.4.11.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.11.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fTolerance : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fTolerance	LREAL	Die Toleranz bezieht sich auf den Eingangswert der zuletzt erkannten Wertänderung. Ist der Eingangswert außerhalb dieser Toleranz, wir eine Wertänderung erkannt.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.4.11.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.4.11.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5 Math

5.1.1.5.1 FB_ALY_Integrator_1Ch

Der *Integrator 1Ch* integriert den Eingangswert über die Zeit mit einer Basiseinheit von einer Sekunde und liefert das Ergebnis dieser Integrationsoperation. Für die Näherung dieses Integrals wird die Trapezregel angewendet. Das Trapez $T(t_n, t_{n+1})$ zwischen zwei aufeinanderfolgenden Zeitstempeln t_n und t_{n+1} mit den Werten y_n und y_{n+1} wird berechnet als

$$T(t_n, t_{n+1}) = (t_{n+1}[s] - t_n[s]) \cdot \frac{y_n + y_{n+1}}{2}$$

.

Wenn der Integrationsmodus „absolut“ („|x|“) in der Konfiguration gewählt wird, werden y_n und y_{n+1} in der obigen Gleichung durch ihre absoluten Werte ersetzt.

In jedem Zyklus wird das Trapez zwischen dem aktuellen und dem letzten Zeitstempel berechnet und zu der Summe der Trapeze ab dem Beginn der Analyse addiert. Zusätzlich kann diese Summe um einen Faktor skaliert werden, der einzeln konfiguriert werden kann.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_Integrator_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fResult: LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fResult	LREAL	Gibt das Ergebnis der Integration aus.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbIntegrator : FB_ALY_Integrator_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    eIntegrationMode : E_ALY_IntegrationMode := E_ALY_IntegrationMode.Direct;
    fFactor : LREAL := 1.0;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbIntegrator.Configure(eIntegrationMode, fFactor);
END_IF

// Get current system time
fbSystemTime.Call();
```

```
// Call algorithm
fbIntegrator.SetChannelValue(nInput);
fbIntegrator.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.5.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.1.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eIntegrationMode : E_ALY_IntegrationMode;
    fFactor : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eIntegrationMode	E_ALY_IntegrationMode [▶ 354]	Integration mode *Direct*: Integration der Eingangswerte *Absolut*: Integration der absoluten Eingangswerte
fFactor	LREAL	Mit diesem Faktor wird das Integral multipliziert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.5.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.2 FB_ALY_MathOperation

Der *Math Operation* führt eine mathematische Operation an zwei oder mehr verschiedenen Eingangskanälen aus und liefert das Ergebnis der mathematischen Operation. Der Operator ist für alle Operanden gleich und kann einzeln konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MathOperation
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fResult: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fResult	LREAL	Gibt das Ergebnis der mathematischen Operation aus.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbMathOperation : FB_ALY_MathOperation(nNumChannels := 3);
    eMathOperator : E_ALY_MathOperator := E_ALY_MathOperator.Addition;
    bUseAbsValues : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInputCh1 : INT;
    fInputCh2 : LREAL;
    nInputCh3 : UDINT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbMathOperation.Configure(eMathOperator, bUseAbsValues);
END_IF

// Call algorithm
fbMathOperation.SetChannelValue(1, nInputCh1);
fbMathOperation.SetChannelValue(2, fInputCh2);
fbMathOperation.SetChannelValue(3, nInputCh3);
fbMathOperation.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.5.2.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.5.2.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.2.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eMathOperator : E_ALY_MathOperator;
    bUseAbsValues : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eMathOperator	E_ALY_MathOperator [▶ 355]	Mathematical operator *Addition* *Subtraction* *Multiplication* *Division* *PowerOf* *Modulo*
bUseAbsValues	BOOL	Wenn TRUE, werden die absoluten Werte der Eingangssignale verwendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.2.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.5.2.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.3 FB_ALY_MathOperation_1Ch

Der Math Operation 1Ch führt eine mathematische Operation an dem Signal des Eingangskanals und einem Referenzwert aus. Der Algorithmus liefert das Ergebnis der mathematischen Operation und der Operator kann einzeln konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_MathOperation_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fResult: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fResult	LREAL	Gibt das Ergebnis der mathematischen Operation aus.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbMathOperation : FB_ALY_MathOperation_1Ch;
    fOperand : LREAL := 50;
    eMathOperator : E_ALY_MathOperator := E_ALY_MathOperator.Addition;
    bUseAbsValues : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbMathOperation.Configure(fOperand, eMathOperator, bUseAbsValues);
END_IF

// Call algorithm
fbMathOperation.SetChannelValue(nInput);
fbMathOperation.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.5.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.3.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    fOperand : LREAL;
    eMathOperator : E_ALY_MathOperator;
    bUseAbsValues : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fOperand	LREAL	Operand für die mathematische Operation
eMathOperator	E_ALY_MathOperator [▶ 355]	Mathematical operator *Addition* *Subtraction* *Multiplication* *Division* *PowerOf* *Modulo*
bUseAbsValues	BOOL	Wenn TRUE, werden die absoluten Werte der Eingangssignale verwendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.3.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.5.3.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.4 FB_ALY_RMS_1Ch

Der *RMS 1Ch* berechnet das quadratische Mittel (root mean square) über die Eingangswerte nach der Formel

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x[n]^2}$$

Die Anzahl an Samples N, die mit in die Berechnung einfließen, kann über die Angabe eines Zeitintervalls konfiguriert werden. Ein kaskadierter Ausgang kann konfiguriert werden, um einen Langzeit-RMS ressourcenschonend zu realisieren und Zwischenergebnisse abzugreifen. Das Zeitintervall der konfigurierten Kaskade muss einem ganzzahligen Vielfachen des Zeitintervalls der vorherigen Kaskade entsprechen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_RMS_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
SetCascades()	Local	Methode zur Konfiguration der Kaskaden. Der Aufruf erfolgt nach dem Aufruf der Configure-Methode.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbRMS_1Ch : FB_ALY_RMS_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nNumCascades : UDINT := 5;
    fSampleRate : UDINT := 1000;
    eStartupBehaviour : E_ALY_CascadeStartupBehaviour := E_ALY_CascadeStartupBehaviour.UsePreviousCa
scadeValue;
    aCascadesConfigArray : ARRAY[1..5] OF LTIME := [LTIME#20MS, LTIME#1S, LTIME#1M, LTIME#1H, LTIME#
1D];
    bConfigure : BOOL := TRUE;
    fInput : LREAL;
    aRMS : ARRAY[1..5] OF LREAL;
    aNewResult : ARRAY[1..5] OF BOOL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbRMS_1Ch.Configure(nNumCascades, fSampleRate, eStartupBehaviour);
    fbRMS_1Ch.SetCascades(ADR(aCascadesConfigArray), SIZEOF(aCascadesConfigArray));
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbRMS_1Ch.SetChannelValue(fInput);
fbRMS_1Ch.Call(fbSystemTime.tSystemTime, ADR(aRMS), SIZEOF(aRMS), ADR(aNewResult), SIZEOF(aNewResult
));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.5.4.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
    pRmsArrayOut : POINTER TO LREAL;
    nRmsArrayOutSize : UDINT;
    pNewResultArrayOut : POINTER TO BOOL;
    nNewResultArrayOutSize : UDINT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.
pRmsArrayOut	POINTER TO LREAL	Zeiger auf ein Array in dem die RMS-Ergebnisse gespeichert werden sollen. Die Dimension entspricht der Anzahl konfigurierter Kaskaden.
nRmsArrayOutSize	UDINT	Größe des RMS-Arrays.
pNewResultArrayOut	POINTER TO BOOL	Zeiger auf ein Array in dem die Stati neuer RMS-Ergebnisse gespeichert werden sollen. Die Dimension entspricht der Anzahl konfigurierter Kaskaden.
nNewResultArrayOutSize	UDINT	Größe des Status-Arrays.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.4.2 Configure

Konfigurieren des Algorithmus. Die Ausgangskaskaden können über die Methode SetCascades() festgelegt werden.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumCascades : UDINT;
    fSampleRate : LREAL;
    eStartupBehaviour : E_ALY_CascadeStartupBehaviour;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumCascades	UDINT	Anzahl an Ausgangskaskaden
fSampleRate	LREAL	Samplerate des zu analysierenden Systems in Hz
eStartupBehaviour	E_ALY_CascadeStartupBehaviour [▶ 353]	Hochlaufverhalten *WaitUntilFilled* Wartet ab, bis die konfigurierte Zeitspanne der Kaskade abgelaufen ist. Erst nach Ablauf der Zeitspanne wird das RMS-Ergebnis als auch das boolesche Flag „NewResult“ erstmalig gesetzt. *UsePreviousCascadeValue* Die RMS-Kaskaden, deren konfigurierte Zeitspanne noch nicht abgelaufen ist, verwenden das nächstkleinere bereits gesetzte RMS-Ergebnis.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.4.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.5.4.4 SetCascades

Konfiguration der Ausgangskaskaden. Zuvor muss die Anzahl an Kaskaden mit der Configure()-Methode festgelegt werden.

Syntax

Definition:

```
METHOD SetCascades : BOOL
VAR_INPUT
    pCascadesConfigArray : POINTER TO LTIME;
    nCascadesConfigArraySize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pCascadesConfigArray	POINTER TO LTIME	Zeiger auf ein Array, mit dem die Zeitintervalle der Kaskaden konfiguriert werden. Das Zeitintervall der konfigurierten Kaskade muss einem ganzzahligen Vielfachen des Zeitintervalls der vorherigen Kaskade entsprechen.
nCascadesConfigArraySize	UDINT	Größe des Konfigurations-Arrays.

Rückgabewert

Name	Typ	Beschreibung
SetCascades	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.4.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.5.5 FB_ALY_SlopeAnalysis_1Ch

Der *Slope Analysis 1Ch* berechnet die Steigung zwischen zwei Werten des Eingangsstroms. Einer dieser beiden Werte ist der aktuelle Eingangswert und der zweite Wert ist der Eingangswert, der eine festgelegte Anzahl (konfiguriert durch den Parameter *Num Values*) von Zyklen zuvor im Eingangsstrom aufgetreten ist. Die Differenz zwischen diesen beiden Werten wird als *Delta Value* zurückgegeben.

Der entsprechende Abstand auf der Zeitkoordinate wird als Differenz der Zeitstempel dieser beiden Werte berechnet und als Ausgangswert *Delta Time* bereitgestellt. Zu beachten ist, dass der Wert *Delta Time* in Nanosekunden angezeigt wird, für die Berechnung der Steigung jedoch auf eine Sekunde als Basiseinheit skaliert wird.

Der *Slope* wird dann als Bruch von *Delta Value* und *Delta Time* (skaliert auf Sekunden) berechnet und der Gradient für den Zeitstempel in der Mitte der beiden in der Berechnung von *Delta Time* verwendeten Zeitstempel geschätzt. Dies ist der Wert, der als *Time Slope* zurückgegeben wird, wenn er einem Zeitstempel des Eingangsstroms entspricht. Bei Konfigurationen, bei denen *Num Values* eine ungerade Zahl ist, gibt es keinen Eingangswert, der exakt mit dem Zeitstempel in der Mitte übereinstimmt. In diesem Fall wird der Zeitstempel des Werts, der direkt auf den berechneten Zeitstempel in der Mitte gefolgt ist, als *Time Slope* zurückgegeben.

Des Weiteren liefert der Algorithmus die minimale Steigung, die maximale Steigung und die Zeitwerte von Minimum und Maximum.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_SlopeAnalysis_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
```

```
    bNewResult: BOOL;
    bConfigured: BOOL;
    fSlope: LREAL;
    fSlopeMin: LREAL;
    fSlopeMax: LREAL;
    fDeltaValue: LREAL;
    fbDeltaTime: FB_ALY_Timespan;
    fbTimeSlope: FB_ALY_DateTime;
    fbTimeSlopeMin: FB_ALY_DateTime;
    fbTimeSlopeMax: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fDeltaValue	LREAL	Differenz der Werte des ersten und letzten Elements des Fensters.
fbDeltaTime	FB_ALY_Timespan	Differenz der Zeitstempelwerte des ersten und letzten Elements des Fensters. Sie wird auf Sekunden skaliert.
fSlope	LREAL	Division von fDeltaValue und fDeltaTime.
fSlopeMin	LREAL	Minimaler aufgetretener Wert von fSlope.
fSlopeMax	LREAL	Maximaler aufgetretener Wert von fSlope.
fbTimeSlope	FB_ALY_DateTime	Zeitstempel der berechneten Steigung. Er wird in der Mitte des konfigurierten Fensters platziert.
fbTimeSlopeMin	FB_ALY_DateTime	Zeitstempel des zuletzt aktualisierten fSlopeMin.
fbTimeSlopeMax	FB_ALY_DateTime	Zeitstempel des zuletzt aktualisierten fSlopeMax.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbSlopeAnalysis : FB_ALY_SlopeAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nNumValues : UDINT := 200;
```

```
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbSlopeAnalysis.Configure(nNumValues);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbSlopeAnalysis.SetChannelValue(nInput);
fbSlopeAnalysis.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.5.5.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.5.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumValues : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumValues	UDINT	Abstand zwischen den zwei Datensätzen für die Berechnung der Steigung.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.5.5.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.5.5.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6 Statistics

5.1.1.6.1 FB_ALY_ArrayStatistics

Der Algorithmus *Array Statistics* berechnet verschiedene statistische Größen auf Basis des Eingangs-Arrays.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ArrayStatistics
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
```

```
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fMin: LREAL;
    nIdxMin: UDINT
    fMax: LREAL;
    nIdxMax: UDINT;
    fMaxDelta: LREAL;
    nIdxMaxDelta: UDINT;
    nCountPeaks: ULINT;
    nCountValleys: ULINT;
    fSum: LREAL;
    fMean: LREAL;
    fStandardDeviation: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fMin	LREAL	Kleinster Wert im Eingangsarray.
nIdxMin	UDINT	Array-Index von `fMin`. Die Indexierung beginnt bei 1.
fMax	LREAL	Größter Wert im Eingangsarray.
nIdxMax	UDINT	Array-Index von `fMax`. Die Indexierung beginnt bei 1.
fMaxDelta	LREAL	Maximum der absoluten Differenz zwischen zwei aufeinanderfolgenden Werten im Eingangsarray.
nIdxMaxDelta	UDINT	Array-Index von `fMaxDelta`. Die Indexierung beginnt bei 1.
nCountPeaks	ULINT	Gesamtanzahl der identifizierten Peaks.
nCountValleys	ULINT	Gesamtanzahl der identifizierten Täler.
fSum	LREAL	Summe über das gesamte Eingangsarray.
fMean	LREAL	Mittelwert über das gesamte Eingangsarray.
fStandardDeviation	LREAL	Standardabweichung über das gesamte Eingangsarray.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen. Die Konfiguration ist für alle Kanäle identisch.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.

Beispiel

```
VAR
    fbArrayStatistics : FB_ALY_ArrayStatistics;
    bUseBesselCorrection : BOOL := TRUE;
    fThresholdReversal : LREAL := 0.5;
    fThresholdDelta : LREAL := 0.5;
```

```
    bConfigure : BOOL := TRUE;
    aInput : ARRAY[1..20] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbArrayStatistics.Configure(bUseBesselCorrection := bUseBesselCorrection, fThresholdReversal :=
fThresholdReversal, fThresholdDelta := fThresholdDelta);
END_IF

// Call algorithm
fbArrayStatistics.Call(ADR(aInput), SIZEOF(aInput));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.1.1 Call

Aufrufen des Algorithmus.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    pArrayIn : PVOID;
    nArrayInSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pArrayIn	PVOID	Zeiger auf das Eingangs-Array.
nArrayInSize	UDINT	Größe des Eingangs-Arrays

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.1.2 Configure

Konfigurieren des Algorithmus.

Konfigurationsoptionen

- **Use Bessel Correction:** Wenn die Checkbox aktiviert ist, wird die Bessel‘sche Korrektur angewendet. Um bei Stichproben ein erwartungstreues Ergebnis zu erhalten, ist dieser Parameter zu aktivieren. Der Parameter ist nur für die Berechnung der Standardabweichung relevant.

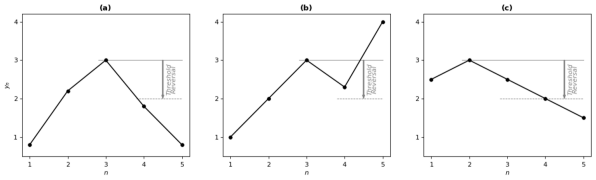
Die empirische Standardabweichung, ohne Bessel’sche Korrektur

$$s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n] - \overline{x})^2}$$

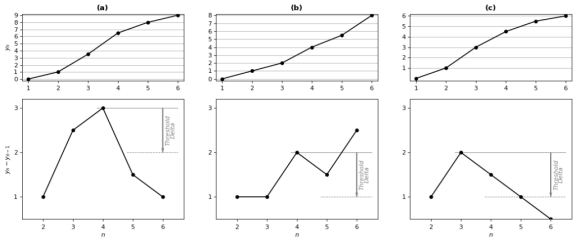
Die empirische Standardabweichung, mit Bessel’scher Korrektur

$$s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$$

- **Threshold Reversal:** Schwellenwert für die Identifizierung von Umkehrungen. Umkehrungen werden nur erkannt, wenn ihre Differenz zur nächsten Umkehrung den Wert von Threshold Reversal überschreitet.
  Nachfolgend sind drei Beispiele für die Identifizierung von Peaks mit dem Parameter *Threshold Reversal* aufgeführt.
  (a) Der Wert y_3 wird direkt nach der Verarbeitung des Werts y_4 als Peak identifiziert, da die Differenz zwischen y_3 und y_4 größer ist als *Threshold Reversal*.
  (b) Der Wert y_3 wird nicht als Peak identifiziert, da die Differenz zwischen y_3 und y_4 kleiner ist als *Threshold Reversal* und die Kurve nach y_4 wieder ansteigt.
  (c) Der Wert y_2 wird nach der Verarbeitung des Werts y_5 als Peak identifiziert, da die Differenz zwischen y_2 und y_5 *Threshold Reversal* überschreitet. Der Wert y_2 kann vorher nicht als Peak identifiziert werden, da die Differenz zwischen y_2 und y_3 (y_4) kleiner als/gleich *Threshold Reversal* ist und nicht bekannt ist, ob die Werte weiter sinken.



- **Threshold Delta:** Schwellenwert für die Identifizierung der Delta-Maxima. Maxima der absoluten Differenz von zwei aufeinanderfolgenden Werten (Delta) werden nur erkannt, wenn die Differenz zwischen aufeinanderfolgenden Deltas *Threshold Delta* überschreitet.
  Nachfolgend sind drei Beispiele für die Identifizierung der Delta-Maxima mit dem Parameter *Threshold Delta* aufgeführt. Die oberen Diagramme zeigen die ursprünglichen Eingangssignale, die unteren das zugehörige Delta.
  (a) Der Wert y_4 wird nach der Verarbeitung des Werts y_5 als Maximum identifiziert, da die Differenz der zwei Deltas *Threshold Delta* überschreitet.
  (b) Es wird kein Maximum identifiziert, da die Differenz zwischen den Deltas kleiner ist als *Threshold Delta*.
  (c) Der Wert y_3 wird nach der Verarbeitung des Werts y_6 als Maximum identifiziert.



Unabhängig von *Threshold Delta* wird mindestens ein Maximum des Deltas zwischen zwei Umkehrungen erkannt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bUseBesselCorrection : BOOL;
    fThresholdReversal : LREAL;
    fThresholdDelta : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bUseBesselCorrection	BOOL	Nutzung der Bessel-Korrektur (s. o.).
fThresholdReversal	LREAL	Schwellenwert für die Identifizierung von Umkehrungen (s. o.).
fThresholdDelta	LREAL	Schwellenwert für die Identifizierung der Delta-Maxima (s. o.).

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.2 FB_ALY_CorrelationFunction

Der *Correlation Function* Baustein berechnet die diskrete Korrelationsfunktion zwischen einem Referenzsignal (Channel Ref) und einem oder mehreren weiteren Signalen (Channel 00, …, Channel 0n). Die Korrelationskoeffizienten werden für Zeitverschiebungen von *m* Zyklen zwischen den beiden Signalen berechnet, wobei der maximale und der minimale Wert für *m* durch die Parameter *Minimum Lag* (negative ganze Zahl) und *Maximum Lag* (positive ganze Zahl) begrenzt ist.

Durch den Parameter *StepSize* wird bestimmt, um wie viele Zyklen die Signale für die Berechnung zweier aufeinanderfolgender Korrelationskoeffizienten verschoben werden. D.h. *m* ist immer ein Vielfaches der *StepSize*. Dementsprechend sind für *Minimum Lag* und *Maximum Lag* auch nur Vielfache der *StepSize* zulässig. Ist die *StepSize* auf eins gesetzt und beispielsweise *Minimum Lag* auf -6 sowie *Maximum Lag* auf +4, so werden Korrelationskoeffizienten für Verschiebungen um -6, -5, -4, -3, -2, -1, 0, +1, +2, +3 und +4 Zyklen berechnet. Setzt man die *StepSize* auf zwei, werden Koeffizienten für Verschiebungen um -6, -4, -2, 0, +2 und +4 Zyklen berechnet.

Die Koeffizienten können über verschiedene Zeitfenster berechnet werden. Diese werden durch den Parameter *Window Mode* gesetzt. Im *Continuous* Modus werden alle Werte seit Beginn der Analyse in die Berechnungen miteinbezogen. Im *SlidingWindow* Modus läuft die Berechnung kontinuierlich über die letzten, durch die *Window Size* gesetzte Anzahl von Zyklen. Im *FixWindow* Modus erfolgt die Berechnung auch über die durch die *Window Size* gesetzte Anzahl von Zyklen. Jedoch startet die Berechnung nach jeden *WindowSize* Zyklen neu und die Ausgangswerte werden nur beim Durchlaufen des letzten Zyklus eines Fensters aktualisiert.

