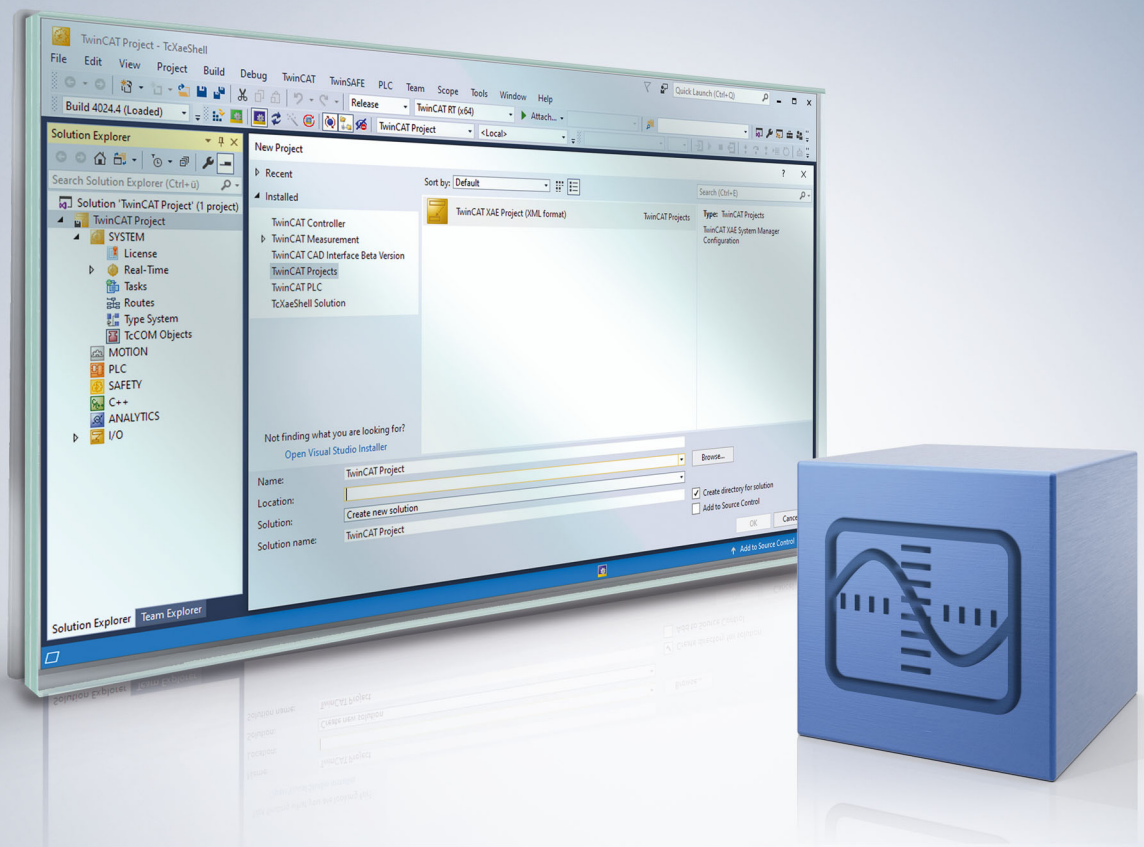
BECKHOFF New Automation Technology

Handbuch | DE

TF3650

TwinCAT 3 | Power Monitoring



31.01.2023 | Version: 1.5

Inhaltsverzeichnis

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®

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

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

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.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

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

Beckhoff bietet in seinem I/O-Portfolio verschiedene Klemmen zur Realisierung einer Energienetzanalyse und -überwachung. Sogenannte Power-Measurement-Klemmen wie die EL3403 oder EL3413 liefern dabei in einem Ein- oder Drei-Phasen-Netz direkt RMS-Werte (Effektivwerte) für Strom und Spannung sowie Wirk-, Blind- und Scheinleistung. Mit den Power-Monitoring-Klemmen EL3773 und EL3783 können sogar die Rohdaten von Strom und Spannung aufgezeichnet werden, sodass Ereignisse im Netz noch genauer detektiert und die Berechnung beliebiger Werte in der Steuerung selbst ausgeführt werden können.