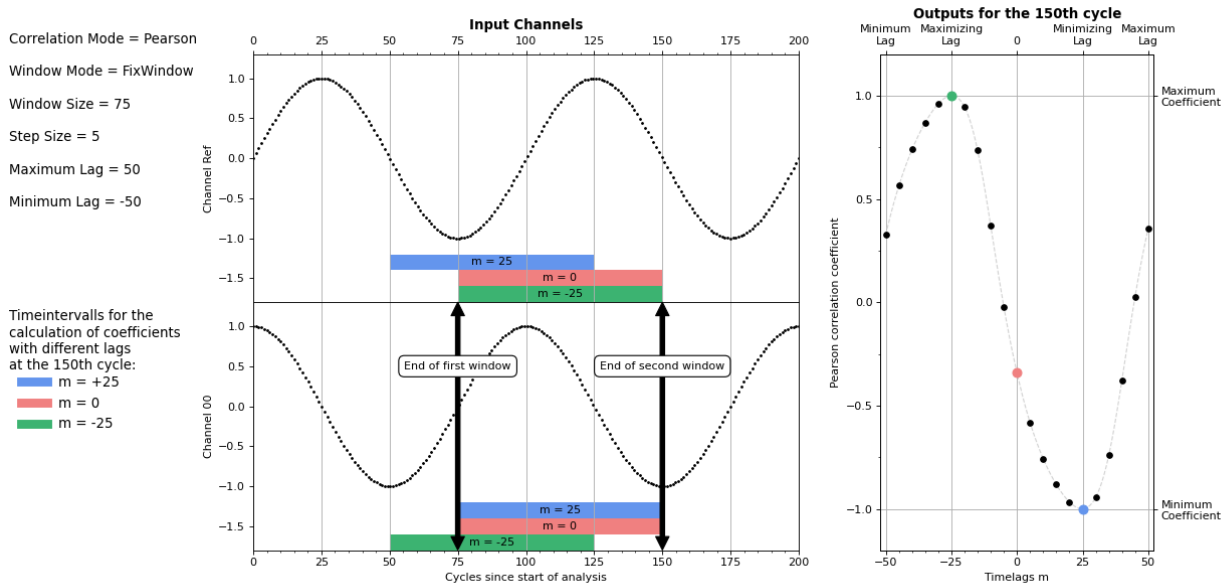
Für *m* ungleich Null verschiebt sich das Fenster für das entsprechende Signal um *m* Zyklen. Damit bleibt die Anzahl von Werten, welche in die Berechnung der Koeffizienten mit einbezogen werden, für alle Werte von *m* gleich. Eine Ausnahme hierzu bilden nur die Ergebnisse aus den ersten Zyklen nach Start der Analyse. Ist die Anzahl der verstrichenen Zyklen kleiner als |*m*|, werden entsprechend weniger Werte in die Berechnung mit einbezogen.

Für positive Werte von *m* werden die Werte des Referenzsignals (Channel Ref) in einem Ringspeicher gespeichert, damit jeweils der vor *m* Zyklen eingegangene Wert des Referenzsignals mit den aktuellen Werten von Channel 00 bis Channel 0n verglichen werden kann. Dies entspricht einer Verschiebung des

Referenzsignals in die Vergangenheit. Für negative Werte von *m* müsste das Referenzsignal dementsprechend in die Zukunft verschoben werden. Da dies offensichtlich nicht möglich ist, wird stattdessen das zweite Signal (Channel 00,..Channel0n) gespeichert und nach hinten verschoben.

Die Korrelationskoeffizienten können nach unterschiedlichen Berechnungsvorschriften berechnet werden. Diese wird durch den *Correlation Mode* bestimmt. In den Modi *Base* und *Normed* werden die Koeffizienten analog zu der Definition aus der Signalverarbeitung berechnet, welche die Korrelation über die Faltung berechnet. Im *Normed* – Modus werden die Koeffizienten zudem durch die Anzahl der Summanden geteilt. *Covariance* und *CovarianceBessel* berechnen die Kovarianz ohne und mit Bessel-Korrektur. Der Modus *Pearson* nutzt die Definition des in der Statistik üblichen Pearson Korrelationskoeffizienten. In den Konfigurationsoptionen sind die genauen Berechnungsvorschriften mathematisch aufgeführt. Hier bezeichnet x_n den Wert des Referenzsignals und y_n den Wert des zweiten Signals (jeweils Channel00, …, Channel0n) zum Zeitstempel t_n (entsprechend dem n-ten Zyklus seit Start der Analyse oder seit Reset, ausgenommen der Zyklen in denen *Enable Execution* = FALSE). Der Wert von *N* hängt von dem gewählten *WindowMode* ab. Für den *SlidingWindow*-Mode und *FixWindow*-Mode ist *N* gleich der *Window Size*, vorausgesetzt seit Start der Analyse sind bereits entsprechend viele Zyklen vergangen, sodass $x_{n\text{-}N\text{-}m}$ (bzw. $y_{n\text{-}N+m}$) aufgenommen wurde, andernfalls reduziert sich *N* zu *n-m+1* bzw. *n+m+1*. Zu beachten ist, dass im *FixWindow*-Mode die Ausgangswerte nur alle *Window Size* Zyklen aktualisiert werden. Im *Continuous*-Mode gilt stets *N* = *n+1*.

In der Abbildung sind beispielhaft für eine Konfiguration (*Correlation Mode* = *Pearson*, *Window Mode* = *FixWindow*, *Window Size* = 75, *Step Size* = 5, *Maximum Lag* = 50, *Minimum Lag* = -50) die Ausgangswerte des Bausteins für zwei Signale (Channel Ref und Channel 00) für einen bestimmten Zyklus (n=150) dargestellt. In den beiden linken Plots sind die Eingangssignale Channel Ref und Channel 00 im Verlauf der Zeit dargestellt. Im rechten Plot ist die diskrete Korrelationsfunktion (die Pearson-Korrelationskoeffizienten in Abhängigkeit von m) zu sehen. Es sind Koeffizienten für die Verschiebungen m= -50, -45, -40, -35, -30, -25, -20, -15, -10, -5, 0, +5, +10, +15, +20, +25, +30, +35, +40, +45, +50 dargestellt. Diese werden im Baustein als Array ausgegeben. Neben den beiden Parametern Minimum und Maximum Lag, sind auf der Abszisse die Ausgangswerte Minimizing und Maximizing Lag gekennzeichent. Die dazugehörigen Koeffizienten Minimum und Maximum Coefficient, welche ebenfalls Ausgänge des Bausteins darstellen, sind auf der Ordinate markiert. Exemplarisch sind für die Verschiebungen m = -25, m = 0 und m =+25 in den Plots der Eingangskanäle (links) die Zeitbereiche farblich hervorgehoben, welche in die Berechnung des jeweiligen Koeffizienten eingehen. Im rechten Plot sind die entsprechenden Punkte jeweils eingefärbt.



Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_CorrelationFunction
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
```

```
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputArray()	Local	Abrufen des Ergebnis-Arrays für einen bestimmten Kanal, ohne neue Werte hinzuzufügen.
GetChannelOutputValues()	Local	Abrufen der Ergebnisse (Einzelwerte) für einen bestimmten Kanal, ohne neue Werte hinzuzufügen.

Beispiel

```
VAR
    fbCorrelationFunction : FB_ALY_CorrelationFunction(nMinLag := -50, nMaxLag := 50, nStepSize := 5
, nNumChannels := 1, eWindowMode := E_ALY_WindowMode.FixWindow, eCorrelationMode := E_ALY_Correlatio
nMode.Pearson, nWindowSize := 75);

    // Configuration
    nMinLag : DINT := -50;
    nMaxLag : DINT := 50;
    nStepSize : UDINT := 5;
    nNumChannels : UDINT := 1;
    eWindowMode : E_ALY_WindowMode := E_ALY_WindowMode.FixWindow;
    eCorrelationMode : E_ALY_CorrelationMode := E_ALY_CorrelationMode.Pearson;
    nWindowSize : UDINT := 75;

    bConfigure : BOOL := TRUE;

    // Inputs
    nInputRef : INT;
    fInputCh1 : LREAL;

    // Results
    fMinCoef : LREAL;
    fMaxCoef : LREAL;
    nMinimizingLag : DINT;
    nMaximizingLag : DINT;
    aCoefficients : ARRAY[-10..10] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
```

```
    fbCorrelationFunction.Configure(
        nMinLag := nMinLag,
        nMaxLag := nMaxLag,
        nStepSize := nStepSize,
        nNumChannels := nNumChannels,
        eWindowMode := eWindowMode,
        eCorrelationMode := eCorrelationMode,
        nWindowSize := nWindowSize);
END_IF

// Call algorithm
fbCorrelationFunction.SetChannelValue(0, nInputRef);
fbCorrelationFunction.SetChannelValue(1, fInputCh1);
fbCorrelationFunction.Call();

fbCorrelationFunction.GetChannelOutputArray(1, ADR(aCoefficients), SIZEOF(aCoefficients));
fbCorrelationFunction.GetChannelOutputValues(1, fMinCoef, fMaxCoef, nMinimizingLag, nMaximizingLag);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.2.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.2.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.2.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nMinLag : DINT;
    nMaxLag : DINT;
    nStepSize : UDINT;
    nNumChannels : UDINT;
    eWindowMode : E_ALY_WindowMode;
    eCorrelationMode : E_ALY_CorrelationMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nMinLag	UDINT	Gibt die minimale Anzahl an Zyklen an, um die die beiden Signale für die Berechnung der Korrelation zueinander verschoben werden. Dies ist eine negative ganze Zahl.
nMaxLag	UDINT	Gibt die maximale Anzahl an Zyklen an, um die die beiden Signale für die Berechnung der Korrelation zueinander verschoben werden. Dies ist eine positive ganze Zahl.
nStepSize	UDINT	Gibt an, um wie viele Zyklen die Signale für die Berechnung zweier aufeinanderfolgender Korrelationskoeffizienten verschoben werden.
nNumChannels	UDINT	Die Anzahl der Kanäle, die mit dem Referenzsignal korreliert werden.
eWindowMode	E_ALY_WindowMode [▶ 357]	Gibt an über welche Art von Fenster die Koeffizienten berechnet werden: *Continuous*: Alle Werte, seit Start der Analyse werden mit gleicher Gewichtung in die Analyse miteinbezogen. *SlidingWindow:* Die Berechnung wird über ein Fenster der Größe Window Size berechnet. Dabei fließen immer die aktuellen Werte in die Analyse ein und es werden mit jedem Zyklus Ausgänge aktualisiert. *FixWindow:* Die Ausgänge werden alle Window Size Zyklen aktualisiert und über ein Fenster der Länge Window Size berechnet.

Name	Typ	Beschreibung
eCorrelationMode	E_ALY_CorrelationMode [▶ 356]	Die Koeffizienten werden nach einer der folgenden Definitionen berechnet: *Base:* $$C_{xy}[m,t_n]=\begin{cases}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}, & m\geq 0\\ \sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}, & m<0\end{cases}$$ *Normed:* $$\hat{C}_{xy}[m,t_n]=\begin{cases}\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}, & m\geq 0\\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}, & m<0\end{cases}$$ *Covariance:* $$cov_{xy}[m,t_n]=\begin{cases}\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}-\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot\frac{1}{N}\sum_{i=0}^{N-1} y_{n-i}, & m\geq 0\\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}-\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot\frac{1}{N}\sum_{i=0}^{N-1} y_{n-i+m}, & m<0\end{cases}$$ *CovarianceBessel:* $$\widetilde{cov}_{xy}[m,t_n]=\frac{N}{N-1}cov_{xy}[m,t_n]$$ *Pearson:* $$\rho_{xy}[m,t_n]=\frac{cov(x,y)[m,t_n]}{\sigma_x[m,t_n]\sigma_y[m,t_n]},where\ \sigma_x^2[m,t_n]=cov_{xx}[m,t_n]$$
nWindowSize	UDINT	Gibt für die Window Mode *SlidingWindow* und *FixWindow* die Anzahl der Zyklen an, über die die Koeffizienten berechnet werden. Für den Window Mode *Continuous* hat das Setzen von nWindowSize keine Auswirkung. Im *SlidingWindow* Mode, werden zusätzlich zu Maximum Lag Werten vom Referenz Channel und Minimum Lag Werten für Channel 00 bis Channel 0n, Window Size Werte für alle Channel zwischengespeichert. Die Größe des Routerspeichers ist beim Setzen dieser Parameter zu berücksichtigen.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.2.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.2.5 GetChannelOutputArray

Abrufen eines kanalspezifischen Ausgangsarrays. Die Ausgangswerte wurden in der zuvor aufgerufenen Call()-Methode aktualisiert.

Syntax

Definition:

```
METHOD GetChannelOutputArray : BOOL
VAR_INPUT
    nChannel : UDINT;
    pCorrCoefsArrayOut : PVOID;
    nCorrCoefsArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
pCorrCoefsArrayOut	PVOID	Pointer zu einem Array, in welches die Koeffizienten geschrieben werden sollen.
nCorrCoefsArrayOutSize	UDINT	Größe des Arrays

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputArray	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.2.6 GetChannelOutputValues

Abrufen von kanalspezifischen Ausgangswerten. Die Ausgangswerte wurden in der zuvor aufgerufenen Call()-Methode aktualisiert.

Syntax

Definition:

```
METHOD GetChannelOutputValues : BOOL
VAR_INPUT
    nChannel : UDINT;
    pCorrCoefsArrayOut : PVOID;
    nCorrCoefsArrayOutSize : UDINT;
END_VAR
VAR_IN_OUT
    fMinCoef : LREAL;
    fMaxCoef : LREAL;
    nMinimizingLag : DINT;
    nMaximizingLag : DINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
fMinCoef	LREAL	Variable wird mit dem minimalen Koeffizienten des Ausgangsarrays beschrieben.
fMaxCoef	LREAL	Variable wird mit dem maximalen Koeffizienten des Ausgangsarrays beschrieben.
nMinimizingLag	DINT	Variable wird mit dem Lag beschrieben, bei dem der Koeffizient minimal ist.
nMaximizingLag	DINT	Variable wird mit dem Lag beschrieben, bei dem der Koeffizient maximal ist.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValues	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.2.7 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (0 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.3 FB_ALY_CorrelationFunctionReference

Der *Correlation Function Reference* Baustein berechnet die diskrete Korrelationsfunktion zwischen einem aufgenommenen Signal (im Folgenden Referenzsignal), welches aus einer tcab-Datei eingelesen wird, und einem oder mehreren Eingangssignalen (Channel 00, …, Channel 0n).

Die Korrelationskoeffizienten werden für Zeitverschiebungen von *m* Zyklen zwischen den beiden Signalen berechnet, wobei der maximale und der minimale Wert für *m* durch die Parameter *Minimum Lag* (negative ganze Zahl) und *Maximum Lag* (positive ganze Zahl) begrenzt ist.

Durch den Parameter *Step Size* wird bestimmt, um wie viele Zyklen die Signale für die Berechnung zweier aufeinanderfolgender Korrelationskoeffizienten verschoben werden. D.h. *m* ist immer ein Vielfaches der *Step Size*. Dementsprechend sind für *Minimum Lag* und *Maximum Lag* auch nur Vielfache der *Step Size* zulässig.

Ist die *Step Size* auf eins gesetzt und beispielsweise *Minimum Lag* auf -6 sowie *Maximum Lag* auf +4, so werden Koeffizienten für Verschiebungen um -6, -5, -4, -3, -2, -1, 0, +1, +2, +3 und +4 Zyklen berechnet. Setzt man die *Step Size* auf zwei, werden Koeffizienten für Verschiebungen um -6, -4, -2, 0, +2 und +4 Zyklen berechnet.

Ab dem Start der Analyse wird von dem eingelesenen Signal pro Zyklus ein Wert verarbeitet. Ist das Ende der Datei erreicht, wird wieder mit dem ersten Wert der Datei begonnen. Das eingelesene Signal wird also als periodisch angenommen. Möchte man nur bestimmte Zeitabschnitte des Eingangssignals mit dem Referenzsignal korrelieren, kann man dies über die Eingänge *Enable Execution* und *Reset* sowie über den Ausgang *New Result* steuern.

Die Korrelationskoeffizienten können über verschiedene Zeitfenster berechnet werden. Diese werden durch den Parameter *Window Mode* gesetzt. Im *Continuous* Modus werden alle Werte seit Beginn der Analyse in die Berechnungen miteinbezogen. Im *SlidingWindow* Modus läuft die Berechnung kontinuierlich über die letzten, durch die *Window Size* gesetzte Anzahl von Zyklen. Im *FixWindow* Modus erfolgt die Berechnung über die Länge des Referenzsignals und die Ausgangswerte werden immer zum Ende der aufgenommenen Sequenz aktualisiert.

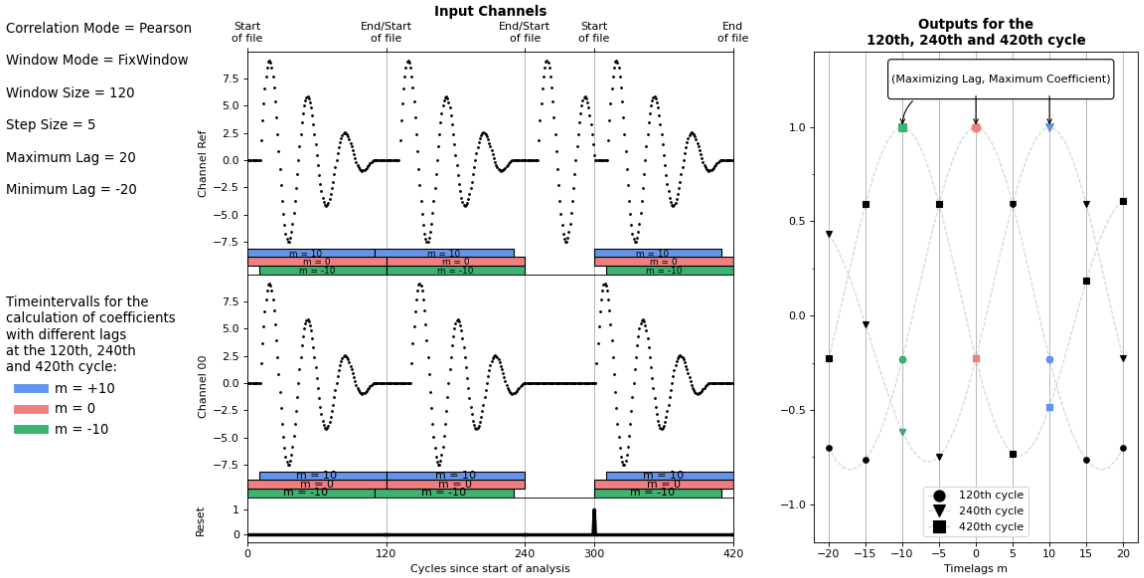
Für *m* ungleich Null verschiebt sich das Fenster für das entsprechende Signal um *m* Zyklen. Damit bleibt die Anzahl von Werten, welche in die Berechnung der Koeffizienten mit einbezogen werden, für alle Werte von *m* gleich. Eine Ausnahme hierzu bilden nur die Ergebnisse aus den ersten Zyklen nach Start der Analyse. Ist die Anzahl der verstrichenen Zyklen kleiner als |*m*|, werden entsprechend weniger Werte in die Berechnung mit einbezogen.

Für positive Werte von *m* werden die Werte des Referenzsignals (Channel Ref) in einem Ringspeicher gespeichert, damit jeweils der vor *m* Zyklen eingegangene Wert des Referenzsignals mit den aktuellen Werten von Channel 00 bis Channel 0n verglichen werden kann. Dies entspricht einer Verschiebung des Referenzsignals in die Vergangenheit. Für negative Werte von *m* müsste das Referenzsignal dementsprechend in die Zukunft verschoben werden. Da dies offensichtlich nicht möglich ist, wird stattdessen das zweite Signal (Channel 00,..Channel0n) gespeichert und nach hinten verschoben.

Die Korrelationskoeffizienten können nach unterschiedlichen Berechnungsvorschriften berechnet werden. Diese wird durch den *Correlation Mode* bestimmt. In den Modi *Base* und *Normed* werden die Koeffizienten analog zu der Definition aus der Signalverarbeitung berechnet, welche die Korrelation über die Faltung berechnet. Im *Normed* – Modus werden die Koeffizienten zudem durch die Anzahl der Summanden geteilt. *Covariance* und *CovarianceBessel* berechnen die Kovarianz ohne und mit Bessel-Korrektur. Der Modus *Pearson* nutzt die Definition des in der Statistik üblichen Pearson Korrelationskoeffizienten. In den Konfigurationsoptionen sind die genauen Berechnungsvorschriften mathematisch aufgeführt. Hier bezeichnet x_n den Wert des Referenzsignals und y_n den Wert des Eingangssignals (jeweils Channel00, …, Channel0n) zum Zeitstempel t_n (entsprechend dem n-ten Zyklus seit Start der Analyse oder seit Reset, ausgenommen der Zyklen in denen *Enable Execution* = FALSE). Der Wert von *N* hängt von dem gewählten *WindowMode* ab. Für den *SlidingWindow*-Mode und *FixWindow*-Mode ist *N* gleich der *Window Size*, vorausgesetzt seit Start der Analyse sind bereits entsprechend viele Zyklen vergangen, sodass x_{n-N-m} (bzw. y_{n-N+m}) aufgenommen wurde, andernfalls reduziert sich *N* zu *n-m+1* bzw. *n+m+1*. Im FixWindow-Mode ist die Window Size nicht manuell zu setzen, sondern entspricht der Länge des eingelesenen Signalabschnitts (Referenzsignal). und die Ausgangswerte werden zum Ende des Signalabschnitts aktualisiert. Im *Continuous*-Mode gilt stets *N* = *n+1*.

In der Abbildung sind beispielhaft für eine Konfiguration (*Correlation Mode* = *Pearson*, *Window Mode* = *FixWindow*, *Window Size* = 120 (= Anzahl an Werten in der Datei), *Step Size* = 5, *Maximum Lag* = 20, *Minimum Lag* = -20) und einen Eingangskanal die verschiedenen Eingangs- und Ausgangswerte des Bausteins dargestellt. Auf der linken Seite sind in den beiden unteren Plots die Eingangssignale Channel 00 und *Reset* abgebildet, darüber ist auf der gleichen Zeitachse die eingelesene Sequenz, gemäß ihrer Verarbeitung, abgebildet. Zu Beginn der Analyse wird begonnen das Signal einzulesen. Ist der letzte Wert aus der Datei im 120ten (bzw. 240ten) Zyklus verarbeitet, wird erneut mit dem ersten begonnen. Im 300ten Zyklus wird ein *Reset* durchgeführt und es wird erneut mit dem ersten Wert der Datei begonnen. Zudem werden alle Werte aus den internen Speichern gelöscht und die Berechnung der Koeffizienten beginnt erneut. Beispielsweise wurde hier erkannt, dass Channel 00 in den vorherigen Zyklen keine gültigen Werte enthielt und jetzt die Schwingung erneut angeregt wurde. In diesem Bereich könnte auch durch *Enable Execution* = FALSE die Analyse unterbrochen werden. Da *WindowMode* = *FixWindow*, werden die Ausgänge nur im 120ten, 240ten und 420ten Zyklus aktualisiert. Die entsprechenden Koeffizienten (von Ausgang Output00) sind im rechten Plot dargestellt. An den Wertepaaren (*Maximizing Lag*, *Maximum Coefficient*) kann man jeweils ablesen, um wie viel das Eingangssignal zum Referenzsignal verschoben ist.

Beispielsweise ist im 420ten Zyklus (*Maximizing Lag*, *Maximum Coefficient*) = (-10,1). Das heißt, hätte der *Reset* 10 Zyklen früher stattgefunden, hätten das Referenzsignal und Channel 00 in diesem Fenster exakt übereinander gepasst.



Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_CorrelationFunctionReference
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bBusy: BOOL;
    eState: E_ALY_ReadState;
    fValueRead: LREAL;
    stFileHeader: ST_ALY_FileHeader;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
bBusy	BOOL	`TRUE`, wenn der FB aufgrund von Dateizugriffen aktiv ist.
eState	E_ALY_ReadState [▶ 355]	Aktueller Status des FB aufgrund des asynchronen Dateizugriffs.
fValueRead	LREAL	Wert des aus einer Datei gelesenen Datenpunkts.
stFileHeader	ST_ALY_FileHeader	Header-Informationen der gelesenen Datei.

Methoden

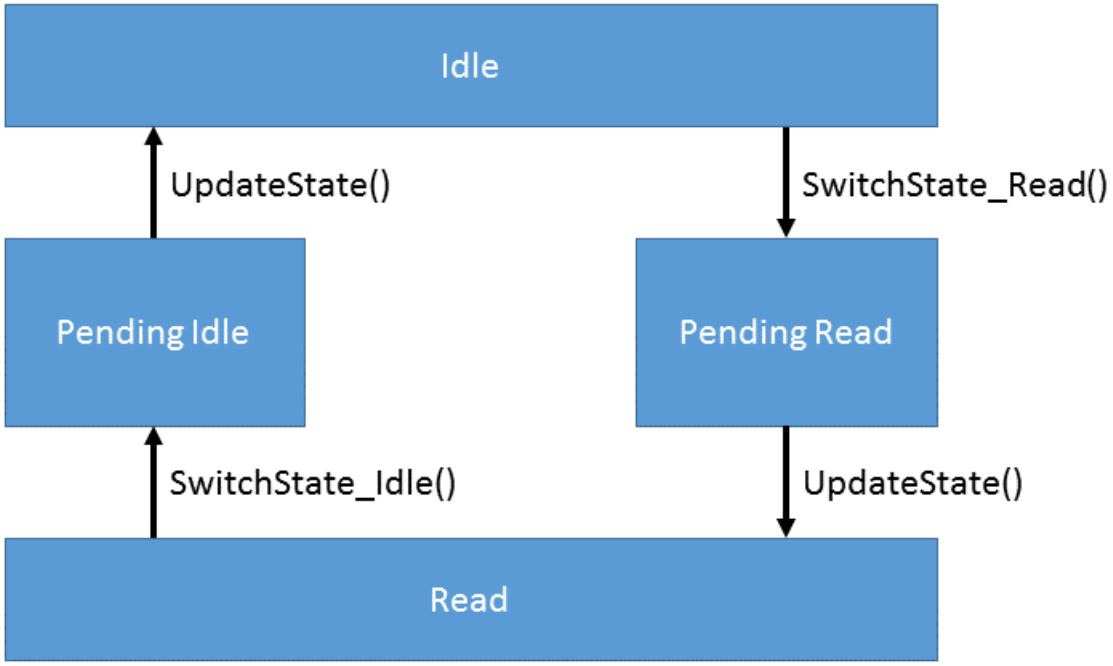
Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputArray()	Local	Abrufen des Ergebnis-Arrays für einen bestimmten Kanal, ohne neue Werte hinzuzufügen.
GetChannelOutputValues()	Local	Abrufen der Ergebnisse (Einzelwerte) für einen bestimmten Kanal, ohne neue Werte hinzuzufügen.
SwitchState_Idle()	Local	Initiieren des Wechsels vom Zustand Read in den Zustand Idle. Siehe Zustandsdiagramm.
SwitchState_Read()	Local	Initiieren des Wechsels vom Zustand Idle in den Zustand Read. Siehe Zustandsdiagramm.
UpdateState()	Local	Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.
GetBusyState()	Local	Gibt TRUE zurück, wenn der FB aktiv ist.

Zustandsdiagramm

Aufgrund des asynchronen Dateizugriffs in Echtzeitanwendungen benötigt dieser Funktionsbaustein eine Zustandsmaschine, um den Dateizugriff vorzubereiten und abzuschließen.

Beim Start ist der Funktionsbaustein im Zustand *Idle*. Um die Daten aus der Datei zu lesen, muss er in den Zustand *Read* wechseln. Daher muss die Methode *SwitchState_Read()* einmal aufgerufen werden, um den Funktionsbaustein in den Zustand *PendingRead* zu versetzen. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Read* befindet. In diesem Zustand kann der Baustein ein oder mehrere Zyklen durchlaufen. Wenn der Funktionsbaustein keine weiteren Zyklen vergleichen soll, kann er wieder in den Zustand *Idle* versetzt werden. Um den Zustandswechsel zu initiieren, muss die Methode *SwitchState_Idle()* aufgerufen werden. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Idle* befindet.

Zustandsdiagramm für den Lesevorgang der Daten:



Beispiel

```
VAR
    fbCorrFctRef : FB_ALY_CorrelationFunctionReference(
        nNumChannels := 1,
        nMinLag := 0,
        nMaxLag := 10,
        nStepSize := 1,
        eWindowMode := E_ALY_WindowMode.FixWindow,
        eCorrelationMode := E_ALY_CorrelationMode.Normed,
        nWindowSize := 120,
        nSegmentSize := 400,
        tTimeout := TIME#5S,
        sFilePath := 'C:\TwinCAT\3.1\Boot\UnitTest_CorrelationFunction.tas');

    // State
    eState : E_ALY_ReadState := E_ALY_ReadState.Idle;
    bRead : BOOL;

    // Inputs
    fInputCh1 : LREAL;

    // Results
    fMinCoef : LREAL;
    fMaxCoef : LREAL;
    nMinimizingLag : DINT;
    nMaximizingLag : DINT;
    aCoefficients : ARRAY[-10..10] OF LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbCorrFctRef.Configure(eCorrelationMode := eCorrelationMode, sFilePath := sFilePath );
END_IF

// Call algorithm
eState := fbCorrFctRef.eState;
CASE eState OF
E_ALY_ReadState.Idle:
    IF bRead THEN
        fbCorrFctRef.SwitchState_Read();
        fbCorrFctRef.UpdateState();
    END_IF
E_ALY_ReadState.Read:
    fbCorrFctRef.SetChannelValue(1, fInputCh1);
    fbCorrFctRef.Call();
    fbCorrFctRef.GetChannelOutputArray(1, ADR(aCoefficients), SIZEOF(aCoefficients));
```

```
    fbCorrFctRef.GetChannelOutputValues(1, fMinCoef, fMaxCoef, nMinimizingLag, nMaximizingLag);
    IF NOT bRead THEN
        fbCorrFctRef.SwitchState_Idle();
        fbCorrFctRef.UpdateState();
    END_IF
E_ALY_ReadState.Pending,
E_ALY_ReadState.PendingIdle,
E_ALY_ReadState.PendingRead:
    fbCorrFctRef.UpdateState();
    eState := fbCorrFctRef.eState;
END_CASE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.3.1 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (0 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.3.2 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 2;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt.

5.1.1.6.3.3 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.3.4 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nMinLag : DINT;
    nMaxLag : DINT;
    nStepSize : UDINT;
    nNumChannels : UDINT;
    eWindowMode : E_ALY_WindowMode;
    eCorrelationMode : E_ALY_CorrelationMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nMinLag	UDINT	Gibt die minimale Anzahl an, um die die beiden Signale für die Berechnung der Korrelation zueinander verschoben werden. Dies ist eine negative ganze Zahl.
nMaxLag	UDINT	Gibt die maximale Anzahl an Zyklen an, um die die beiden Signale für die Berechnung der Korrelation zueinander verschoben werden. Dies ist eine positive ganze Zahl.
nStepSize	UDINT	Gibt an, um wie viele Zyklen die Signale für die Berechnung zweier aufeinanderfolgender Korrelationskoeffizienten verschoben werden.
nNumChannels	UDINT	Die Anzahl der Kanäle, die mit dem Referenzsignal korreliert werden.
eWindowMode	E_ALY_WindowMode [▶ 357]	Gibt an über welche Art von Fenster die Koeffizienten berechnet werden: *Continuous*: Alle Werte, seit Start der Analyse werden mit gleicher Gewichtung in die Analyse miteinbezogen. *SlidingWindow:* Die Berechnung wird über ein Fenster der Größe Window Size berechnet. Dabei fließen immer die aktuellen Werte in die Analyse ein und es werden mit jedem Zyklus Ausgänge aktualisiert. *FixWindow:* Die Ausgänge werden alle Window Size Zyklen aktualisiert und über ein Fenster der Länge Window Size berechnet.

Name	Typ	Beschreibung
eCorrelationMode	E_ALY_CorrelationMode [▶ 356]	Die Koeffizienten werden nach einer der folgenden Definitionen berechnet: *Base:* $$C_{xy}[m,t_n]=\begin{cases}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}, & m\geq 0\\ \sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}, & m<0\end{cases}$$ *Normed:* $$\hat{C}_{xy}[m,t_n]=\begin{cases}\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}, & m\geq 0\\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}, & m<0\end{cases}$$ *Covariance:* $$cov_{xy}[m,t_n]=\begin{cases}\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot y_{n-i}-\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i-m}\cdot\frac{1}{N}\sum_{i=0}^{N-1} y_{n-i}, & m\geq 0\\ \frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot y_{n-i+m}-\frac{1}{N}\sum_{i=0}^{N-1} x_{n-i}\cdot\frac{1}{N}\sum_{i=0}^{N-1} y_{n-i+m}, & m<0\end{cases}$$ *CovarianceBessel:* $$\widetilde{cov}_{xy}[m,t_n]=\frac{N}{N-1}cov_{xy}[m,t_n]$$ *Pearson:* $$\rho_{xy}[m,t_n]=\frac{cov(x,y)[m,t_n]}{\sigma_x[m,t_n]\sigma_y[m,t_n]},where\ \sigma_x^2[m,t_n]=cov_{xx}[m,t_n]$$
nWindowSize	UDINT	Gibt für die Window Mode *SlidingWindow* und *FixWindow* die Anzahl der Zyklen an, über die die Koeffizienten berechnet werden. Für den Window Mode *Continuous* hat das Setzen von nWindowSize keine Auswirkung. Im *SlidingWindow* Mode, werden zusätzlich zu Maximum Lag Werten vom Referenz Channel und Minimum Lag Werten für Channel 00 bis Channel 0n, Window Size Werte für alle Channel zwischengespeichert. Die Größe des Routerspeichers ist beim Setzen dieser Parameter zu berücksichtigen.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.3.5 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.3.6 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt TRUE zurück, wenn der FB aktiv ist.

5.1.1.6.3.7 GetChannelOutputArray

Abrufen eines kanalspezifischen Ausgangsarrays. Die Ausgangswerte wurden in der zuvor aufgerufenen Call()-Methode aktualisiert.

Syntax

Definition:

```
METHOD GetChannelOutputArray : BOOL
VAR_INPUT
    nChannel : UDINT;
    pCorrCoefsArrayOut : PVOID;
    nCorrCoefsArrayOutSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
pCorrCoefsArrayOut	PVOID	Pointer zu einem Array, in welches die Koeffizienten geschrieben werden sollen.
nCorrCoefsArrayOutSize	UDINT	Größe des Arrays

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputArray	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.3.8 GetChannelOutputValues

Abrufen von kanalspezifischen Ausgangswerten. Die Ausgangswerte wurden in der zuvor aufgerufenen Call()-Methode aktualisiert.

Syntax

Definition:

```
METHOD GetChannelOutputValues : BOOL
VAR_INPUT
    nChannel : UDINT;
    pCorrCoefsArrayOut : PVOID;
    nCorrCoefsArrayOutSize : UDINT;
END_VAR
VAR_IN_OUT
    fMinCoef : LREAL;
    fMaxCoef : LREAL;
    nMinimizingLag : DINT;
    nMaximizingLag : DINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
fMinCoef	LREAL	Variable wird mit dem minimalen Koeffizienten des Ausgangsarrays beschrieben.
fMaxCoef	LREAL	Variable wird mit dem maximalen Koeffizienten des Ausgangsarrays beschrieben.
nMinimizingLag	DINT	Variable wird mit dem Lag beschrieben, bei dem der Koeffizient minimal ist.
nMaximizingLag	DINT	Variable wird mit dem Lag beschrieben, bei dem der Koeffizient maximal ist.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValues	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.3.9 SwitchState_Idle

Initiieren des Wechsels vom Zustand *Read* in den Zustand *Idle*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Idle : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Idle	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.6.3.10 SwitchState_Read

Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Read*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Read : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Read	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.6.3.11 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.4 FB_ALY_LinearRegressionFitting

Der Baustein Linear Regression Fitting nähert eine Variable (den Input Dependent) durch die Linearkombination mehrerer anderer Variablen (Input 01 … Input 0n) an. Dies geschieht durch das inkrementelle stochastische Gradientenverfahren. Zum Ende der Analyse werden die berechneten Koeffizienten in eine Datei geschrieben.

Die Linearkombination ist gegeben durch folgende Gleichung:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i \times \text{Input 0i}$$

In jedem Zyklus werden die Werte für β_0 bis β_n mit folgender Vorschrift neu berechnet:

$$\beta_i = \begin{cases} \beta_i - \gamma \times \text{Input 0i} \times (y - \text{Dependent}), & i = 1, 2, \ldots, n \\ \beta_i - \gamma \times (y - \text{Dependent}), & i = 0 \end{cases}$$

Dies entspricht der Minimierung der quadratischen Abweichung der berechneten Werte y (vom Baustein als

Result ausgegeben) von dem dazugehörigen Eingangswert Dependent. Der Parameter γ entspricht der Step Size und gibt an, wie stark die Parameter angeglichen werden. Je größer der Wert, desto schneller nähern sich die Koeffizienten einem lokalen Optimum an. Wird der Wert jedoch zu groß gewählt, kann es sein, dass der Algorithmus nicht konvergiert.

Typischerweise werden zunächst mit dem Baustein Linear Regression Fitting die Gewichte für die Regression einer Zielvariable angepasst. Anschließend kann mithilfe des Bausteins Linear Regression Inference und den angepassten Gewichten die Zielvariable anhand der Eingangsvariablen vorhergesagt werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_LinearRegressionFitting
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fResult: LREAL;
    fMSE: LREAL;
    bBusy: BOOL;
    eState: E_TeachState;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fResult	LREAL	Gibt den angenäherten Wert für die Inputs des aktuellen Zyklus mit den daraus aktualisierten Koeffizienten aus.
fMSE	LREAL	Gibt den MSE (mean squared error) zwischen errechnetem Wert Result und Eingangswert Dependent an.
bBusy	BOOL	`TRUE`, wenn der FB aufgrund eines Dateizugriffs aktiv ist.
eState	E_ALY_TeachState [▶ 356]	Aktueller Zustand des Funktionsbausteins. Siehe Zustandsdiagramm.

Methoden

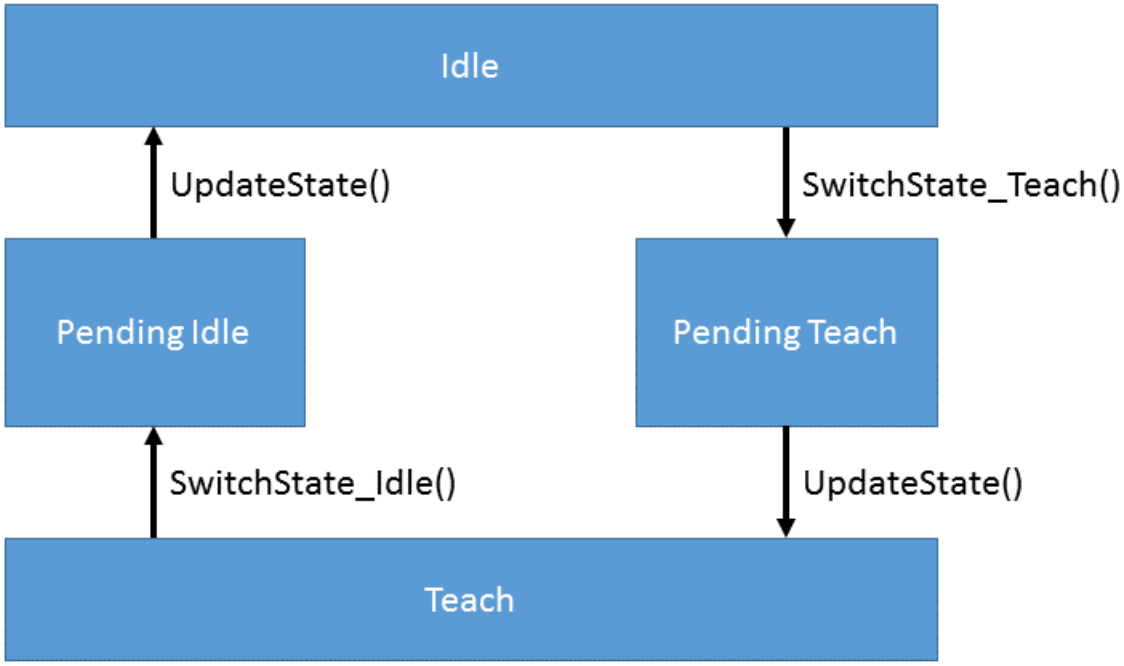
Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
GetBusyState()	Local	Diese Methode liefert den Zustand Busy des Funktionsbausteins.
GetChannelOutputValue()	Local	Methode für den Empfang von Werten von verschiedenen Ausgangskanälen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SwitchState_Idle()	Local	Methode, um den Funktionsbaustein in den Zustand Idle zu versetzen, wenn der Einlernvorgang abgeschlossen ist.
SwitchState_Teach()	Local	Methode, um den Funktionsbaustein in den Einlernmodus zu versetzen.
UpdateState()	Local	Methode zur Verwendung für den Übergang von einem Zustand in einen anderen Zustand.

Zustandsdiagramm

Aufgrund des asynchronen Dateizugriffs in Echtzeitanwendungen benötigt dieser Funktionsbaustein eine Zustandsmaschine, um den Dateizugriff vorzubereiten und abzuschließen.

Beim Start ist der Funktionsbaustein im Zustand *Idle*. Um die eingehenden Daten in eine Datei zu schreiben, muss er in den Zustand *Teach* wechseln. Daher muss die Methode *SwitchState_Teach()* einmal aufgerufen werden, um den Funktionsbaustein in den Zustand *PendingTeach* zu versetzen. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Teach* befindet. In diesem Zustand können ein oder mehrere Einlernzyklen durchlaufen werden. Wenn der Funktionsbaustein keine weiteren Zyklen einlernen soll, kann er wieder in den Zustand *Idle* versetzt werden. Um den Zustandswechsel zu initiieren, muss die Methode *SwitchState_Idle()* aufgerufen werden. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Idle* befindet.