Die Function TC3 Power Monitoring ist eine SPS-Bibliothek zur Auswertung von Strom- und Spannungsrohdaten, die von Power-Monitoring-Klemmen bereitgestellt werden.

Die Bibliothek stellt Funktionsbausteine zur Berechnung von RMS-Werten (Effektivwerten) für Strom, Spannung und Leistung zur Verfügung. Diese können als Momentan- oder Durchschnittswert ausgegeben werden. Zusätzlich stehen am Funktionsbaustein auch Maximal- und Minimalwerte zur Verfügung. Frequenzen und Frequenzspektren können wie die Harmonischen im Netz und deren Belastung in Form der Total Harmonic Distortion (THD) bestimmt werden. Des Weiteren bietet die Bibliothek Funktionsbausteine an, mit denen die Frequenzen und Drehfelder ermittelt werden können.

Produktinformationen

Die aktuelle Version der Power-Monitoring-Bibliothek ist als Download auf der Beckhoff Homepage verfügbar. Die SPS-Bibliothek bietet verschiedene Algorithmen für die Analyse von Strom und Spannung in einem Ein- und Drei-Phasen-Netz. Einige Algorithmen basieren auf Implementierungen der TwinCAT-Condition-Monitoring-Bibliothek.

Produktkomponenten

Das Produkt TF3650 Power Monitoring besteht aus den folgenden Komponenten:

SPS-Bibliotheken	Tc3_PowerMonitoring.compiled-library Tc3_MultiArray.compiled-library
Treiber	TcPowerMonitoring.sys TcMultiArray.sys

3 Installation

3.1 Systemvoraussetzungen

Engineering-System

Ein Engineering-System beschreibt einen Rechner, der für die Entwicklung von Programmcode genutzt wird und keinen Programmcode ausführt. Ein Engineering-System muss die folgenden Voraussetzungen erfüllen:

- TwinCAT 3 XAE (Engineering Installation) Build 4024.0 oder höher
- Installation von TF3650 Power Monitoring
- Für das Engineering kann eine 7-Tage-Trial-Lizenz wiederholbar aktiviert werden (siehe auch Lizenzierung [▶ 12])

Runtime-System

Ein Runtime-System beschreibt einen Industrie- oder Embedded-PC auf dem Programmcode ausgeführt wird. Ein Runtime-System muss die folgenden Voraussetzungen erfüllen:

- TwinCAT 3 XAR (Runtime Installation) Build 4024.0 oder höher
- 32-Bit- und 64-Bit-Systeme werden unterstützt
- Betriebssysteme Win10 und TwinCAT/BSD
- Eine Lizenz für TC1200 PLC und für TF3650 Power Monitoring
- Für Testzwecke kann eine 7-Tage-Trial-Lizenz wiederholbar aktiviert werden.

Engineering und Runtime auf dem gleichen System

Wenn Engineering und Runtime auf dem gleichen System genutzt werden sollen, müssen folgende Systemvoraussetzungen erfüllt sein:

- TwinCAT 3 XAE (Engineering Installation) Build 4024.0 oder höher
- Eine Lizenz für TC1200 PLC und für TF3650 Power Monitoring
- Für Testzwecke kann eine 7-Tage-Trial-Lizenz wiederholbar aktiviert werden.