Zustandsdiagramm für den Einlernvorgang der Daten:



Beispiel

```
VAR
    fbLinearRegressionFitting : FB_ALY_LinearRegressionFitting(nNumChannels := 1);
    bBias : BOOL := TRUE;
    fStepSize : LREAL := 0.01;
    nMiniBatchSize : UDINT := 1;
    tTimeout : TIME := T#5S;
    bInvolveExistingFile : BOOL := TRUE;
    sFilePath : STRING := 'C:\TwinCAT\3.1\Boot\LinearRegressionFitting.tas';
    bConfigure : BOOL := TRUE;
    eState : E_ALY_TeachState := E_ALY_TeachState.Idle;
    bTeach : BOOL;
    bFit : BOOL := TRUE;
    fInputX : LREAL;
    fInputY : LREAL;
    fOut0 : LREAL;
    fOut1 : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbLinearRegressionFitting.Configure(bBias, fStepSize, nMiniBatchSize, tTimeout, bInvolveExisting
File, sFilePath);
END_IF

// Call algorithm
eState := fbLinearRegressionFitting.eState;
CASE eState OF
E_ALY_TeachState.Idle:
    IF bTeach THEN
        fbLinearRegressionFitting.SwitchState_Teach();
        fbLinearRegressionFitting.UpdateState();
    END_IF
E_ALY_TeachState.Teach:
    fbLinearRegressionFitting.SetChannelValue(1, fInputX);
    fbLinearRegressionFitting.Call(bFit := bFit, dependent := fInputY);
    fbLinearRegressionFitting.GetChannelOutputValue(0, fOut0);
    fbLinearRegressionFitting.GetChannelOutputValue(1, fOut1);
    IF NOT bTeach THEN
        fbLinearRegressionFitting.SwitchState_Idle();
        fbLinearRegressionFitting.UpdateState();
    END_IF
E_ALY_TeachState.Pending,
E_ALY_TeachState.PendingIdle,
E_ALY_TeachState.PendingTeach:
```

```
    fbLinearRegressionFitting.UpdateState();
    eState := fbLinearRegressionFitting.eState;
END_CASE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.4.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.4.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    bFit : BOOL;
    dependent : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bFit	BOOL	Die Parameteranpassung kann aktiviert oder deaktiviert werden.
dependent	ANY	Abhängiger Wert, der durch die Linearkombination der Eingangswerte entsteht.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.4.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bBias : BOOL;
    fStepSize : LREAL;
    nMiniBatchSize : UDINT
    tTimeout : TIME;
    nSetNumTeaches : UDINT;
    bInvolveExistingFile : BOOL;
    sFilePath : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bBias	BOOL	Wenn FALSE, wird der Bias Output 00 auf null gesetzt und nicht weiter angenähert.
fStepSize	LREAL	Gibt an, wie stark die Koeffizienten nach jeder neuen Berechnung angepasst werden.
nMiniBatchsize	UDINT	Gibt an über wie viele Zyklen der MSE berechnet werden soll, bevor die Koeffizienten anhand diesem angepasst werden.
tTimeout	TIME	Zeitüberschreitung für asynchrone Operationen.
bInvolveExistingFile	BOOL	Wenn TRUE, vorhandene Datei einbeziehen (falls vorhanden). Wenn FALSE, neue Datei erstellen.
sFilePath	STRING(255)	Pfad zur eingelernten Datei, z. B. *C:\TwinCAT\3.1\Boot\Teach.tas*

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.4.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.4.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.4.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (0 (Bias) bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.4.7 SwitchState_Idle

Initiieren des Wechsels vom Zustand *Teach* in den Zustand *Idle*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Idle : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Idle	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.6.4.8 SwitchState_Teach

Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Teach*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Teach : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Teach	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.6.4.9 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt TRUE zurück, wenn der FB aktiv ist.

5.1.1.6.4.10 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.6.5 FB_ALY_LinearRegressionInference

Der Baustein Linear Regression Inference berechnet die Linearkombination der Inputs (Input 01 .. Input 0n) mit den Koeffizienten (Weights 00.. Weights 0n).

$$\text{Result} = \text{Weights 00} + \sum_{i=1}^{n} \text{Weights 0i} \times \text{Input 0i}$$

Die Parameter Weights 00 bis Weights 0n können entweder manuell gesetzt werden, oder über eine Datei, wie sie vom Baustein Linear Regression Fitting erzeugt wird über Drag / Drop auf das Parameterfeld automatisch gesetzt werden. Typischerweise werden zunächst mit dem Baustein Linear Regression Fitting die Gewichte für die Regression einer Zielvariable angepasst. Anschließend kann mithilfe des Bausteins Linear Regression Inference und den angepassten Gewichten die Zielvariable anhand der Eingangsvariablen vorhergesagt werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_LinearRegressionInference
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bBusy: BOOL;
    eState: E_ALY_ConfigState;
    stFileHeader: ST_ALY_FileHeader;
    fResult : LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bBusy	BOOL	TRUE, wenn der FB aufgrund eines Dateizugriffs aktiv ist.
eState	E_ALY_ConfigState [▶ 357]	Konfigurationsstatus des Funktionsbausteins.
stFileHeader	ST_ALY_FileHeader	Header-Informationen der gelesenen Datei.
fResult	LREAL	Gibt den aus der Linearkombination berechneten Wert an.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus nachdem die einzelnen Koeffizienten gesetzt wurden.
ConfigureChannel()	Local	Konfiguration der Koeffizienten.
ConfigureFromFile()	Local	Konfiguration des Algorithmus über eine zuvor erstellte Datei.
FB_init	Local	Initialisieren der Anzahl der Eingangskanäle.
GetBusyState()	Local	Diese Methode liefert den Zustand Busy des Funktionsbausteins.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
UpdateState()	Local	Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Beispiel – Konfiguration durch Parameter

```
VAR
    fbLinearRegressionInference : FB_ALY_LinearRegressionInference(nNumChannels := 1);
    fCoefficient0 : LREAL := 10.0;
    fCoefficient1 : LREAL := 1.0;
    bConfigure : BOOL := TRUE;
    fInputX : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbLinearRegressionInference.ConfigureChannel(0, fCoefficient0);
    fbLinearRegressionInference.ConfigureChannel(1, fCoefficient1);
    fbLinearRegressionInference.Configure();
END_IF

// Call algorithm
fbLinearRegressionInference.SetChannelValue(1, fInputX);
fbLinearRegressionInference.Call();
END_CASE
```

Beispiel – Konfiguration durch Datei

```
VAR
    fbLinearRegressionInference : FB_ALY_LinearRegressionInference(nNumChannels := 1);
    tTimeout : TIME := T#5S;
    sFilePath : STRING := 'C:\TwinCAT\3.1\Boot\LinearRegressionFitting.tas';
    bConfigure : BOOL := TRUE;
    fInputX : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbLinearRegressionInference.ConfigureFromFile(tTimeout, sFilePath);
END_IF

// Update pending state
IF fbLinearRegressionInference.eState = E_ALY_ConfigState.Pending THEN
    fbLinearRegressionInference.UpdateState();
END_IF

// Call algorithm
IF fbLinearRegressionInference.eState = E_ALY_ConfigState.Configured THEN
    fbLinearRegressionInference.SetChannelValue(1, fInputX);
    fbLinearRegressionInference.Call();
END_IF
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.5.1 FB_init

Initialisieren der Anzahl der Eingangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.5.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.5.3 Configure

Konfigurieren des Algorithmus. Die Koeffizienten müssen zuvor mit der Methode ConfigureChannel() gesetzt werden.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.5.4 ConfigureChannel

Konfiguration der Koeffizienten für die Berechnung der Linearkombination. Die Koeffizienten werden erst nach dem Aufruf der Methode Configure() übernommen.

Syntax

Definition:

```
METHOD ConfigureChannel : BOOL
VAR_INPUT
    nChannel : UDINT;
    fCoefficient : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex für die Koeffizienten. Beginnend bei null.
fCoefficient	LREAL	Koeffizient für die Berechnung der Linearkombination.

Rückgabewert

Name	Typ	Beschreibung
ConfigureChannel	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.5.5 ConfigureFromFile

Konfigurieren des Algorithmus auf Basis einer zuvor erstellen Datei. Nach Aufruf dieser Methode muss die Methode UpdateState() solange aufgerufen werden, bis der Status des Funktionsbausteins ungleich Pending ist.

Syntax

Definition:

```
METHOD ConfigureFromFile : BOOL
VAR_INPUT
    tTimeout : TIME;
    sFilePath : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimeout	TIME	Zeitüberschreitung für asynchrone Operationen.
sFilePath	STRING(255)	Pfad zur eingelernten Datei, z. B. *C:\TwinCAT\3.1\Boot\Teach.tas*

Rückgabewert

Name	Typ	Beschreibung
ConfigureFromFile	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.5.6 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung durch den Aufruf der Methode ConfigureFromFile() initiiert wurde. Die Methode muss so lange zyklisch aufgerufen werden, bis der Zustand des Funktionsbausteins ungleich Pending ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt TRUE zurück, wenn erfolgreich

5.1.1.6.5.7 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.5.8 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt TRUE zurück, wenn der FB aktiv ist.

5.1.1.6.5.9 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.6 FB_ALY_StandardDeviation

Der Algorithmus *Standard Deviation* berechnet die empirische Standardabweichung für eine konfigurierbare Anzahl von Eingangskanälen. Die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden und die Art der Berechnung können konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StandardDeviation
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen. Die Konfiguration ist für alle Kanäle identisch.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputValue()	Local	Methode für den Empfang von Werten von verschiedenen Ausgangskanälen.

Beispiel

```
VAR
    fbStandardDeviation : FB_ALY_StandardDeviation(nNumChannels := 1);
    bUseBesselCorrection : BOOL :=  TRUE;
```

```
    eWindowMode : E_ALY_WindowMode := E_ALY_WindowMode.FixWindow;
    nWindowSize : UDINT := 100;
    bConfigure : BOOL := TRUE;
    fInput : LREAL;
    fStandardDeviation : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbStandardDeviation.Configure(bUseBesselCorrection, eWindowMode, nWindowSize);
END_IF

// Call algorithm
fbStandardDeviation.SetChannelValue(1, fInput);
fbStandardDeviation.Call();
fbStandardDeviation.GetChannelOutputValue(1, fStandardDeviation);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.6.1 FB_init

Initialisieren der Anzahl unabhängiger Ein- und Ausgangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.6.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.6.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    bUseBesselCorrection : BOOL;
    eWindowMode : E_ALY_WindowMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bUseBesselCorrection	BOOL	Wenn der Parameter bUseBesselCorrection auf *TRUE* gesetzt ist, wird die Bessel'sche Korrektur angewendet. Um bei Stichproben ein erwartungstreues Ergebnis zu erhalten, ist dieser Parameter zu aktivieren. Die empirische Standardabweichung, ohne Bessel'sche Korrektur $s = \sqrt{\frac{1}{N-1}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$ Die empirische Standardabweichung, mit Bessel'scher Korrektur $s' = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(x[n]-\overline{x})^2}$
eWindowMode	E_ALY_WindowMode [▶ 357]	Verwendeter Fenstermodus. Beeinflusst die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden sowie den Zeitpunkt der Berechnungen. • *Continuous*: Alle Eingangswerte seit dem Start des Algorithmus werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch. • *Fix Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt alle *N* Aufrufe. • *Sliding Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch.
nWindowSize	UDINT	Abhängig von dem verwendeten Fenstermodus kann die Anzahl der Werte *N* konfiguriert werden, die mit in die Berechnung einbezogen werden. Beim Fenstermodus *Continuous* wird dieser Parameter ignoriert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.6.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.6.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.6.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.7 FB_ALY_ClearanceFactor

Der Algorithmus *Clearance Factor* berechnet das gleichnamige Signal-Feature aus den Eingangswerten. Der Ausgangswert berechnet sich aus dem Verhältnis des Spitzenwertes des Eingangssignals zu dem quadrierten Mittelwert der Quadratwurzeln des absoluten Eingangssignals.

$$\text{Clearance Factor} = \frac{\max(|x|)}{\left(\frac{1}{N}\sum_{n=1}^{N}\sqrt{|x[n]|}\right)^2}$$

Die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden und die Art der Berechnung können konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ClearanceFactor
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen. Die Konfiguration ist für alle Kanäle identisch.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputValue()	Local	Methode für den Empfang von Werten von verschiedenen Ausgangskanälen.

Beispiel

```
VAR
    fbClearanceFactor : FB_ALY_ClearanceFactor(nNumChannels := 1);
    eWindowMode : E_ALY_WindowMode := E_ALY_WindowMode.FixWindow;
    nWindowSize : UDINT := 100;
    bConfigure : BOOL := TRUE;
    fInput : LREAL;
    fClearanceFactor : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbClearanceFactor.Configure(eWindowMode, nWindowSize);
END_IF

// Call algorithm
fbClearanceFactor.SetChannelValue(1, fInput);
fbClearanceFactor.Call();
fbClearanceFactor.GetChannelOutputValue(1, fClearanceFactor);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.7.1 FB_init

Initialisieren der Anzahl unabhängiger Ein- und Ausgangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.7.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.7.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eWindowMode : E_ALY_WindowMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eWindowMode	E_ALY_WindowMode [▶ 357]	Verwendeter Fenstermodus. Beeinflusst die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden sowie den Zeitpunkt der Berechnungen. • *Continuous*: Alle Eingangswerte seit dem Start des Algorithmus werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch. • *Fix Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt alle *N* Aufrufe. • *Sliding Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch.
nWindowSize	UDINT	Abhängig von dem verwendeten Fenstermodus kann die Anzahl der Werte *N* konfiguriert werden, die mit in die Berechnung einbezogen werden. Beim Fenstermodus *Continuous* wird dieser Parameter ignoriert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.7.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.7.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.7.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.8 FB_ALY_ImpulseFactor

Der Algorithmus *Impulse Factor* berechnet das gleichnamige Signal-Feature aus den Eingangswerten. Der Ausgangswert berechnet sich aus dem Verhältnis des Spitzenwertes des Eingangssignals zu dem Mittelwert des Eingangssignals.

$$\text{Impulse Factor} = \frac{\max(|x|)}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

Die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden und die Art der Berechnung können konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ImpulseFactor
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen. Die Konfiguration ist für alle Kanäle identisch.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputValue()	Local	Methode für den Empfang von Werten von verschiedenen Ausgangskanälen.

Beispiel

```
VAR
    fbImpulseFactor : FB_ALY_ImpulseFactor(nNumChannels := 1);
    eWindowMode : E_ALY_WindowMode := E_ALY_WindowMode.FixWindow;
    nWindowSize : UDINT := 100;
    bConfigure : BOOL := TRUE;
```

```
    fInput : LREAL;
    fImpulseFactor : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbImpulseFactor.Configure(eWindowMode, nWindowSize);
END_IF

// Call algorithm
fbImpulseFactor.SetChannelValue(1, fInput);
fbImpulseFactor.Call();
fbImpulseFactor.GetChannelOutputValue(1, fImpulseFactor);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.8.1 FB_init

Initialisieren der Anzahl unabhängiger Ein- und Ausgangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.8.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.8.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eWindowMode : E_ALY_WindowMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eWindowMode	E_ALY_WindowMode [▶ 357]	Verwendeter Fenstermodus. Beeinflusst die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden sowie den Zeitpunkt der Berechnungen. • *Continuous*: Alle Eingangswerte seit dem Start des Algorithmus werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch. • *Fix Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt alle *N* Aufrufe. • *Sliding Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch.
nWindowSize	UDINT	Abhängig von dem verwendeten Fenstermodus kann die Anzahl der Werte *N* konfiguriert werden, die mit in die Berechnung einbezogen werden. Beim Fenstermodus *Continuous* wird dieser Parameter ignoriert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.8.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.8.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.8.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.9 FB_ALY_ShapeFactor

Der Algorithmus *Shape Factor* berechnet das gleichnamige Signal-Feature aus den Eingangswerten. Der Ausgangswert berechnet sich aus dem Verhältnis des quadratischen Mittels (RMS) des Eingangssignals zu dem Mittelwert der Absolutwerte des Eingangssignals.

$$\text{Shape Factor} = \frac{x_{rms}}{\frac{1}{N}\sum_{n=1}^{N}|x[n]|}$$

Die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden und die Art der Berechnung können konfiguriert werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_ShapeFactor
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen. Die Konfiguration ist für alle Kanäle identisch.
FB_init()	Local	Initialisieren der Anzahl der Eingangskanäle.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
GetChannelOutputValue()	Local	Methode für den Empfang von Werten von verschiedenen Ausgangskanälen.

Beispiel

```
VAR
    fbShapeFactor : FB_ALY_ShapeFactor(nNumChannels := 1);
    eWindowMode : E_ALY_WindowMode := E_ALY_WindowMode.FixWindow;
    nWindowSize : UDINT := 100;
    bConfigure : BOOL := TRUE;
    fInput : LREAL;
    fShapeFactor : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
    fbShapeFactor.Configure(eWindowMode, nWindowSize);
END_IF

// Call algorithm
fbShapeFactor.SetChannelValue(1, fInput);
fbShapeFactor.Call();
fbShapeFactor.GetChannelOutputValue(1, fShapeFactor);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.6.9.1 FB_init

Initialisieren der Anzahl unabhängiger Ein- und Ausgangskanäle.

Syntax

Definition:

```
METHOD FB_init : BOOL
VAR_INPUT
    nNumChannels : UDINT := 1;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumChannels	UDINT	Initialisieren der Anzahl der Eingangskanäle.

Rückgabewert

Name	Typ	Beschreibung
FB_init	BOOL	Nicht benutzt

5.1.1.6.9.2 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.9.3 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eWindowMode : E_ALY_WindowMode;
    nWindowSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eWindowMode	E_ALY_WindowMode [▶ 357]	Verwendeter Fenstermodus. Beeinflusst die Anzahl der Eingangsdaten, die mit in die Berechnung einbezogenen werden sowie den Zeitpunkt der Berechnungen. • *Continuous*: Alle Eingangswerte seit dem Start des Algorithmus werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch. • *Fix Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt alle *N* Aufrufe. • *Sliding Window*: Nur die letzten *N* Eingangswerte werden mit in die Berechnung einbezogen. Die Berechnung erfolgt zyklisch.
nWindowSize	UDINT	Abhängig von dem verwendeten Fenstermodus kann die Anzahl der Werte *N* konfiguriert werden, die mit in die Berechnung einbezogen werden. Beim Fenstermodus *Continuous* wird dieser Parameter ignoriert.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.6.9.4 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.6.9.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.6.9.6 GetChannelOutputValue

Abholen eines kanalspezifischen Ausgangswerts. Der Ausgangswert wird nur aktualisiert, wenn zuvor die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD GetChannelOutputValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    output : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nNumChannels)
output	ANY	Ausgabewert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
GetChannelOutputValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.7 Traning Base

5.1.1.7.1 FB_ALY_TimeBasedTeachPath_1Ch

Der *Time Based Teach Path 1Ch* schreibt die Eingangsdaten periodisch, entsprechend der konfigurierten Anzahl von Einlernvorgängen, in eine Datei. Dadurch werden die Werte nicht für jede Periode nacheinander geschrieben, sondern die Werte einer neuen Periode mit den vorhandenen Werten verglichen. Die Periode kann durch die Eingangswerte *Start Period* und *Stop Period* festgelegt werden (boolsche Signale sind erforderlich). Gemäß dem Teach -Modus wird jeder Wert überschrieben oder beibehalten, so dass das Ergebnis ein eingelerntes Eingangssignal ist, das später beispielsweise als Referenzsignal für den Algorithmus Time Based Envelope 1Ch verwendet werden kann.

Syntax

Definition:

```
FUNCTION_BLOCK FB_TimeBasedTeachPath_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bBusy: BOOL;
    bTeaching: BOOL;
    eState: E_TeachState;
    nWrittenValues: ULINT;
    nValuesInFile: ULINT;
    nCurrentTeachCycles: ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bBusy	BOOL	TRUE, wenn der FB aufgrund eines Dateizugriffs aktiv ist.
bTeaching	BOOL	TRUE, wenn der Algorithmus das Einlernen verarbeitet, ansonsten FALSE. Der Einlernvorgang startet, wenn das Flag Start Period auf TRUE schaltet und stoppt, wenn das Flag Stop Period auf TRUE schaltet.
eState	E_ALY_TeachState [▶ 356]	Aktueller Zustand des Funktionsbausteins. Siehe Zustandsdiagramm.
nWrittenValues	ULIINT	Gesamtanzahl der geschriebenen Werte während des Einlernprozesses. Nicht zu verwechseln mit der Anzahl der Values in File, die jeden Einlernzyklus überschrieben werden.
nValuesInFile	ULINT	Anzahl der Werte, die gegenwärtig in die Datei geschrieben werden.
nCurrentTeachCycles	ULINT	Anzahl der Einlernzyklen in der Datei.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
GetBusyState()	Local	Diese Methode liefert den Zustand Busy des Funktionsbausteins.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SwitchState_Idle()	Local	Methode, um den Funktionsbaustein in den Zustand Idle zu versetzen, wenn der Einlernvorgang abgeschlossen ist.
SwitchState_Teach()	Local	Methode, um den Funktionsbaustein in den Einlernmodus zu versetzen.
UpdateState()	Local	Methode zur Verwendung für den Übergang von einem Zustand in einen anderen Zustand.

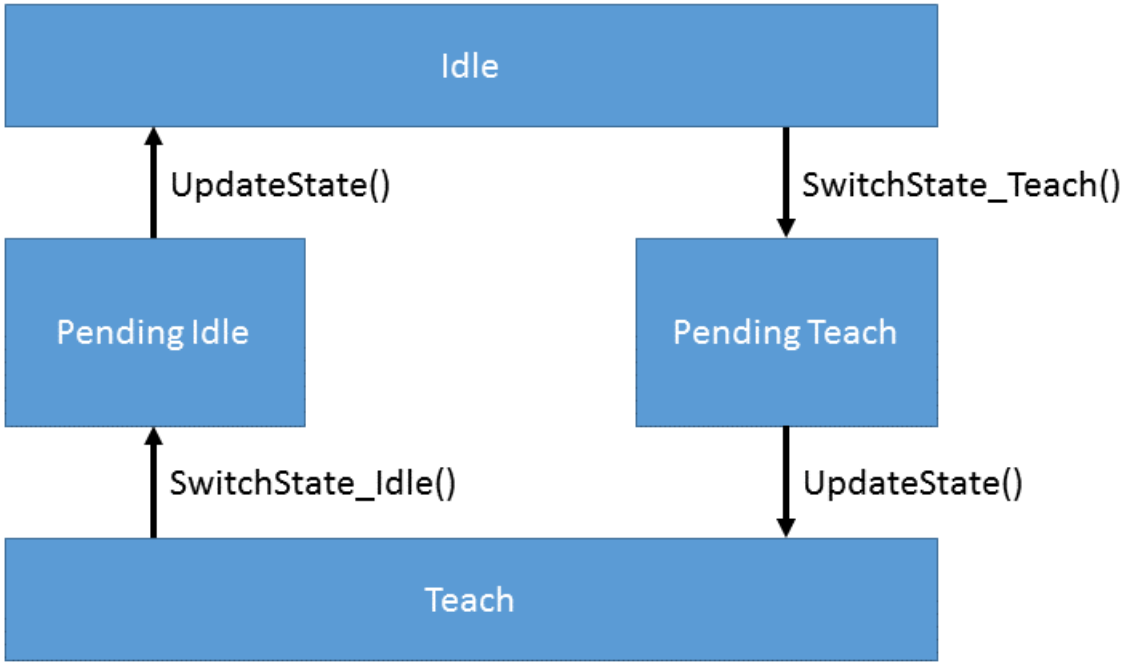
Zustandsdiagramm

Aufgrund des asynchronen Dateizugriffs in Echtzeitanwendungen benötigt dieser Funktionsbaustein eine Zustandsmaschine, um den Dateizugriff vorzubereiten und abzuschließen.

Beim Start ist der Funktionsbaustein im Zustand *Idle*. Um die eingehenden Daten in eine Datei zu schreiben, muss er in den Zustand *Teach* wechseln. Daher muss die Methode *SwitchState_Teach()* einmal aufgerufen werden, um den Funktionsbaustein in den Zustand *PendingTeach* zu versetzen. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Teach* befindet. In

diesem Zustand können ein oder mehrere Einlernzyklen durchlaufen werden. Wenn der Funktionsbaustein keine weiteren Zyklen einlernen soll, kann er wieder in den Zustand *Idle* versetzt werden. Um den Zustandswechsel zu initiieren, muss die Methode *SwitchState_Idle()* aufgerufen werden. Anschließend muss die Methode *UpdateState()* aufgerufen werden, bis sich der Funktionsbaustein im Zustand *Idle* befindet.

Zustandsdiagramm für den Einlernvorgang der Daten:



Beispiel

```
VAR
    fbTimeBasedTeach : FB_ALY_TimeBasedTeachPath_1Ch;
    eTeachMode : E_ALY_TeachMode := E_ALY_TeachMode.Mean;
    nSegmentSize : UDINT := 200;
    tTimeout : TIME := T#5S;
    nSetNumTeaches : UDINT := 10;
    bInvolveExistingFile : BOOL := TRUE;
    sFilePath : STRING := 'C:\TwinCAT\3.1\Boot\TimeBasedTeach.tas';
    bNegateStartPeriod : BOOL := FALSE;
    bNegateStopPeriod : BOOL := FALSE;
    bConfigure : BOOL := TRUE;
    eState : E_ALY_TeachState := E_ALY_TeachState.Idle;
    bTeach : BOOL;
    fInput : LREAL;
    bStartPeriod : BOOL;
    bStopPeriod : BOOL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbTimeBasedTeach.Configure(eTeachMode, nSegmentSize, tTimeout, nSetNumTeaches, bInvolveExistingF
ile, sFilePath, bNegateStartPeriod, bNegateStopPeriod);
END_IF

// Call algorithm
eState := fbTimeBasedTeach.eState;
CASE eState OF
E_ALY_TeachState.Idle:
    IF bTeach THEN
        fbTimeBasedTeach.SwitchState_Teach();
        fbTimeBasedTeach.UpdateState();
    END_IF
E_ALY_TeachState.Teach:
    fbTimeBasedTeach.SetChannelValue(fInput);
    fbTimeBasedTeach.Call(bStartPeriod := bStartPeriod, bStopPeriod := bStopPeriod);

    IF NOT bTeach THEN
```

```
        fbTimeBasedTeach.SwitchState_Idle();
        fbTimeBasedTeach.UpdateState();
    END_IF
E_ALY_TeachState.Pending,
E_ALY_TeachState.PendingIdle,
E_ALY_TeachState.PendingTeach:
    fbTimeBasedTeach.UpdateState();
    eState := fbTimeBasedTeach.eState;
END_CASE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.7.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    bStartPeriod : ULINT;
    bStopPeriod : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bStartPeriod	BOOL	Steigende Flanke startet Einlernvorgang.
bStopPeriod	BOOL	Steigende Flanke stoppt Einlernvorgang.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.7.1.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eTeachMode : E_ALY_TeachMode;
    nSegmentSize : UDINT;
    tTimeout : TIME;
    nSetNumTeaches : UDINT;
    bInvolveExistingFile : BOOL;
    sFilePath : STRING(255);
    bNegateStartPeriod : BOOL;
    bNegateStopPeriod : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eTeachMode	E_ALY_TeachMode [▶ 356]	Konfigurieren des Einlernmodus *Minimum*: Minimum des gespeicherten Datenpunkts und Eingangsdatenpunkts. *Maximum*: Maximum des gespeicherten Datenpunkts und Eingangsdatenpunkts. *Mean:* Mittelwert des gespeicherten Datenpunkts und Eingangsdatenpunkts in Bezug auf nCurrentTeachCycles.
nSegmentSize	UDINT	Anzahl der gepufferten Elemente für Dateioperation.
tTimeout	TIME	Zeitüberschreitung für asynchrone Operationen.
nSetNumTeaches	UDINT	Einlernanzahl, die der Datei im Zustand Read hinzugefügt werden soll. „0“ = nicht begrenzt.
bInvolveExistingFile	BOOL	Wenn TRUE, vorhandene Datei einbeziehen (falls vorhanden). Wenn FALSE, neue Datei erstellen.
sFilePath	STRING(255)	Pfad zur eingelernten Datei, z. B. *C:\TwinCAT\3.1\Boot\Teach.tas*
bNegateStartPeriod	BOOL	Negieren des Eingangsparameters bStartPeriod.
bNegateStopPeriod	BOOL	Negieren des Eingangsparameters bStopPeriod.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.7.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.7.1.4 GetBusyState

Gibt TRUE zurück, wenn der Funktionsbaustein aufgrund eines asynchronen Dateizugriffs beschäftigt ist.

Syntax

Definition:

```
METHOD GetBusyState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetBusyState	BOOL	Gibt `TRUE` zurück, wenn der FB aktiv ist.

5.1.1.7.1.5 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.7.1.6 SwitchState_Idle

Initiieren des Wechsels vom Zustand *Teach* in den Zustand *Idle*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Idle : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Idle	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.7.1.7 SwitchState_Teach

Initiieren des Wechsels vom Zustand *Idle* in den Zustand *Teach*. Siehe Zustandsdiagramm.

Syntax

Definition:

```
METHOD SwitchState_Teach : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SwitchState_Teach	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

5.1.1.7.1.8 UpdateState

Aktualisierung des Zustands, nachdem eine Zustandsänderung initiiert wurde und bis der Zielzustand nicht erreicht ist.

Syntax

Definition:

```
METHOD UpdateState : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
UpdateState	BOOL	Gibt `TRUE` zurück, wenn erfolgreich

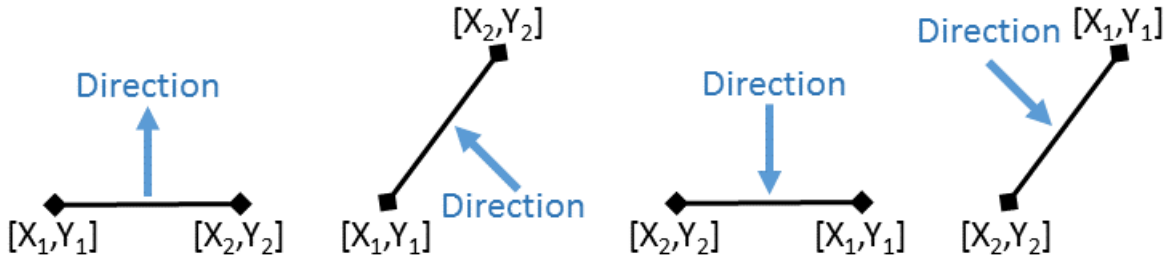
5.1.1.8 XY Path

5.1.1.8.1 FB_ALY_XyGateMonitor_2Ch

XY Gate Monitor 2Ch zählt die Anzahl der Schnittpunkte eines XY-Eingangs mit einem bestimmten Gate oder seiner Projektion (gerade Linie zwischen den Gate-Punkten) abhängig vom konfigurierten Gate Mode. Die Analyseperiode kann mit den Eingängen *Start* und *Stop* gestartet werden. Der Algorithmus ist richtungsempfindlich, d. h. es werden nur Schnittpunkte in der richtigen Richtung gezählt. Die Richtungsauslegung hängt von der Reihenfolge der Gate-Punkte (X1/Y1) und (X2/Y2) ab. Die möglichen Richtungen der Schnittpunkte sind nachstehend veranschaulicht.

Richtungen der Schnittpunkte:

Der blaue Pfeil stellt die Signalrichtung dar und die schwarzen Linien veranschaulichen das Gate mit seinen Gate-Punkten (X1/Y1) und (X2/Y2). Die Richtung der Schnittpunkte wird gezählt, wenn sich das Signal gegen den Uhrzeigersinn um den ersten Gate-Punkt (X1/Y1) dreht.



Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XyGateMonitor_2Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bExecuting: BOOL;
    bGateIntersection: BOOL;
    bOutlierIntersection: BOOL;
    fPosIntersectionX: LREAL;
    fPosIntersectionY: LREAL;
    nCountGateIntersections: ULINT;
    nCountOutlierIntersections: ULINT;
    fbTimeLastIntersection: FB_ALY_DateTime;
    eClassification: E_ALY_Classification_2Cls;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bExecuting	BOOL	TRUE, wenn die Call()-Methode mit bStart:=TRUE aufgerufen wurde. FALSE, wenn die Call-Methode mit bStop:=TRUE aufgerufen wurde.
bGateIntersection	BOOL	TRUE, wenn ein Gate-Schnittpunkt erkannt wurde.
bOutlierIntersection	BOOL	TRUE, wenn eine Ausreißer-Schnittpunkt erkannt wurde.
fPosIntersectionX	LREAL	X-Koordinate des letzten Schnittpunkts.
fPosIntersectionY	LREAL	Y-Koordinate des letzten Schnittpunkts.
nCountGateIntersections	ULINT	Anzahl der erkannten Schnittpunkte.
nCountOutlierIntersections	ULINT	Anzahl der erkannten Ausreißer.
fbTimeLastIntersection	FB_ALY_DateTime	Zeitstempel des letzten Schnittpunkts.
eClassification	E_ALY_Classification_2Cls	Einstufungsergebnis. Abhängig vom konfigurierten Gate Mode.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXyGateMonitor : FB_ALY_XyGateMonitor_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    eGateMode : E_ALY_GateMode := E_ALY_GateMode.IntersectGate;
    stGatePos1 : ST_ALY_XyPosition := ( X:= 0.0, Y := 0.0);
    stGatePos2 : ST_ALY_XyPosition := ( X:= 0.0, Y := 4.0);
    bConfigure : BOOL := TRUE;
    fInputChX : LREAL;
    fInputChY : LREAL;
    bStart : BOOL;
    bStop : BOOL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbXyGateMonitor.Configure(eGateMode, stGatePos1, stGatePos2);
END_IF

// Get current system time
```

```
fbSystemTime.Call();

// Call algorithm
fbXyGateMonitor.SetChannelValue(1, fInputChX);
fbXyGateMonitor.SetChannelValue(2, fInputChY);
fbXyGateMonitor.Call(bStart, bStop, fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.8.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    bStart : BOOL;
    bStop : BOOL;
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bStart	BOOL	Start der Gate-Beobachtung.
bStop	BOOL	Stopp der Gate-Beobachtung.
tTimestamp	ULINT	Aktueller Zeitstempel.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.1.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.8.1.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    eGateMode : E_ALY_GateMode;
```

```
    stGatePos1 : ST_ALY_XyPosition;
    stGatePos2 : ST_ALY_XyPosition;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
eGateMode	E_ALY_GateMode [▶ 356]	Konfigurierter Gate-Modus *Intersect Gate:* Bestimmt, ob das XY-Signal das Gate in der konfigurierten Richtung kreuzt. Liegt während der Analyseperiode ein Schnittpunkt vor, wird dies als OK eingestuft, ansonsten NOK. *Not Intersect Gate:* Überwacht, ob sich das XY-Signal während der Analyseperiode nicht mit dem Gate in der konfigurierten Richtung kreuzt. Dann wird dies als OK eingestuft, ansonsten NOK. *Intersect Projection:* Bestimmt, ob das XY-Signal die Projektion des Gates in der konfigurierten Richtung kreuzt. Liegt während der Analyseperiode ein Schnittpunkt vor, wird dies als OK eingestuft, ansonsten NOK. *Not Intersect Projection:* Überwacht, ob sich das XY-Signal während der Analyseperiode nicht mit der Projektion des Gates in der konfigurierten Richtung kreuzt. Dann wird dies als OK eingestuft, ansonsten NOK. *Intersect Gate Or Projection:* Bestimmt, ob das XY-Signal das Gate oder seine Projektion in der konfigurierten Richtung kreuzt. Liegt während der Analyseperiode ein Schnittpunkt vor, wird dies als OK eingestuft, ansonsten NOK.
stCorner1	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der ersten Ecke in XY-Koordinaten.
stCorner2	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der zweiten Ecke in XY-Koordinaten.
stCorner3	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der dritten Ecke in XY-Koordinaten.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Wert der X-Koordinate 2: Wert der Y-Koordinate
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.2 FB_ALY_XyShapeMonitor_Circle_2Ch

Der *XY Shape Monitor Circle 2Ch* zählt die Anzahl der Schnittpunkte eines XY-Eingangs mit einer bestimmten Kreisform.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XyShapeMonitor_Circle_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bWithinShape: BOOL;
    bIntersection: BOOL;
    nCountIntersections: ULINT;
    fbTimeLastIntersection: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert `TRUE`, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bWithinShape	BOOL	TRUE, wenn sich das Eingangssignal gegenwärtig innerhalb der bestimmten Form befindet.
bIntersection	BOOL	TRUE, wenn das Eingangssignal gegenwärtig die bestimmte Form kreuzt.
nCountIntersections	ULINT	Gesamtanzahl der Schnittpunkte von Eingangssignal und Form.
fbTimeLastIntersection	FB_ALY_DateTime	Zeitstempel des letzten Schnittpunkts.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXyShapeMonitor_Circle : FB_ALY_XyShapeMonitor_Circle_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stCentre : ST_ALY_XyPosition := ( X:= 4.0, Y := 2.0);
    fRadius : LREAL := 1.5;
    bConfigure : BOOL := TRUE;
    fInputChX : LREAL;
    fInputChY : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbXyShapeMonitor_Circle.Configure(stCentre, fRadius);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbXyShapeMonitor_Circle.SetChannelValue(1, fInputChX);
fbXyShapeMonitor_Circle.SetChannelValue(2, fInputChY);
fbXyShapeMonitor_Circle.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.8.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.8.2.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.8.2.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    stCentre : ST_ALY_XyPosition;
    fRadius : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stCentre	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position des Kreismittelpunkts in XY-Koordinaten.
fRadius	LREAL	Radius des Kreises.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.8.2.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Wert der X-Koordinate 2: Wert der Y-Koordinate
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.8.3 FB_ALY_XyShapeMonitor_Rectangle_2Ch

Der *XY Shape Monitor Rectangle 2Ch* zählt die Anzahl der Schnittpunkte eines XY-Eingangs mit einer bestimmten Rechteckform.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XyShapeMonitor_Rectangle_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bWithinShape: BOOL;
    bIntersection: BOOL;
    nCountIntersections: ULINT;
    fbTimeLastIntersection: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bWithinShape	BOOL	TRUE, wenn sich das Eingangssignal gegenwärtig innerhalb der bestimmten Form befindet.
bIntersection	BOOL	TRUE, wenn das Eingangssignal gegenwärtig die bestimmte Form kreuzt.
nCountIntersections	ULINT	Gesamtanzahl der Schnittpunkte von Eingangssignal und Form.
fbTimeLastIntersection	FB_ALY_DateTime	Zeitstempel des letzten Schnittpunkts.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXyShapeMonitor_Rectangle : FB_ALY_XyShapeMonitor_Rectangle_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stLowerLeftCorner : ST_ALY_XyPosition := ( X:= 0.0, Y := 0.0);
    fLength_X : LREAL := 10;
    fLength_Y : LREAL := 5;
    bConfigure : BOOL := TRUE;
    fInputChX : LREAL;
    fInputChY : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbXyShapeMonitor_Rectangle.Configure(stLowerLeftCorner, fLength_X, fLength_Y );
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
```

```
fbXyShapeMonitor_Rectangle.SetChannelValue(1, fInputChX);
fbXyShapeMonitor_Rectangle.SetChannelValue(2, fInputChY);
fbXyShapeMonitor_Rectangle.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.8.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.3.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.8.3.3 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    stLowerLeftCorner : ST_ALY_XyPosition;
    fLength_X : LREAL;
    fLength_Y : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stLowerLeftCorner	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der unteren linken Ecke des Rechtecks in XY-Koordinaten.
fLenght_X	LREAL	Länge des Rechtecks in positiver X-Richtung.
fLenght_Y	LREAL	Länge des Rechtecks in positiver Y-Richtung.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.3.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Wert der X-Koordinate 2: Wert der Y-Koordinate
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.4 FB_ALY_XyShapeMonitor_Triangle_2Ch

Der *XY Shape Monitor Triangle 2Ch* zählt die Anzahl der Schnittpunkte eines XY-Eingangs mit einer bestimmten Dreieckform.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XyShapeMonitor_Triangle_2Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bWithinShape: BOOL;
```

```
    bIntersection: BOOL;
    nCountIntersections: ULINT;
    fbTimeLastIntersection: FB_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
bWithinShape	BOOL	TRUE, wenn sich das Eingangssignal gegenwärtig innerhalb der bestimmten Form befindet.
bIntersection	BOOL	TRUE, wenn das Eingangssignal gegenwärtig die bestimmte Form kreuzt.
nCountIntersections	ULINT	Gesamtanzahl der Schnittpunkte von Eingangssignal und Form.
fbTimeLastIntersection	FB_ALY_DateTime	Zeitstempel des letzten Schnittpunkts.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
ConfigureChannel()	Local	Kanalspezifische Konfiguration für den jeweiligen Algorithmus.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXyShapeMonitor_Triangle : FB_ALY_XyShapeMonitor_Triangle_2Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    stCorner1 : ST_ALY_XyPosition := ( X:= 0.0, Y := 0.0);
    stCorner2 : ST_ALY_XyPosition := ( X:= 0.0, Y := 6.0);
    stCorner3 : ST_ALY_XyPosition := ( X:= 4.0, Y := 3.0);
    bConfigure : BOOL := TRUE;
    fInputChX : LREAL;
    fInputChY : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;
```

```
    fbXyShapeMonitor_Triangle.Configure(stCorner1, stCorner2, stCorner3);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbXyShapeMonitor_Triangle.SetChannelValue(1, fInputChX);
fbXyShapeMonitor_Triangle.SetChannelValue(2, fInputChY);
fbXyShapeMonitor_Triangle.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.8.4.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.4.2 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.8.4.3 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex 1: Wert der X-Koordinate 2: Wert der Y-Koordinate
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.8.4.4 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    stCorner1 : ST_ALY_XyPosition;
    stCorner2 : ST_ALY_XyPosition;
    stCorner3 : ST_ALY_XyPosition;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stCorner1	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der ersten Ecke in XY-Koordinaten.
stCorner2	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der zweiten Ecke in XY-Koordinaten.
stCorner3	ST_ALY_XyPosition [▶ 352]	Struktur zum Speichern der Position der dritten Ecke in XY-Koordinaten.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9 Specific

5.1.1.9.1 XTS

5.1.1.9.1.1 FB_ALY_XtsAccelerationAnalysis_1Ch

Der *XTS Acceleration Analysis 1Ch* berechnet die aktuelle Beschleunigung eines XTS-Movers. Zu diesem Zweck muss die Länge des XTS in Millimetern angegeben werden, und als Eingangssignal wird die Position des Movers benötigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XtsAccelerationAnalysis_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fAcceleration: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fAcceleration	LREAL	Aktuelle Beschleunigung des XTS-Movers in m/s².