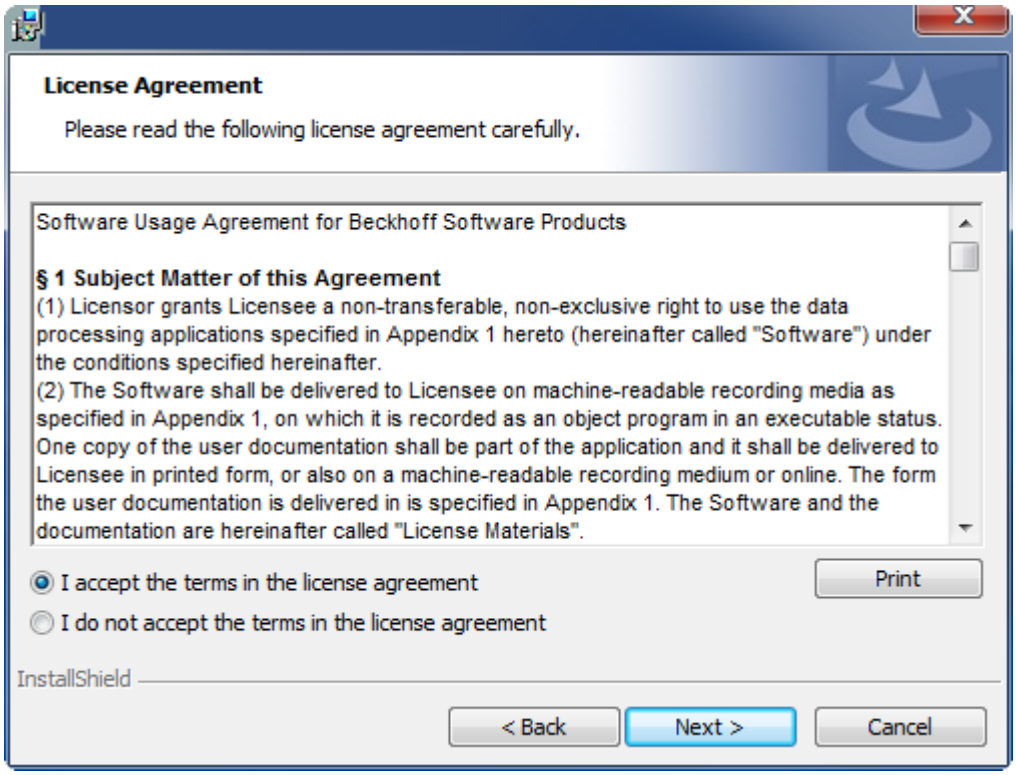
3.2 Installation

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

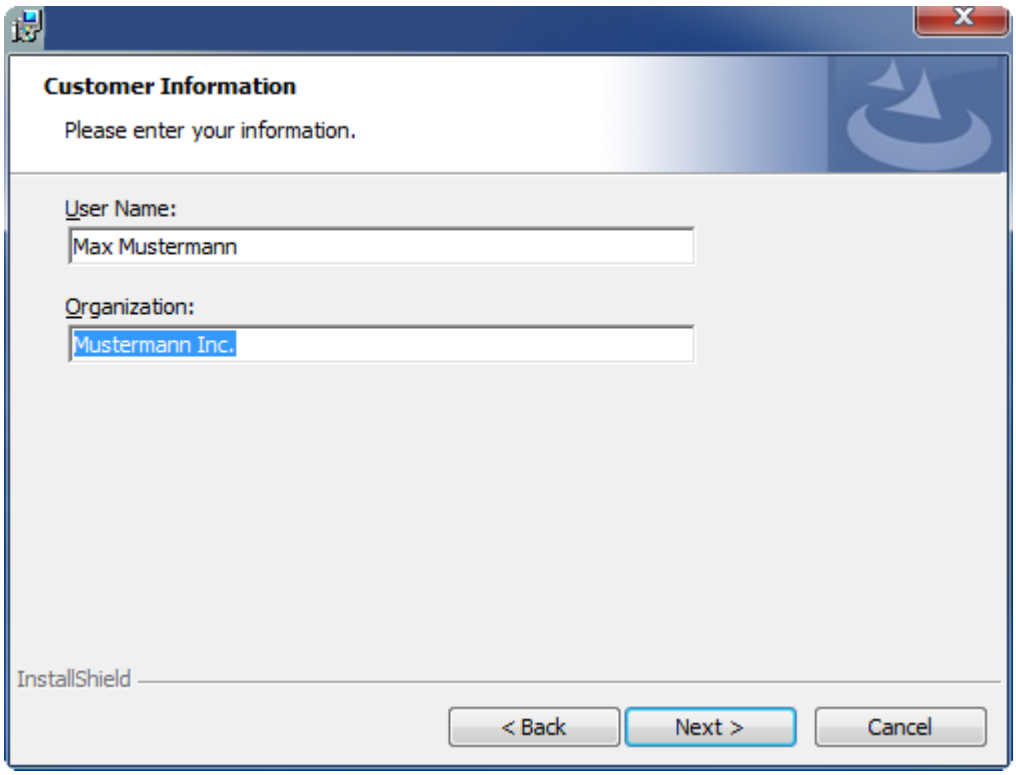
✓ Die Setup-Datei der TwinCAT 3 Function wurde von der Beckhoff-Homepage heruntergeladen.