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXtsAcceleration : FB_ALY_XtsAccelerationAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nXtsLenght : UDINT := 4000;
    bConfigure : BOOL := TRUE;
    fPosition : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

   fbXtsAcceleration.Configure(nXtsLenght);
```

```
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbXtsAcceleration.SetChannelValue(fPosition);
fbXtsAcceleration.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.9.1.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9.1.1.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nXtsLength : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nXtsLength	UDINT	Länge des gegebenen XTS-Systems in Millimetern.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9.1.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.9.1.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.1.2 FB_ALY_XtsDistanceIntegrator_1Ch

Der *XTS Distance Integrator 1Ch* berechnet die zurückgelegte Entfernung eines XTS-Movers. Der Algorithmus liefert die Gesamtentfernung, die positive Entfernung und die negative Entfernung. Zu diesem Zweck muss die Länge des XTS in Millimetern angegeben werden, und als Eingangssignal wird die Position des Movers benötigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XtsDistanceIntegrator_1Ch
VAR_INPUT
    bPersistent: BOOL;
END_VAR
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fDistance: LREAL;
    fDistancePos: LREAL;
    fDistanceNeg: LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bPersistent	BOOL	Ist der Wert TRUE, werden die internen Daten persistent gespeichert.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fDistance	LREAL	Gesamtentfernung, die der XTS-Mover zurückgelegt hat. In m.
fDistancePos	LREAL	Positive Entfernung, die der XTS-Mover zurückgelegt hat (Richtung: vorwärts). In m.
fDistanceNeg	LREAL	Negative Entfernung, die der XTS-Mover zurückgelegt hat (Richtung: rückwärts). In m.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXtsDistance : FB_ALY_XtsDistanceIntegrator_1Ch;
    nXtsLenght : UDINT := 4000;
    bConfigure : BOOL := TRUE;
    fPosition : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

   fbXtsDistance.Configure(nXtsLenght);
END_IF

// Call algorithm
fbXtsDistance.SetChannelValue(fPosition);
fbXtsDistance.Call();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.9.1.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.1.2.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nXtsLength : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nXtsLength	UDINT	Länge des gegebenen XTS-Systems in Millimetern.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.1.2.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.9.1.2.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.1.3 FB_ALY_XtsVelocityAnalysis_1Ch

XTS Velocity Analysis 1Ch berechnet die aktuelle Geschwindigkeit eines XTS-Movers. Zu diesem Zweck muss die Länge des XTS in Millimetern angegeben werden, und als Eingangssignal wird die Position des Movers benötigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_XtsVelocityAnalysis_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fVelocity: LRAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fVelocity	LREAL	Aktuelle Geschwindigkeit des XTS-Movers in m/s.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbXtsVelocity : FB_ALY_XtsVelocityAnalysis_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nXtsLenght : UDINT := 4000;
    bConfigure : BOOL := TRUE;
    fPosition : LREAL;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

   fbXtsVelocity.Configure(nXtsLenght);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbXtsVelocity.SetChannelValue(fPosition);
fbXtsVelocity.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.9.1.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9.1.3.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nXtsLength : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nXtsLength	UDINT	Länge des gegebenen XTS-Systems in Millimetern.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9.1.3.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.9.1.3.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.9.2 Wind Turbine

5.1.1.9.2.1 FB_ALY_WtTurbulence_1Ch

WT Turbulence 1Ch berechnet den Mittelwert der Windgeschwindigkeit, die Turbulenz und die Turbulenzintensität nach der Norm *EN 61400-1*. Als Eingangssignal wird die Windgeschwindigkeit benötigt. Die Ausgangswerte werden in einem Zyklus von 10 Minuten aktualisiert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_WtTurbulence_1Ch
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fMean: LREAL;
    fTurbulence: LREAL;
    fTurbulenceIntensity: LREAL;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fMean	LREAL	Mittelwert der Windgeschwindigkeit.
fTurbulence	LREAL	Turbulenz des Windes. Nach der EN-Norm ist dies die Standardabweichung der Windgeschwindigkeit während eines Zeitintervalls von 10 Minuten.
fTurbulenceIntensity	LREAL	Intensität der Windturbulenz.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Beispiel

```
VAR
    fbWtTurbulence : FB_ALY_WtTurbulence_1Ch;
    fbSystemTime : FB_ALY_GetSystemTime;
    nNumCycles : UDINT;
    bConfigure : BOOL := TRUE;
    nInput : INT;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    fbWtTurbulence.Configure(nNumCycles);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbWtTurbulence.SetChannelValue(nInput);
fbWtTurbulence.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.9.2.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.2.1.2 Configure

Konfigurieren des Algorithmus. Kanalspezifische Parameter werden mit Hilfe der Methode ConfigueChannel() festgelegt.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    nNumCycles : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nNumCycles	UDINT	Gibt die Anzahl der SPS-Zyklen an, die in das Zeitintervall für die Berechnung passen, nach EN 61400-1 ist dies ein Intervall von zehn Minuten.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.9.2.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.9.2.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10 Reporting

Das 24/7-Reporting kann in TwinCAT Analytics mit den Algorithmen der Kategorie *Reporting* umgesetzt werden. Die Reporting Collectoren sammeln die Daten und versenden diese an den Reporting-Server. Die Reporting Trigger lösen die Erstellung eines Reports aus.

5.1.1.10.1 Reporting Collector

Die Reporting Collectoren sammeln Daten und versenden diese nach einem Event in einer Daten-Nachricht an den Reporting Server.

5.1.1.10.1.1 Reporting Collector Edge

Der Reporting Collector Edge sammelt die Daten der Eingangskanäle und sendet die Daten je nach Konfiguration nach einem Ereignis oder nach dem Befüllen des Buffers an den Reporting Server. Ein Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einer bestimmten Schwelle passiert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_CollectorEdge
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    nBufferCount: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
nBufferCount	UDINT	Gibt die Anzahl an Elementen im Buffer an.

Methoden

Name	Typ	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetDisplayName()	Local	Methode zur Beschreibung des Eingangskanals.
SetEdgeValue()	Local	Methode zur Übergabe von Werten an den Edge-Channel des Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbRpt_CollectorEdge: FB_Rpt_CollectorEdge(sNetID:='', nNumChannels:=3);
    fbSystemTime : FB_ALY_GetSystemTime;
    bEdge: BOOL;
    fDataInCh1: LREAL;
    fDataInCh2: LREAL;
    fDataInCh3: LREAL;

    stThresholdEdge: ST_ALY_Threshold;
    sReportName: STRING(255);
    fTolerance: LREAL;
    sDataKey: STRING(255);
    nBufferSize: UDINT;
    bIncludeTimestamps: BOOL;
    aChannelNames: ARRAY [1..3] OF STRING(255);
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    fTolerance:= 0;
    sDataKey:= 'DataKey101';
    nBufferSize:= 1;
```

```
    bIncludeTimestamps:= TRUE;
    aChannelNames[1]:= 'Min';
    aChannelNames[2]:= 'Max';
    aChannelNames[3]:= 'Avg';
    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbRpt_CollectorEdge.Configure(
        sConfigId:= sReportName,
        sDataKey:= sDataKey,
        nBufferSize:= nBufferSize,
        bBufferOnEvent:= bBufferOnEvent,
        bIncludeTimestamps:= bIncludeTimestamps,
        stThresholdEdge:= stThresholdEdge);

    fbRpt_CollectorEdge.fTolerance:= fTolerance;
    fbRpt_CollectorEdge.SetDisplayName(nChannel:= 1, sDisplayName:= aChannelNames[1]);
    fbRpt_CollectorEdge.SetDisplayName(nChannel:= 2, sDisplayName:= aChannelNames[2]);
    fbRpt_CollectorEdge.SetDisplayName(nChannel:= 3, sDisplayName:= aChannelNames[3]);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbRpt_CollectorEdge.SetChannelValue(nChannel:= 1, Input:= fDataInCh1);
fbRpt_CollectorEdge.SetChannelValue(nChannel:= 2, Input:= fDataInCh2);
fbRpt_CollectorEdge.SetChannelValue(nChannel:= 3, Input:= fDataInCh3);

fbRpt_CollectorEdge.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.1.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.1.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    sDataKey : STRING;
    stThresholdEdge : ST_ALY_Threshold;
    bIncludeTimestamps : BOOL;
    nBufferSize : UDINT;
    bBufferOnEvent : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
sDataKey	STRING	Der Data Key sollte innerhalb eines Reports eindeutig sein. Über den Data Key lässt sich das Daten Objekt eindeutig in den Report einordnen.
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.
bIncludeTimestamps	BOOL	Fügt eine Spalte mit den Zeitstempeln ein.
nBufferSize	UDINT	Gibt die Größe der gepufferten Daten an. Ist die Buffer Size gleich eins, wird eine Key-Value-Struktur aufgebaut. Ist die Buffer Size größer als eins, werden die Daten in einer Tabelle dargestellt.
bBufferOnEvent	BOOL	Gibt an, wie die Daten gesammelt und gepuffert werden sollen. Wenn der Parameter True ist, wird bei jedem Edge-Signal am Eingang die Daten der Inputs im Buffer zwischengespeichert. Sobald der Ausgang Buffer Count den gleichen Wert wie der Parameter Buffer Size hat, werden die Daten an den TcReportingServer versendet. Wenn der Parameter False ist, werden bei jedem Zyklus die Daten der Inputs im Buffer zwischengespeichert. Sobald die Buffer Size erreicht ist, ersetzen die neuen Daten die vorherigen Daten. Bei einem Edge-Signal am Eingang werden die Daten an den TcReportingServer versendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.10.1.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.1.5 SetDisplayName

Festlegen einer kanalspezifischen Beschreibung. Im Reporting dient die Beschreibung bei einer Tabelle als Tabellenüberschrift und bei einem Key-Value-Paar als Key-Wert.

Syntax

Defintion:

```
METHOD SetDisplayName : BOOL
VAR_INPUT
    nChannel : UDINT;
    sDisplayName : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
sDisplayName	STRING(255)	Eingangswert vom Typ String.

Rückgabewert

Name	Typ	Beschreibung
SetDisplayName	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.1.6 SetEdgeValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetEdgeValue : BOOL
VAR_INPUT
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.1.2 Reporting Collector Interval

Der Reporting Collector Interval sammelt die Daten der Eingangskanäle und sendet die Daten je nach Konfiguration nach einem Ereignis oder nach dem Befüllen des Buffers an den Reporting Server. Ein Ereignis wird ausgelöst, wenn die Zeitspanne des konfigurierten Intervalls abgelaufen ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_CollectorInterval
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    fbTimeCurrentInterval: FB_ALY_Timespan;
    nBufferCount: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
fbTimeCurrentInterval	FB_ALY_Timespan	Gibt die Zeitspanne bis zum nächsten Event an.
nBufferCount	UDINT	Gibt die Anzahl an Elementen im Buffer an.

Methoden

Name	Typ	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetDisplayName()	Local	Methode zur Beschreibung des Eingangskanals.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Beispiel

```
VAR
    fbRpt_CollectorInterval: FB_Rpt_CollectorInterval(sNetID:='', nNumChannels:=3);
    fbSystemTime : FB_ALY_GetSystemTime;
    fDataInCh1: LREAL;
    fDataInCh2: LREAL;
    fDataInCh3: LREAL;

    tInterval : LTIME;
    sReportName: STRING(255);
    sDataKey: STRING(255);
    nBufferSize: UDINT;
    bIncludeTimestamps: BOOL;
    aChannelNames: ARRAY [1..3] OF STRING(255);
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    sDataKey:= 'DataKey101';
    nBufferSize:= 1;
    bIncludeTimestamps:= TRUE;
    aChannelNames[1]:= 'Min';
    aChannelNames[2]:= 'Max';
    aChannelNames[3]:= 'Avg';
    tInterval := LTIME#5S;

    fbRpt_CollectorInterval.Configure(
        sConfigId:= sReportName,
        sDataKey:= sDataKey,
        tInterval:= tInterval,
        nBufferSize:= nBufferSize,
        bBufferOnEvent:= bBufferOnEvent,
        bIncludeTimestamps:= bIncludeTimestamps);

    fbRpt_CollectorInterval.SetDisplayName(nChannel:= 1, sDisplayName:= aChannelNames[1]);
    fbRpt_CollectorInterval.SetDisplayName(nChannel:= 2, sDisplayName:= aChannelNames[2]);
    fbRpt_CollectorInterval.SetDisplayName(nChannel:= 3, sDisplayName:= aChannelNames[3]);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbRpt_CollectorInterval.SetChannelValue(nChannel:= 1, Input:= fDataInCh1);
fbRpt_CollectorInterval.SetChannelValue(nChannel:= 2, Input:= fDataInCh2);
fbRpt_CollectorInterval.SetChannelValue(nChannel:= 3, Input:= fDataInCh3);

fbRpt_CollectorInterval.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.1.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.2.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    sDataKey : STRING;
    tInterval : LTIME;
    bIncludeTimestamps : BOOL;
    nBufferSize : UDINT;
    bBufferOnEvent : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
sDataKey	STRING	Der Data Key sollte innerhalb eines Reports eindeutig sein. Über den Data Key lässt sich das Daten Objekt eindeutig in den Report einordnen.
tInterval	LTIME	Gibt den Zeitintervall an, in dem die Daten an den Reporting Server gesendet werden sollen.
bIncludeTimestamps	BOOL	Fügt eine Spalte mit den Zeitstempeln ein.
nBufferSize	UDINT	Gibt die Größe der gepufferten Daten an. Ist die Buffer Size gleich eins, wird eine Key-Value-Struktur aufgebaut. Ist die Buffer Size größer als eins, werden die Daten in einer Tabelle dargestellt.
bBufferOnEvent	BOOL	Gibt an, wie die Daten gesammelt und gepuffert werden sollen. Wenn der Parameter True ist, wird bei jedem Edge-Signal am Eingang die Daten der Inputs im Buffer zwischengespeichert. Sobald der Ausgang Buffer Count den gleichen Wert wie der Parameter Buffer Size hat, werden die Daten an den TcReportingServer versendet. Wenn der Parameter False ist, werden bei jedem Zyklus die Daten der Inputs im Buffer zwischengespeichert. Sobald die Buffer Size erreicht ist, ersetzen die neuen Daten die vorherigen Daten. Bei einem Edge-Signal am Eingang werden die Daten an den TcReportingServer versendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.2.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.10.1.2.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.2.5 SetDisplayName

Festlegen einer kanalspezifischen Beschreibung. Im Reporting dient die Beschreibung bei einer Tabelle als Tabellenüberschrift und bei einem Key-Value-Paar als Key-Wert.

Syntax

Defintion:

```
METHOD SetDisplayName : BOOL
VAR_INPUT
    nChannel : UDINT;
    sDisplayName : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
sDisplayName	STRING(255)	Eingangswert vom Typ String.

Rückgabewert

Name	Typ	Beschreibung
SetDisplayName	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.2.6 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.1.3 Reporting Collector Time

Der Reporting Collector Time sammelt die Daten der Eingangskanäle und sendet die Daten je nach Konfiguration nach einem Ereignis oder nach dem Befüllen des Buffers an den Reporting Server. Ein Ereignis wird ausgelöst, wenn die konfigurierte Einschaltzeit erreicht wird. Die Einschaltzeit und die Tage der Wochen lassen sich konfigurieren.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_CollectorTime
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    fbTimeUntilNextSwitch: FB_ALY_Timespan;
    nBufferCount: UDINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
fbTimeUntilNextSwitch	FB_ALY_Timespan	Gibt die Zeitspanne bis zum nächsten Event an.
nBufferCount	UDINT	Gibt die Anzahl an Elementen im Buffer an.

Methoden

Name	Typ	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
SetDisplayName()	Local	Methode zur Beschreibung des Eingangskanals.

Beispiel

```
VAR
    fbRpt_CollectorTime: FB_Rpt_CollectorTime(sNetID:='', nNumChannels:=3);
    fbSystemTime : FB_ALY_GetSystemTime;
    fDataInCh1: LREAL;
    fDataInCh2: LREAL;
    fDataInCh3: LREAL;

    tTimeOn : LTIME;
    nDayOfWeekMask : WORD;
    sReportName: STRING(255);
    sDataKey: STRING(255);
    nBufferSize: UDINT;
    bIncludeTimestamps: BOOL;
    aChannelNames: ARRAY [1..3] OF STRING(255);
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    sDataKey:= 'DataKey101';
    nBufferSize:= 1;
    bIncludeTimestamps:= TRUE;
    aChannelNames[1]:= 'Min';
    aChannelNames[2]:= 'Max';
    aChannelNames[3]:= 'Avg';
    tTimeOn := LTIME#5H;
    nDayOfWeekMask := 2#0011_0101;

    fbRpt_CollectorTime.Configure(
        sConfigId:= sReportName,
        sDataKey:= sDataKey,
        tTimeOn:= tTimeOn,
        nDayOfWeekMask:= nDayOfWeekMask,
        nBufferSize:= nBufferSize,
        bBufferOnEvent:= bBufferOnEvent
        bIncludeTimestamps:= bIncludeTimestamps);

    fbRpt_CollectorTime.SetDisplayName(nChannel:= 1, sDisplayName:= aChannelNames[1]);
    fbRpt_CollectorTime.SetDisplayName(nChannel:= 2, sDisplayName:= aChannelNames[2]);
    fbRpt_CollectorTime.SetDisplayName(nChannel:= 3, sDisplayName:= aChannelNames[3]);
END_IF

// Get current system time
fbSystemTime.Call();

// Call algorithm
fbRpt_CollectorTime.SetChannelValue(nChannel:= 1, Input:= fDataInCh1);
fbRpt_CollectorTime.SetChannelValue(nChannel:= 2, Input:= fDataInCh2);
fbRpt_CollectorTime.SetChannelValue(nChannel:= 3, Input:= fDataInCh3);

fbRpt_CollectorTime.Call(fbSystemTime.tSystemTime);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.1.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.1.3.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    sDataKey : STRING;
    tTimeOn : LTIME;
    nDayOfWeekMask : WORD;
    bIncludeTimestamps : BOOL;
    nBufferSize : UDINT;
    bBufferOnEvent : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
sDataKey	STRING	Der Data Key sollte innerhalb eines Reports eindeutig sein. Über den Data Key lässt sich das Daten Objekt eindeutig in den Report einordnen.
tTimeOn	LTIME	Gibt die Einschaltzeit an.
nDayOfWeekMask	WORD	Gibt die Wochentage, an denen die Zeitschaltung aktiv sein soll.
bIncludeTimestamps	BOOL	Fügt eine Spalte mit den Zeitstempeln ein.
nBufferSize	UDINT	Gibt die Größe der gepufferten Daten an. Ist die Buffer Size gleich eins, wird eine Key-Value-Struktur aufgebaut. Ist die Buffer Size größer als eins, werden die Daten in einer Tabelle dargestellt.
bBufferOnEvent	BOOL	Gibt an, wie die Daten gesammelt und gepuffert werden sollen. Wenn der Parameter True ist, wird bei jedem Edge-Signal am Eingang die Daten der Inputs im Buffer zwischengespeichert. Sobald der Ausgang Buffer Count den gleichen Wert wie der Parameter Buffer Size hat, werden die Daten an den TcReportingServer versendet. Wenn der Parameter False ist, werden bei jedem Zyklus die Daten der Inputs im Buffer zwischengespeichert. Sobald die Buffer Size erreicht ist, ersetzen die neuen Daten die vorherigen Daten. Bei einem Edge-Signal am Eingang werden die Daten an den TcReportingServer versendet.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.1.3.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.10.1.3.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.1.3.5 SetDisplayName

Festlegen einer kanalspezifischen Beschreibung. Im Reporting dient die Beschreibung bei einer Tabelle als Tabellenüberschrift und bei einem Key-Value-Paar als Key-Wert.

Syntax

Defintion:

```
METHOD SetDisplayName : BOOL
VAR_INPUT
    nChannel : UDINT;
    sDisplayName : STRING(255);
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
sDisplayName	STRING(255)	Eingangswert vom Typ String.

Rückgabewert

Name	Typ	Beschreibung
SetDisplayName	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.2 Reporting Trigger

Die Reporting Trigger versenden nach einem Event eine Trigger-Nachricht an den Reporting Server und lösen damit die Erstellung eines Reports aus.

5.1.1.10.2.1 Reporting Trigger Edge

Der Reporting Trigger Edge löst die Erstellung eines Reports aus, nachdem ein Ereignis ausgelöst wurde. Ein Ereignis wird ausgelöst, wenn das Signal des Eingangskanals die konfigurierte Flanke bei einem bestimmten Schwellenwert überschreitet. Intern bleiben die Eingänge, die einmal True waren auf True. Die Eingänge werden lediglich auf False zurückgesetzt, sobald alle Eingänge mindestens einmal True waren. Dadurch kann der Ausgang bNewResult von mehreren Reporting Collectoren als ein Eingang genutzt werden und sobald alle Reporting Collectoren eine Daten-Nachricht gesendet haben, wird eine Trigger-Nachricht versendet.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_TriggerEdge
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    sOverview: STRING(255) := '';
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
sOverview	STRING	Zeigt an, welche Eingangskanäle mindestens einmal TRUE waren.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
SetChannelValue()	Local	Methode zur Übergabe von Werten an den Algorithmus.

Eigenschaften

Name	Typ	Zugriff	Definitionsort	Initialwert	Beschreibung
fTolerance	LREAL	Get, Set	Lokal	0.0	Toleranzwert für die Vergleiche Equal / NotEqual

Beispiel

```
VAR
    fbRptTriggerEdge: FB_Rpt_TriggerEdge(sNetId:= '', nNumChannels:=2);
    fbSystemTime : FB_ALY_GetSystemTime;
    fEdgeCh1: BOOL;
    fEdgeCh2: BOOL;
    stThresholdEdge: ST_ALY_Threshold;
    fTolerance: LREAL;
    sReportName: STRING(255);
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    fTolerance:= 0;
    stThresholdEdge.eComparisonOperator := E_ALY_ComparisonOperator.Equals;
    stThresholdEdge.fThreshold := 1;

    fbRptTriggerEdge.Configure(sConfigId:= sReportName, stThresholdEdge := stThresholdEdge);
END_IF

fbRptTriggerEdge.SetChannelValue(nChannel:= 1, input:= fEdgeCh1);
fbRptTriggerEdge.SetChannelValue(nChannel:= 2, input:= fEdgeCh1);

fbRptTriggerEdge.Call(tTimestamp);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.2.1.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.2.1.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    stThresholdEdge : ST_ALY_Threshold;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
stThresholdEdge	ST_ALY_Threshold [▶ 352]	Kombination von Vergleichsoperator und Schwelle für die Flankenerkennung.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.2.1.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt TRUE zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.10.2.1.4 SetChannelValue

Festlegen eines kanalspezifischen Eingangswerts. Der Eingangswert wird erst verwendet, wenn die Call()-Methode aufgerufen worden ist.

Syntax

Definition:

```
METHOD SetChannelValue : BOOL
VAR_INPUT
    nChannel : UDINT;
    input : ANY;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nChannel	UDINT	Kanalindex (1 bis nChannels)
Input	ANY	Eingangswert eines beliebigen Datentyps.