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.

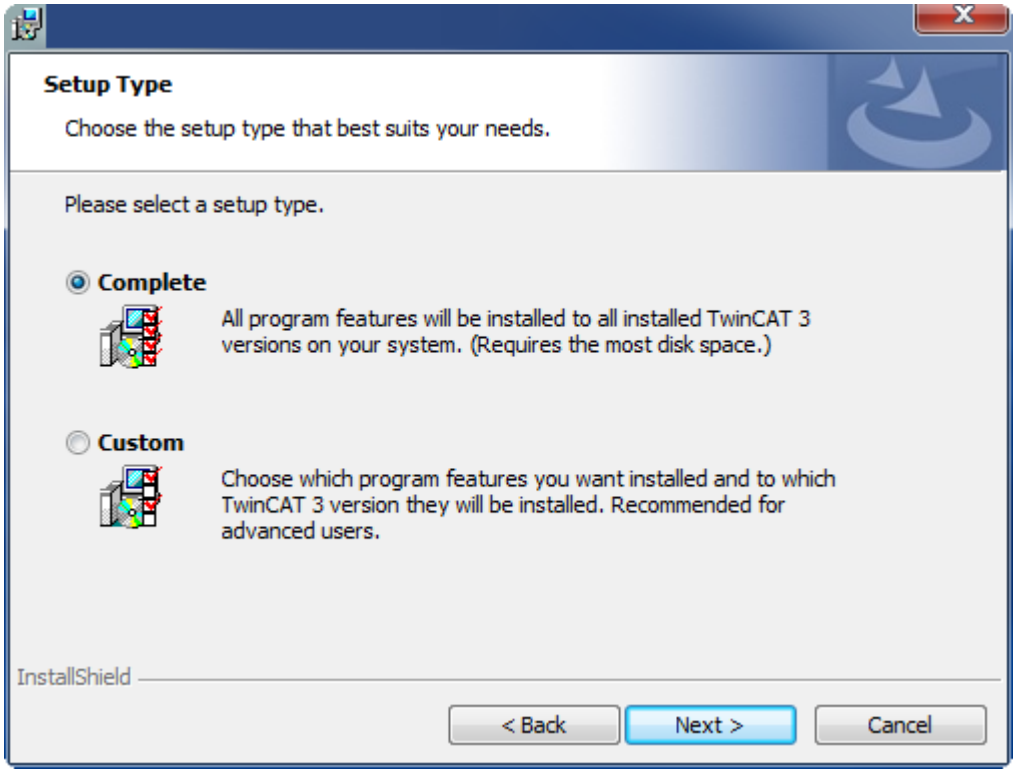
2. Akzeptieren Sie die Endbenutzerbedingungen und klicken Sie auf **Next**.



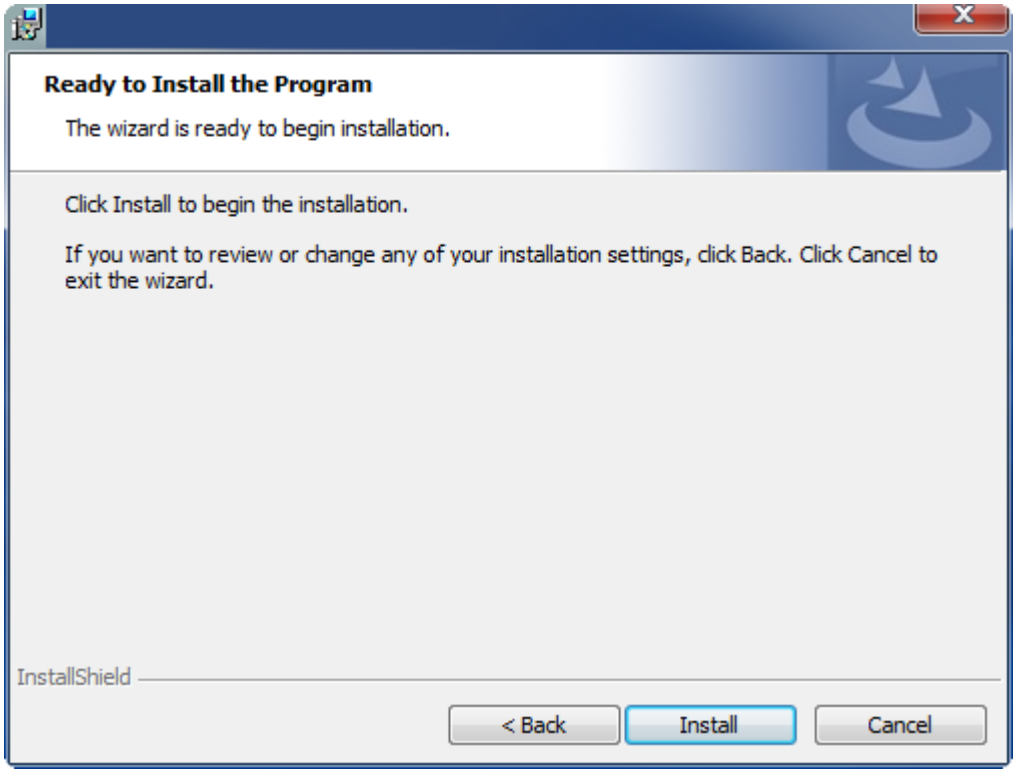
3. Geben Sie Ihre Benutzerdaten ein.



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**.