Rückgabewert

Name	Typ	Beschreibung
SetChannelValue	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.2.2 Reporting Trigger Interval

Der Reporting Trigger Interval löst die Erstellung eines Reports aus, nachdem ein Ereignis ausgelöst wurde. Ein Ereignis wird ausgelöst, wenn die Zeitspanne des konfigurierten Intervalls abgelaufen ist.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_TriggerInterval
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    fbTimeCurrentInterval: FB_ALY_Timespan;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist `TRUE`, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang `TRUE`.
bConfigured	BOOL	Zeigt `TRUE` an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
fbTimeCurrentInterval	FB_ALY_Timespan	Gibt die Zeitspanne bis zum nächsten Event an.

Methoden

Name	Typ	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Beispiel

```
VAR
    fbRptTriggerInterval: FB_Rpt_TriggerInterval(sNetId:= '', nNumChannels:=2);
    fbSystemTime : FB_ALY_GetSystemTime;
    sReportName: STRING(255);
    tInterval: LTIME;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    tInterval := LTIME#5S;

    fbRptTriggerInterval.Configure(sConfigId:= sReportName, tInterval:= tInterval);
END_IF


fbRptTriggerInterval.Call(tTimestamp);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.2.2.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.2.2.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    tInterval : LTIME;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
tInterval	LTIME	Gibt den Zeitintervall an, in dem die Daten an den Reporting Server gesendet werden sollen.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.2.2.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.1.10.2.2.4 Pause

Mit der Pause-Methode ist es möglich, die Ausführung des Bausteins zu pausieren. Das intern ablaufende Zeitintervall läuft nach Aufruf der Pause-Funktion nicht weiter. Ein einmaliger Aufruf der Methode genügt. Erst beim nächsten Aufruf der Call-Methode wird das Intervall fortgesetzt.

Syntax

Definition:

```
METHOD Pause : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Pause	BOOL	Gibt TRUE zurück, wenn erfolgreich.

5.1.1.10.2.3 Reporting Trigger Time

Der Reporting Trigger Time löst die Erstellung eines Reports aus, nachdem ein Ereignis ausgelöst wurde. Ein Ereignis wird ausgelöst, wenn die konfigurierte Einschaltzeit erreicht wird. Die Einschaltzeit und die Tage der Wochen lassen sich konfigurieren.

Syntax

Definition:

```
FUNCTION_BLOCK FB_Rpt_TriggerTime
VAR_OUTPUT
    ipResultMessage: Tc3_EventLogger.I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    fbTimeLastEvent: FB_ALY_DateTime;
    fbTimeUntilNextSwitch: FB_ALY_Timespan;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Beinhaltet nähere Informationen zum aktuellen Rückgabewert. Für diesen speziellen Schnittstellenzeiger ist intern sichergestellt, dass er immer gültig/zugewiesen ist.
bError	BOOL	Der Ausgang ist TRUE, wenn ein Fehler auftritt.
bNewResult	BOOL	Wenn ein neues Ergebnis berechnet wurde, ist der Ausgang TRUE.
bConfigured	BOOL	Zeigt TRUE an, wenn der Baustein erfolgreich konfiguriert ist.
fbTimeLastEvent	FB_ALY_DateTime	Speichert den Zeitstempel der zuletzt versendeten Nachricht an den Reporting-Server.
fbTimeUntilNextSwitch	FB_ALY_Timespan	Gibt die Zeitspanne bis zum nächsten Event an.

Methoden

Name	Typ	Beschreibung
Call()	Local	Methode zur Berechnung der Ausgänge für eine bestimmte Konfiguration.
Configure()	Local	Allgemeine Konfiguration des Algorithmus mit seinen parametrisierten Bedingungen.
Reset()	Local	Setzt alle internen Zustände oder die bisher durchgeführten Berechnungen zurück.
Pause()	Local	Methode zum Pausieren der Ausführung inklusive der internen Zeitintervalle.

Beispiel

```
VAR
    fbRptTriggerTime: FB_Rpt_TriggerTime(sNetId:= '', nNumChannels:=2);
    fbSystemTime : FB_ALY_GetSystemTime;
```

```
    sReportName: STRING(255);
    nDayOfWeekMask : WORD;
END_VAR
```

```
// Configure algorithm
IF bConfigure THEN
    bConfigure := FALSE;

    sReportName:= 'Beckhoff Report Template';
    tTimeOn := LTIME#5H;
    nDayOfWeekMask := 2#0011_0101;

    fbRptTriggerTime.Configure(sConfigId:= sReportName, tTimeOn:= tTimeOn, nDayOfWeekMask:= nDayOfWe
ekMask);
END_IF

fbRptTriggerTime.Call(tTimestamp);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.1.10.2.3.1 Call

Aufrufen des Algorithmus nach Festlegung eines neuen Eingangswerts. Ein neuer Eingang kann mit Hilfe der Methode SetChannelValue() zugewiesen werden.

Syntax

Definition:

```
METHOD Call : BOOL
VAR_INPUT
    tTimestamp : ULINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
tTimestamp	ULINT	Aktueller Zeitstempel mit einer Auflösung von 1 ns.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.2.3.2 Configure

Konfigurieren des Algorithmus.

Syntax

Definition:

```
METHOD Configure : BOOL
VAR_INPUT
    sConfigId : STRING;
    tTimeOn : LTIME;
    nDayOfWeekMask : WORD;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
sConfigId	STRING	Gibt den Namen des Reports an. Der Name muss dem Namen der Konfigurationsdatei beim Reporting Server entsprechen.
tTimeOn	LTIME	Gibt die Einschaltzeit an.
nDayOfWeekMask	WORD	Gibt die Wochentage, an denen die Zeitschaltung aktiv sein soll.

Rückgabewert

Name	Typ	Beschreibung
Configure	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.1.10.2.3.3 Reset

Zurücksetzen des Algorithmus.

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt `TRUE` zurück, wenn das Zurücksetzen erfolgreich war.

5.1.2 System

5.1.2.1 FB_ALY_DateTime

Funktionsbaustein zum Speichern und Verarbeiten von Zeitstempeln. Zeitstempel werden als Rohwerte mit einer Genauigkeit von 1 ns gespeichert. Mit Hilfe der Methoden dieses FB kann die Zeitspanne geändert, verglichen oder formatiert werden.

Methoden

Name	Definitionsort	Beschreibung
AddRaw	Local	Methode zum Addieren einer Zeitspanne im Rohformat.
AddTimespan	Local	Methode zum Addieren einer in FB_ALY_Timespan [▶ 324] gespeicherten Zeitspanne.
SubRaw	Local	Methode zum Subtrahieren einer Zeitspanne im Rohformat.
SubTimespan	Local	Methode zum Subtrahieren einer in FB_ALY_Timespan [▶ 324] gespeicherten Zeitspanne.
EqualsTo	Local	Methode für den Vergleich von zwei Zeitstempeln.
ToString	Local	Methode für die Formatierung des aktuellen Rohwerts als String.

Beispiel

```
VAR
    fbDateTime : FB_ALY_DateTime;
    fbSystemTime : FB_ALY_GetSystemTime;
    sFormattedDateTime : STRING(29);
END_VAR
```

```
// Get current system time
fbSystemTime.Call();
```

```
fbDateTime.nRaw := fbSystemTime.tSystemTime;
fbDateTime.AddRaw(TO_LINT(LTIME#1H));
sFormattedDateTime := fbDateTime.ToString()
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.2.1.1 AddRaw

Aufruf zur Addition einer Zeitspanne im Rohformat zum Zeitstempelwert.

Syntax

Definition:

```
METHOD AddRaw : ULINT
VAR_INPUT
    nRaw : LINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nRaw	LINT	Rohwert, der zum Zeitstempel addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
AddRaw	ULINT	Gibt den resultierenden Zeitstempelwert zurück.

5.1.2.1.2 AddTimespan

Aufruf zur Addition einer Zeitspanne zum Zeitstempelwert.

Syntax

Definition:

```
METHOD AddTimespan : ULINT
VAR_INPUT
    ipTimespan : I_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipTimespan	I_ALY_Timespan	Schnittstellenzeiger auf eine Timespan-Instanz, die zum Zeitstempel addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
AddTimespan	ULINT	Gibt den resultierenden Zeitstempelwert zurück.

5.1.2.1.3 SubRaw

Aufruf zur Subtraktion einer Zeitspanne im Rohformat vom Zeitstempelwert.

Syntax

Definition:

```
METHOD SubRaw : BOOL
VAR_INPUT
    nRaw : LINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nRaw	LINT	Rohwert, der vom Zeitstempel subtrahiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
SubRaw	ULINT	Gibt den resultierenden Zeitstempelwert zurück.

5.1.2.1.4 SubTimespan

Aufruf zur Subtraktion einer Zeitspanne vom Zeitstempelwert.

Syntax

Definition:

```
METHOD SubTimespan : ULINT
VAR_INPUT
    ipTimespan : I_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipTimespan	I_ALY_Timespan	Schnittstellenzeiger auf eine Timespan-Instanz, die zum Zeitstempel addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
SubTimespan	ULINT	Gibt den resultierenden Zeitstempelwert zurück.

5.1.2.1.5 EqualsTo

Aufruf zum Vergleich des aktuellen Zeitstempels mit dem Eingangszeitstempel.

Syntax

Definition:

```
METHOD EqualsTo : BOOL
VAR_INPUT
    ipDateTime: I_ALY_DateTime;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipDateTime	I_ALY_DateTime	Schnittstellenzeiger auf eine DateTime-Instanz für den Vergleich.

Rückgabewert

Name	Typ	Beschreibung
EqualsTo	BOOL	Gibt TRUE zurück, wenn die Zeitstempel gleich sind.

5.1.2.1.6 ToString

Methode für die Formatierung des aktuellen Rohwerts als String. Das Ausgabeformat ist JJJJ-MM-TT-hh:mm:ss.nnnnnnnnn.

Rückgabewert

Name	Typ	Beschreibung
ToString	STRING(29)	Gibt den formatierten String zurück.

5.1.2.2 FB_ALY_GetSystemTime

Ruft die lokale Systemzeit mit einer Auflösung von 1 ns ab. Die Genauigkeit dieser Zeit ist 100 ns. Der Wert wird durch die Call()-Methode aktualisiert.

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_DateTime
    tSystemTime : ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
tSystemTime	ULINT	Aktuelle Systemzeit.

Methoden

Name	Definitionsort	Beschreibung
Call()	Local	Methode zur Erfassung der aktuellen Systemzeit.

Beispiel

```
VAR
    fbSystemTime : FB_ALY_GetSystemTime;
    tSystemTime : ULINT;
END_VAR
```

```
// Gather current system time
fbSystemTime.Call();

// Allocate system time
tSystemTime := fbSystemTime.tSystemTime;
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.2.2.1 Call

Aufruf zur Erfassung der aktuellen Systemzeit. Die Systemzeit wird in den Ausgangsdaten des FB gespeichert.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt `TRUE` zurück, wenn erfolgreich.

5.1.2.3 FB_ALY_Timespan

Funktionsbaustein zum Speichern und Verarbeiten von Zeitspannen. Zeitspannen werden als Rohwerte mit einer Genauigkeit von 1 ns gespeichert. Mit Hilfe der Methoden dieses FB kann die Zeitspanne geändert, verglichen oder formatiert werden.

Methoden

Name	Definitionsort	Beschreibung
AddRaw	Local	Methode zum Addieren einer Zeitspanne im Rohformat.
AddTimespan	Local	Methode zum Addieren einer in FB_ALY_Timespan [▶ 324] gespeicherten Zeitspanne.
SubRaw	Local	Methode zum Subtrahieren einer Zeitspanne im Rohformat.
SubTimespan	Local	Methode zum Subtrahieren einer in FB_ALY_Timespan [▶ 324] gespeicherten Zeitspanne.
EqualsTo	Local	Methode für den Vergleich von zwei Zeitstempeln.
ToFormatString	Local	Methode für die Formatierung des aktuellen Rohwerts als String.
ToString	Local	Methode für die Formatierung des aktuellen Rohwerts als String mit konfigurierbarer Genauigkeit.
TotalDays	Local	Methode zum Abrufen der Gesamtanzahl der Tage.
TotalHours	Local	Methode zum Abrufen der Gesamtanzahl der Stunden.
TotalMinutes	Local	Methode zum Abrufen der Gesamtanzahl der Minuten.
TotalSeconds	Local	Methode zum Abrufen der Gesamtanzahl der Sekunden.

Beispiel

```
VAR
    fbTimespan : FB_ALY_Timespan;
    sTimespan : STRING;
    sFormattedTimespan : STRING;
END_VAR
```

```
fbTimespan.AddRaw(TO_LINT(LTIME#1S));
sTimespan := fbTimespan.ToString(eAccuracy := E_ALY_TimestampAccuracy.Second);
sFormattedTimespan := fbTimespan.ToFormatString();
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.2.3.1 AddRaw

Aufruf zur Addition einer Zeitspanne im Rohformat zum Zeitspannenwert.

Syntax

Definition:

```
METHOD AddRaw : ULINT
VAR_INPUT
    nRaw : LINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nRaw	LINT	Rohwert, der zu der Zeitspanne addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
AddRaw	ULINT	Gibt den resultierenden Zeitspannenwert zurück.

5.1.2.3.2 AddTimespan

Aufruf zur Addition einer Zeitspanne zum Zeitspannenwert.

Syntax

Definition:

```
METHOD AddTimespan : ULINT
VAR_INPUT
    ipTimespan : I_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipTimespan	I_ALY_Timespan	Schnittstellenzeiger auf eine Timespan-Instanz, die zur Zeitspanne addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
AddTimespan	ULINT	Gibt den resultierenden Zeitspannenwert zurück.

5.1.2.3.3 SubRaw

Aufruf zur Subtraktion einer Zeitspanne im Rohformat vom Zeitspannenwert.

Syntax

Definition:

```
METHOD SubRaw : BOOL
VAR_INPUT
    nRaw : LINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nRaw	LINT	Rohwert, der von der Zeitspanne subtrahiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
SubRaw	ULINT	Gibt den resultierenden Zeitspannenwert zurück.

5.1.2.3.4 SubTimespan

Aufruf zur Subtraktion einer Zeitspanne vom Zeitspannenwert.

Syntax

Definition:

```
METHOD SubTimespan : ULINT
VAR_INPUT
    ipTimespan : I_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipTimespan	I_ALY_Timespan	Schnittstellenzeiger auf eine Timespan-Instanz, die zur Zeitspanne addiert werden soll.

Rückgabewert

Name	Typ	Beschreibung
SubTimespan	ULINT	Gibt den resultierenden Zeitspannenwert zurück.

5.1.2.3.5 EqualsTo

Aufruf zum Vergleich der aktuellen Zeitspanne mit der Eingangszeitspanne.

Syntax

Definition:

```
METHOD EqualsTo : BOOL
VAR_INPUT
    ipTimespan : I_ALY_Timespan;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
ipTimespan	I_ALY_Timespan	Schnittstellenzeiger auf eine Timespan-Instanz für den Vergleich.

Rückgabewert

Name	Typ	Beschreibung
EqualsTo	BOOL	Gibt TRUE zurück, wenn die Zeitspannen gleich sind.

5.1.2.3.6 ToString

Methode für die Formatierung des aktuellen Rohwerts als String mit konfigurierbarer Genauigkeit. Ein Ausgangsbeispiel ist 1d2m3s4ms5ns.

Eingänge

Name	Typ	Beschreibung
eAccuracy	E_ALY_TimestampAccuracy [▶ 358]	Definieren der Genauigkeit des Ausgangsstrings.

Rückgabewert

Name	Typ	Beschreibung
ToString	STRING	Gibt den formatierten String zurück.

5.1.2.3.7 ToFormatString

Methode für die Formatierung des aktuellen Rohwerts als String. Das Ausgabeformat ist tt:hh:mm:ss,f.

Rückgabewert

Name	Typ	Beschreibung
ToFormatString	STRING	Gibt den formatierten String zurück.

5.1.2.3.8 TotalDays

Methode zum Abrufen der Gesamtanzahl der Tage.

Rückgabewert

Name	Typ	Beschreibung
TotalDays	DINT	Gibt die Gesamtanzahl der Tage zurück.

5.1.2.3.9 TotalHours

Methode zum Abrufen der Gesamtanzahl der Stunden.

Rückgabewert

Name	Typ	Beschreibung
TotalHours	DINT	Gibt die Gesamtanzahl der Stunden zurück.

5.1.2.3.10 TotalMinutes

Methode zum Abrufen der Gesamtanzahl der Minuten.

Rückgabewert

Name	Typ	Beschreibung
TotalMinutes	DINT	Gibt die Gesamtanzahl der Minuten zurück.

5.1.2.3.11 TotalSeconds

Methode zum Abrufen der Gesamtanzahl der Sekunden.

Rückgabewert

Name	Typ	Beschreibung
TotalSeconds	LINT	Gibt die Gesamtanzahl der Sekunden zurück.

5.1.3 IoT

5.1.3.1 IoT Symbol

5.1.3.1.1 FB_ALY_IotSymbol_BOOL

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_BOOL
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.2 FB_ALY_IotSymbol_BYTE

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_BYTE
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.3 FB_ALY_IotSymbol_DINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_DINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.4 FB_ALY_IotSymbol_DWORD

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_DWORD
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.5 FB_ALY_IotSymbol_INT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_INT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.6 FB_ALY_IotSymbol_LINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_LINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.7 FB_ALY_IotSymbol_LREAL

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_LREAL
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.8 FB_ALY_IotSymbol_LWORD

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_LWORD
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.9 FB_ALY_IotSymbol_REAL

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_REAL
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.10 FB_ALY_IotSymbol_SINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_SINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.11 FB_ALY_IotSymbol_STRING

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_STRING
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.12 FB_ALY_IotSymbol_UDINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_UDINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.13 FB_ALY_IotSymbol_UINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_UINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.14 FB_ALY_IotSymbol_ULINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_ULINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.15 FB_ALY_IotSymbol_USINT

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_USINT
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.1.16 FB_ALY_IotSymbol_WORD

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_IotSymbol_WORD
VAR_INPUT
    stConfig : ST_ALY_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.2 FB_ALY_StreamHelper

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StreamHelper
VAR_INPUT
    stConfig : ST_ALY_StreamHelper_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bConnected: BOOL;
    sStream: STRING(255);
    nNumIotSymbolsRegistered: UDINT;
    tCycleTime: LTIME;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALY_IotSymbol_Config	Struktur für die Konfiguration des FB

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bConnected	BOOL	TRUE, wenn StreamHelper mit einem Message-Broker verbunden ist.
sStream	STRING(255)	Thema/Stream
nNumIotSymbolsRegistered	UDINT	Anzahl der registrierten IoT-Symbole.
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nNumElements	UDINT	Anzahl der eingegangenen Elemente seit dem letzten Aufruf.

Methoden

Name	Definitionsort	Beschreibung
Call	Local	
Configure	Local	
GetTimestampElement	Local	
AddIotSymbol	Local	
ContainsIotSymbol	Local	
ReleaseAllIotSymbols	Local	
ReleaseIotSymbol	Local	

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.3 BinaryStream

5.1.3.3.1 FB_ALY_BinaryStream_File

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_StreamHelper
VAR_INPUT
    nCycleTime: UDINT;
    stSystemID: GUID;
    sPath: T_MaxString;
    nMaxFileSize: UINT;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipResultMessage: I_TcMessage;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCycleTime	UDINT	Aufzeichnung der Zykluszeit
stSystemID	GUID	TwinCAT System-ID
sPath	T_MaxString	Ordnerpfad zur Anmeldung
nMaxFileSize	UINT	Max. Dateigröße

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Eventlogger

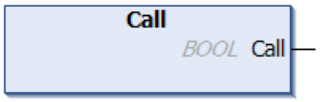
Methoden

Name	Definitionsort	Beschreibung
Call [▶ 346]	Local	Methode für Hintergrundkommunikation mit dem TwinCAT-Treiber. Die Methode muss zyklisch aufgerufen werden.
Close [▶ 346]	Local	Schließen der geöffneten tay-Datei..
CreateDescriptionFile [▶ 346]	Local	Erstellen einer *.tad-Datei für die Streambeschreibung.
CreateSymbolsFile [▶ 347]	Local	Erstellen einer *.tas-Datei für die Symbolbeschreibung.
WriteData [▶ 347]	Local	Schreiben der Daten in die *.tay-Datei.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.3.1.1 Call



Syntax

```
METHOD Call : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	

5.1.3.3.1.2 Close

Syntax

```
METHOD Close : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
Close	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.1.3 CreateDescriptionFile

Syntax

```
METHOD CreateDescriptionFile : BOOL
VAR_INPUT
    nDataSize: UDINT;
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nDataSize	UDINT	Größe von einem Beispiel
sLayout	GUID	Layout-GUID, erstellt durch CreateSymbolic.