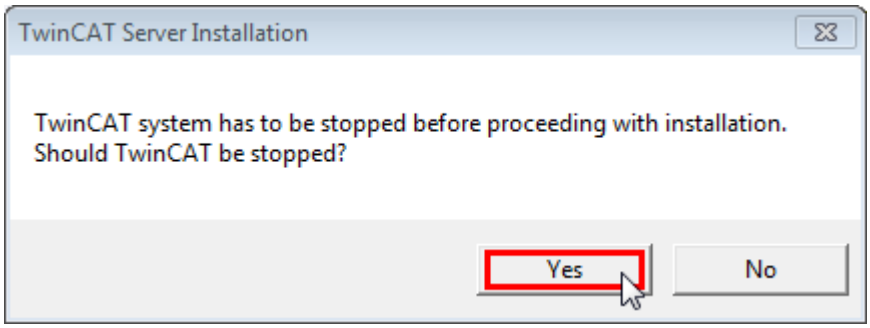


5. Wählen Sie **Next** und anschließend **Install**, um die Installation zu beginnen.

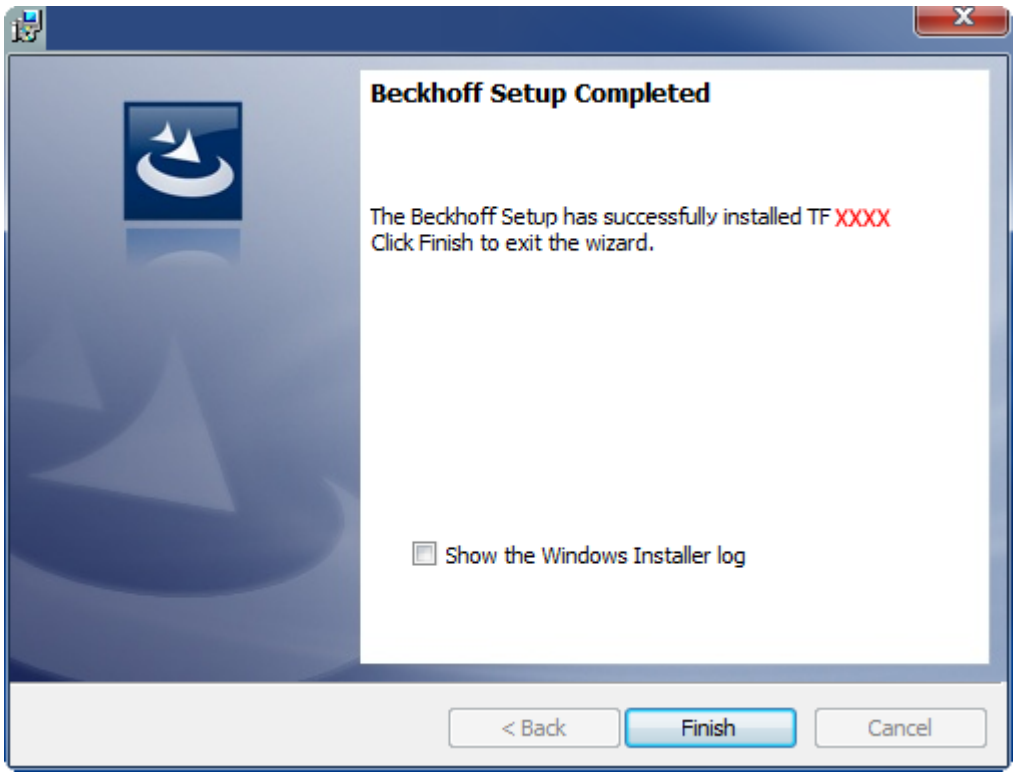


⇨ Ein Dialog weist Sie darauf hin, dass das TwinCAT-System für die weitere Installation gestoppt werden muss.

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 [▶ 12]).

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“.

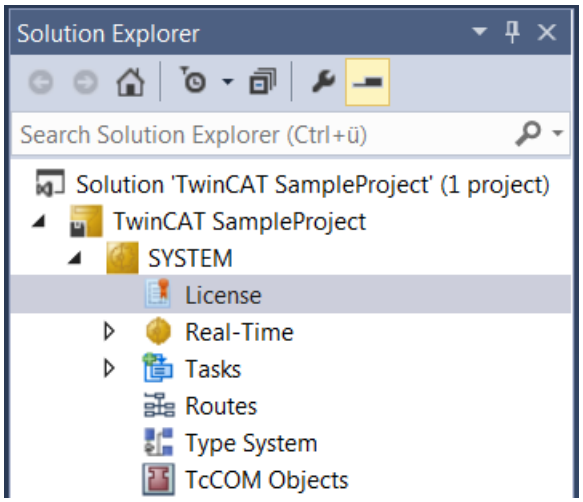
Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function

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

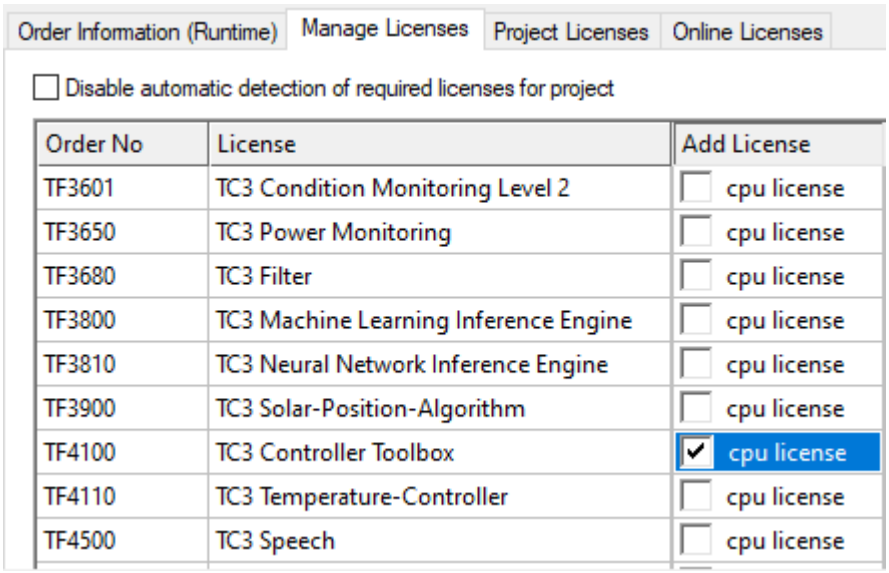
1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).