Rückgabewert

Name	Typ	Beschreibung
CreateDescriptionFile	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.1.4 CreateSymbolsFile

Syntax

```
METHOD CreateSymbolsFile : BOOL
VAR_INPUT
    pSymbol: PVOID;
    nSymbolSize: UDINT;
    sDataType: STRING;
END_VAR
VAR_OUTPUT
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pSymbol	PVOID	Zeiger auf ADS-Symbol
nSymbolSize	UDINT	Größe des ADS-Symbols
sDataType	STRING	Name des Datentyps des ADS-Symbols

Ausgänge

Name	Typ	Beschreibung
sLayout	GUID	Layout-Hash der Symbolbeschreibung

Rückgabewert

Name	Typ	Beschreibung
CreateSymbolsFile	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.1.5 WriteData

Syntax

```
METHOD WriteData : BOOL
VAR_INPUT
    pData: PVOID;
    nDataSize: UDINT;
    nDataCount: INT;
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pData	PVOID	Zeiger auf Beispiel-Array
nDataSize	UDINT	Größe des Beispiel-Arrays
nDataCount	INT	Anzahl der Beispiel-Array-Elemente
sLayout	GUID	Symbolik-Layout

Rückgabewert

Name	Typ	Beschreibung
WriteData	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.2 FB_ALY_BinaryStream_Mqtt

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALY_BinaryStream_Mqtt
VAR_INPUT
    nCycleTime: UDINT;
    stSystemID: GUID;
    sStreamTopic: T_MaxString;
    stMqttConnSettings: ST_MqttConnectionSettings;
END_VAR
VAR_OUTPUT
    bError: BOOL;
    ipResultMessage: I_TcMessage;
    eConnectionState: ETcIotMqttClientState;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nCycleTime	UDINT	Aufzeichnung der Zykluszeit
stSystemID	GUID	TwinCAT System-ID
sStreamTopic	T_MaxString	Topic, zu dem gestreamt wird
stMqttConnSettings	ST_MqttConnectionSettings	Einstellungen für die MQTT-Verbindung mit dem Message-Broker

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Eventlogger
eConnectionState	ETcIotMqttClientState	Gibt den Status der Verbindung zwischen Client und Broker als Aufzählung ETcIotMqttClientState an.

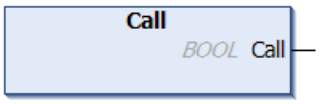
Methoden

Name	Definitionsort	Beschreibung
Call [▶ 349]	Local	Methode für Hintergrundkommunikation mit dem TwinCAT-Treiber. Die Methode muss zyklisch aufgerufen werden.
CloseStream [▶ 349]	Local	Schließen des Analytics-Binärstroms
CreateDescription [▶ 349]	Local	Erstellen der Binärstrombeschreibung
CreateSymbolic [▶ 350]	Local	Erstellen der Binärstromsymbolik
SendData [▶ 350]	Local	Senden der Binärstromdaten

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

5.1.3.3.2.1 Call



Syntax

```
METHOD Call : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	

5.1.3.3.2.2 CloseStream

Syntax

```
METHOD CloseStream : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
CloseStream	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.2.3 CreateDescription

Syntax

```
METHOD CreateDescription : BOOL
VAR_INPUT
    nDataSize: UDINT;
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nDataSize	UDINT	Größe von einem Beispiel
sLayout	GUID	Layout-GUID, erstellt durch CreateSymbolic

Rückgabewert

Name	Typ	Beschreibung
CreateDescription	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.2.4 CreateSymbolic

Syntax

```
METHOD CreateSymbolic : BOOL
VAR_INPUT
    pSymbol: PVOID;
    nSymbolSize: UDINT;
    sDataType: STRING;
END_VAR
VAR_OUTPUT
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pSymbol	PVOID	Zeiger auf ADS-Symbol
nSymbolSize	UDINT	Größe des ADS-Symbols
sDataType	STRING	Name des Datentyps des ADS-Symbols

Ausgänge

Name	Typ	Beschreibung
sLayout	GUID	Layout-Hash der Symbolbeschreibung

Rückgabewert

Name	Typ	Beschreibung
CreateSymbolic	BOOL	TRUE, wenn es geschehen ist.

5.1.3.3.2.5 SendData

Syntax

```
METHOD SendData : BOOL
VAR_INPUT
    pData: PVOID;
    nDataSize: UDINT;
    nDataCount: INT;
    sLayout: GUID;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pData	PVOID	Zeiger auf Beispiel-Array
nDataSize	UDINT	Größe des Beispiel-Arrays
nDataCount	INT	Anzahl der Beispiel-Array-Elemente
sLayout	GUID	Symbolik-Layout

Rückgabewert

Name	Typ	Beschreibung
SendData	BOOL	TRUE, wenn es geschehen ist.

5.2 Funktionen

5.2.1 F_RawTimespan_TO_Structured

Funktion zur Umwandlung eines rohen Zeitspanneneingangs in einen Ausgang vom Typ ST_ALY_Timespan [▶ 358].

Eingänge

Name	Typ	Beschreibung
tTimespan	LINT	Umzuwandelnde Zeitspanne.

Rückgabewert

Name	Typ	Beschreibung
F_RawTimespan_TO_Structured	ST_ALY_Timespan [▶ 358]	Gibt die strukturierte Zeitspanne zurück.

5.2.2 F_StructuredTimespan_TO_Raw

Funktion zur Umwandlung eines Eingangs vom Typ ST_ALY_Timespan [▶ 358] in eine rohe Zeitspanne.

Eingänge

Name	Typ	Beschreibung
stTimespan	ST_ALY_Timespan [▶ 358]	Umzuwandelnde strukturierte Zeitspanne.

Rückgabewert

Name	Typ	Beschreibung
F_StructuredTimespan_TO_Raw	LINT	Gibt die rohe Zeitspanne zurück.

5.3 Datentypen

5.3.1 Allgemeines

5.3.1.1 ST_ALY_Threshold

```
TYPE ST_ALY_Threshold :
STRUCT
    eComparisonOperator : E_ALY_ComparisonOperator;
    fThreshold          : LREAL;
END_STRUCT
END_TYPE
```

5.3.1.2 ST_ALY_ThresholdOnOff

```
TYPE ST_ALY_ThresholdOnOff :
STRUCT
    On  : ST_ALY_Threshold;
    Off : ST_ALY_Threshold;
END_STRUCT
END_TYPE
```

5.3.1.3 ST_ALY_XyPosition

```
TYPE ST_ALY_XyPosition :
STRUCT
    X : LREAL;
    Y : LREAL;
END_STRUCT
END_TYPE
```

5.3.1.4 E_ALY_Classification_2Cls

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_Classification_2Cls :
(
    NotInitialized := 0,
    OK : =1,
    NOK := 2
) := NotInitialized;
END_TYPE
```

5.3.1.5 E_ALY_Classification_3Cls

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_Classification_3Cls :
(
    NotInitialized := 0,
    OK := 1,
    Warning := 2,
    Alarm := 3
) := NotInitialized;
END_TYPE
```

5.3.1.6 E_ALY_Classification_Bounds

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_Classification_Bounds :
(
    NotInitialized := 0,
    WithinBounds := 1,
    Smaller := 2,
    Bigger := 3
```

```
) := NotInitialized;
END_TYPE
```

5.3.2 Config

5.3.2.1 ST_ALY_IotSymbolString_Config

```
TYPE ST_ALY_IotSymbolString_Config :
STRUCT
    sSymbolName         : STRING(255) := '';       // Symbol name (e.g. MAIN.bOn)
    nStringSize         : UDINT := SIZEOF(STRING); // String lenth plus null termination
    nOversamplingFactor : UDINT := 1;              // Oversampling factor
    nArrayElements      : UDINT := 1;              // Number of array elements
END_STRUCT
END_TYPE
```

5.3.2.2 ST_ALY_IotSymbol_Config

```
TYPE ST_ALY_IotSymbol_Config :
STRUCT
    sSymbolName         : STRING(255) := ''; // Symbol name (e.g. MAIN.bOn)
    nOversamplingFactor : UDINT := 1;        // Oversampling factor
    nArrayElements      : UDINT := 1;        // Number of array elements
END_STRUCT
END_TYPE
```

5.3.2.3 ST_ALY_StreamHelper_Config

```
TYPE ST_ALY_StreamHelper_Config :
STRUCT
    nObjectID       : OTCID := 0;  // Object ID of referenced StreamHelper
    nNumInputBuffer : UDINT := 20; // Number of input buffer to be reserved for every symbol
END_STRUCT
END_TYPE
```

5.3.2.4 E_ALY_BandMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_BandMode :
(
    NotInitialized := 0,
    Absolute := 1,
    Relative := 2,
) := NotInitialized;
END_TYPE
```

5.3.2.5 E_ALY_CascadeStartupBehaviour

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_CascadeStartupBehaviour :
(
    NotInitialized := 0,
    WaitUntilFilled := 1,
    UsePreviousCascadeValue := 2,
) := NotInitialized;
END_TYPE
```

5.3.2.6 E_ALY_ComparisonOperator

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_ComparisonOperator :
(
    NotInitialized := 0,
    GreaterThan := 1,
```

```
    GreaterThanOrEqualTo := 2,
    LessThan := 3,
    LessThanOrEqualTo := 4,
    Equals := 5,
    NotEqualTo := 6
) := NotInitialized;
END_TYPE
```

5.3.2.7 E_ALY_CountMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_CountMode :
(
    NotInitialized := 0,
    Cyclic := 1,
    OnChange := 2,
) := NotInitialized;
END_TYPE
```

5.3.2.8 E_ALY_DayOfWeekMask

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_DayOfWeekMask :
(
    None := 0,
    Sunday := 1,
    Monday := 2,
    Tuesday := 4,
    Wednesday := 8,
    Thursday := 16,
    Friday := 32,
    Saturday := 64,
    EveryDay := 127,
    MondayToFriday := 62,
    Weekend := 65
) := NotInitialized;
END_TYPE
```

5.3.2.9 E_ALY_IntegrationMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_IntegrationMode :
(
    NotInitialized := 0,
    Direct := 1,
    Absolute := 2,
) := NotInitialized;
END_TYPE
```

5.3.2.10 E_ALY_LogicOperator

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_LogicOperator :
(
    NotInitialized := 0,
    AND_ := 1,
    OR_ := 2,
    NAND_ := 3,
    NOR_ := 4,
    XOR_ := 5
) := NotInitialized;
END_TYPE
```

5.3.2.11 E_ALY_MathOperator

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_MathOperator :
(
    NotInitialized := 0,
    Addition := 1,
    Subtraction := 2,
    Multiplication := 3,
    Division := 4,
    PowerOf := 5,
    Modulo := 6,
) := NotInitialized;
END_TYPE
```

5.3.2.12 E_ALY_MovingAvgStartupBehaviour

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_MovingAvgStartupBehaviour :
(
    NotInitialized := 0, (* Not initialized *)
    ZeroPadding := ,     (* All buffers are initialized with zero. *)
    UseFirstValue := 2,  (* Use first occuring value to initialize buffers. *)
    WaitUntilFilled := 3,(* Wait with calculation until filled. *)
    AvgOverExisting := 4 (* Average over existing elements. *)
) := NotInitialized;
END_TYPE
```

5.3.2.13 E_ALY_ReadState

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_ReadState :
(
    Idle := 0,
    Read := 1,
    Pending := 2,
    PendingRead := 3,
    PendingIdle := 4
) := Idle;
END_TYPE
```

5.3.2.14 E_ALY_StateHistMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_StateHistMode :
(
    NotInitialized := 0,
    Absolute := 1,
    Relative := 2
) := NotInitialized ;
END_TYPE
```

5.3.2.15 E_ALY_StringCompareMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_StringCompareMode :
(
    NotInitialized := 0,
    Equals := 1,
    BeginsWith := 2,
    Contains := 3
) := NotInitialized;
END_TYPE
```

5.3.2.16 E_ALY_TeachMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_TeachMode :
(
    NotInitialized := 0,
    Minimum := 1,
    Maximum := 2,
    Mean := 3,
) := NotInitialized;
END_TYPE
```

5.3.2.17 E_ALY_TeachState

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_TeachState :
(
    Idle := 0,
    Teach := 1,
    Pending := 2,
    PendingTeach := 3,
    PendingIdle := 4
) := Idle;
END_TYPE
```

5.3.2.18 E_ALY_TimerMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_TimerMode :
(
    NotInitialized := 0, (* Not initialized *)
    TON := 1,            (* Switch-on delay *)
    TOF := 2,            (* Switch-off delay *)
    TP := 3              (* Pulse *)
) := NotInitialized;
END_TYPE
```

5.3.2.19 E_ALY_GateMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_LogicOperator :
(
    NotInitialized:=0,
    IntersectGate:=1,
    NotIntersectGate:=2,
    IntersectProjection:=3,
    NotIntersectProjection:=4,
    IntersectGateOrProjection:=5
) := NotInitialized;
END_TYPE
```

5.3.2.20 E_ALY_CorrelationMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_CorrelationMode :
(
    NotInitialized := 0,
    Base := 1,
    Normed := 2,
    Covariance := 3,
    CovarianceBesel :=4,
    Pearson :=5
) := NotInitialized;
END_TYPE
```

5.3.2.21 E_ALY_WindowMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_WindowMode :
(
    NotInitialized := 0,
    SlidingWindow := 1,
    FixWindow := 2,
    Continuous := 3
) := NotInitialized;
END_TYPE
```

5.3.2.22 E_ALY_KMeansInitMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_KMeansInitMode :
(
    NotInitialized := 0,
    Random := 1,
    Equidistant := 2,
    Values := 3,
) := NotInitialized;
END_TYPE
```

5.3.2.23 E_ALY_FunctionType

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_FunctionType :
(
    NotInitialized := 0,
    Constt := 1,
    Sine := 2,
    Triangle := 3,
    Rectangle := 4,
    Sawtooth := 5,
) := NotInitialized;
END_TYPE
```

5.3.2.24 E_ALY_ConfigState

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_ConfigState :
(
    NotConfigured := 0,
    Pending := 1,
    Configured := 2,
) := NotConfigured ;
END_TYPE
```

5.3.2.25 E_ALY_HistMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_HistMode :
(
    NotInitialized := 0,
    Absolute := 1,
    Relative := 2
) := NotInitialized ;
END_TYPE
```

5.3.3 Time

5.3.3.1 ST_ALY_Timespan

```
TYPE ST_ALY_Timespan :
STRUCT
    bIsNegative   : BOOL;
    wDays         : WORD;
    wHours        : WORD;
    wMinutes      : WORD;
    wSeconds      : WORD;
    wMilliseconds : WORD;
    wMicroseconds : WORD;
    wNanoseconds  : WORD;
END_STRUCT
END_TYPE
```

5.3.3.2 E_ALY_TimestampAccuracy

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_ALY_TimestampAccuracy :
(
    Day,
    Hour,
    Minute,
    Second,
    Millisecond,
    Microsecond,
    Nanosecond
);
END_TYPE
```

6 Beispiele

6.1 Kombination von Algorithmen mit lokalen Eingängen

Das Beispiel zeigt, wie Analysealgorithmen einbezogen und kombiniert werden. Es werden Algorithmen vom Typ FB_ALY_EdgeCounter_1Ch [▶ 31], FB_ALY_MathOperation_1Ch [▶ 206] und FB_ALY_MinMaxAvg_1Ch [▶ 72] verwendet.

Das Beispiel steht hier zum Download bereit:

https://infosys.beckhoff.com/content/1031/tf3500_tc3_analytics_library/Resources/6917579531.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x64, x86)	Tc3_Analytics

7 Anhang

7.1 Returncodes

Die folgende Liste enthält Returncodes der TwinCAT Analytics Library.

Ereignis-ID (Hex)	Ereignistext
16#1001	Fehler während der Initialisierung. Kein Routerspeicher vorhanden. Größe des Routerspeichers prüfen.
16#1002	Fehler im Übergang PREOP->SAFEOP
16#1003	Fehler im Übergang SAFEOP->OP
16#1004	Fehler im Übergang SAFEOP->OP: Kein Task zugeordnet. Modul wird nicht zyklisch ausgeführt.
16#1005	Fehler im Übergang OP->SAFEOP
16#1006	Fehler im Übergang SAFEOP->PREOP
16#1007	Fehler während der Initialisierung. Ungültige Objekt-ID.
16#1008	Fehler während der Initialisierung. Ungültiger Symbolname.
16#1009	Fehler während der Initialisierung. Ungültige Symbolgröße.
16#100A	Fehler während der Initialisierung. Ungültige Anzahl von Puffern.
16#100B	Fehler während der Initialisierung. Ungültige Datenbereichsnummer.
16#100C	Fehler während der Initialisierung. Ungültige Anzahl von Kanälen.
16#2001	Fehler während der Konfiguration. Kein Routerspeicher vorhanden. Größe des Routerspeichers prüfen.
16#2002	Fehler während der Konfiguration. Ein Null-Zeiger wurde zugeordnet.
16#2003	Der konfigurierte Datentyp ist nicht gültig.
16#2004	Der konfigurierte Zählmodus ist nicht gültig.
16#2005	Der konfigurierte Integrationsmodus ist nicht gültig.
16#2006	Der konfigurierte Zustandshistogrammmodus ist nicht gültig.
16#2007	Der konfigurierte Vergleichsoperator ist nicht gültig.
16#2008	Der konfigurierte Einlernmodus ist nicht gültig
16#2009	Der konfigurierte logische Operator ist nicht gültig.
16#200A	Der konfigurierte mathematische Operator ist nicht gültig.
16#200B	Das konfigurierte Startverhalten ist nicht gültig.
16#200C	Die konfigurierten Grenzen sind nicht gültig.
16#200D	Die Anzahl der konfigurierten Werte ist nicht gültig.
16#200E	Die konfigurierte Sample Rate ist nicht gültig.
16#200F	Das konfigurierte Intervall ist nicht gültig.
16#2010	Das konfigurierte Intervall ist zu kurz.
16#2011	Die Anzahl der konfigurierten Teilmengen ist nicht gültig.
16#2012	Die Anzahl der konfigurierten Klassen ist nicht gültig.
16#2013	Der konfigurierte Dateipfad ist nicht gültig.
16#2014	Die konfigurierte Segmentgröße ist nicht gültig.
16#2015	Der konfigurierte Timeout ist nicht gültig.
16#2016	Der konfigurierte Bandmodus ist nicht gültig.
16#2017	Die konfigurierte Länge ist nicht gültig
16#2018	Der konfigurierte Timermodus ist nicht gültig.
16#2019	Die konfigurierten Wochentage sind nicht gültig.
16#201A	Die konfigurierte Schwelle ist nicht gültig.
16#201B	Der konfigurierte Stringvergleichsmodus ist nicht gültig.
16#201C	Der konfigurierte Gate-Modus ist nicht gültig.
16#201D	Fehler während der Konfiguration. Ungültiger Symbolname.
16#201E	Fehler während der Konfiguration. Ungültige Symbolgröße.
16#201F	Fehler während der Konfiguration. Ungültige Anzahl von Puffern.
16#2020	Fehler während der Konfiguration. Ungültige Anzahl von Bändern.
16#2021	Fehler während der Konfiguration. Unzulässige Schwellenreihenfolge.

Ereignis-ID (Hex)	Ereignistext
16#2022	Fehler während der Konfiguration. Initialisierung fehlt.
16#3001	Fehler während der Laufzeit. Kein Routerspeicher vorhanden. Größe des Routerspeichers prüfen.
16#3002	Fehler während der Laufzeit. Ein Null-Zeiger wurde zugeordnet.
16#3003	Fehler während der Laufzeit. Ungültige Eingangsgröße.
16#3004	Fehler während der Laufzeit. Ungültige Ausgangsgröße.
16#3005	Fehler während der Laufzeit. Division durch null.
16#3006	Fehler während der Laufzeit. Zeitstempel kann nicht null sein.
16#3007	Fehler während der Laufzeit. Konfiguration fehlt.
16#3008	Fehler während der Laufzeit. Ungültiger Zustand.
16#3009	Fehler während der Laufzeit. Zyklischer Aufrufer ist zugewiesen. Methoden können nicht aufgerufen werden.
16#300A	Fehler während der Laufzeit. Außerhalb des Bereichs.
16#300B	Fehler während der Laufzeit. Ungültige Header-Größe.
16#300C	Fehler während der Laufzeit. Ungültige Datei.
16#300D	Fehler während der Laufzeit. Ungültiges Intervall.
16#300E	Fehler während der Laufzeit. Symbol bereits zugewiesen.
16#300F	Fehler während der Laufzeit. Kein StreamHelper zugewiesen.
16#3010	Fehler während der Laufzeit. Ungültiger Eingangskanal.
16#3011	Fehler während der Laufzeit. Ungültiger Datentyp.
16#3012	Fehler während der Laufzeit. Ungültige Einheit.

7.2 FAQ – Häufig gestellte Fragen und Antworten

In diesem Abschnitt werden häufig gestellte Fragen beantwortet, um Ihnen die Arbeit mit dem TwinCAT Analytics Library zu erleichtern. Falls Sie weitere Fragen haben, wenden Sie sich bitte an unser Support-Team support@beckhoff.com.

Kann ich die Analytics Library unabhängig vom Workflow von TwinCAT Analytics verwenden? [▶ 362]

Ist es möglich, die TE3500 Analytics Workbench mit automatischer Codegenerierung zu verwenden, wenn ich die Analytics Library in meiner lokalen Anwendung nutzen möchte? [▶ 362]

?Kann ich die Analytics Library unabhängig vom Workflow von TwinCAT Analytics verwenden?

!Ja, selbstverständlich. Sie können die Bibliothek als Standard-SPS-Bibliothek für Ihre lokale Maschinenanwendung ohne MQTT-Kommunikation und andere Analytics-Produkte verwenden.

?Ist es möglich, die TE3500 Analytics Workbench mit automatischer Codegenerierung zu verwenden, wenn ich die Analytics Library in meiner lokalen Anwendung nutzen möchte?

!Ja, Sie können im Konfigurator der TE3500 Analytics Workbench im Dialog Deploy Analytics Runtime die Option „Add to existing Solution“ wählen. Dadurch wird der Analytics-Teil Ihrem lokalen Anwendungscode hinzugefügt, indem die richtige Lösung in der Pfadsteuerung verwendet wird. Zudem ist es möglich, aus einer ADS Route Variablen in den Konfigurator einzufügen. In der Codegenerierung ist dann ein Platzhalter, wo Sie das Mapping manuell durchführen müssen.

Mehr Informationen:
www.beckhoff.de/tf3510

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