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.**



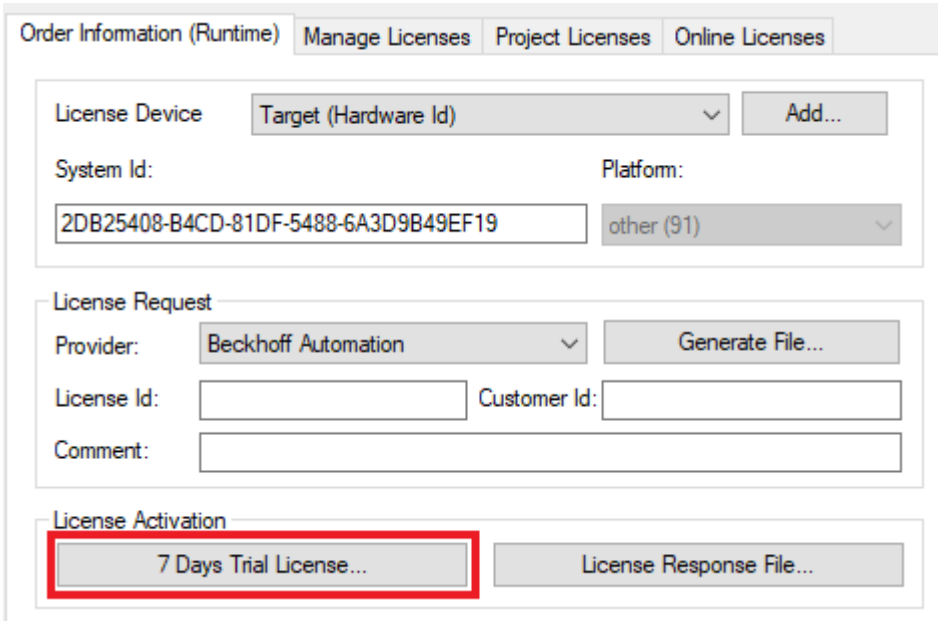
   ⇨ Der TwinCAT-3-Lizenzmanager öffnet sich.
5. Öffnen Sie die Registerkarte **Manage Licenses.** Aktivieren Sie in der Spalte **Add License** das Auswahlkästchen für die Lizenz, die Sie Ihrem Projekt hinzufügen möchten (z. B. „TF4100 TC3 Controller Toolbox“).



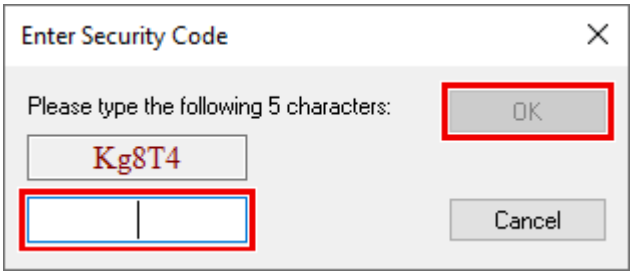
6. Öffnen Sie die Registerkarte **Order Information (Runtime).**

   ⇨ 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

4.1 Speicherverwaltung

Die Power-Monitoring-Bibliothek verwendet Teile der Condition-Monitoring-Bibliothek, die intern wiederum TcCOM-Objekte nutzt. Die TcCOM-Objekte werden von den installierten Treibern zur Verfügung gestellt. Die Instanzen werden dynamisch im TwinCAT-AMS-Routerspeicher angelegt.

Notwendigkeit dynamischer Speicherverwaltung

Alle Speicheranforderungen und Initialisierungen werden innerhalb der Initialisierungsphase umgesetzt bzw. durchgeführt. Da die Anzahl der Elemente der Eingangsdaten und der internen Strukturen von der Konfiguration der jeweiligen Bausteine abhängt, wird der Speicherplatz für diese grundsätzlich dynamisch angelegt. Bei Verwendung der Condition-Monitoring-Bibliothek geschieht dies automatisch.

Da alle Speicherbelegungen bei der Initialisierung erfolgen und somit die Initialisierung von Bausteinen unter Umständen eine relativ große Speichermenge beansprucht, kann die Initialisierung an dieser Stelle aufgrund Speichermangels fehlschlagen, später jedoch nicht.
Der belegte Speicher wird wieder freigegeben, wenn das Objekt gelöscht wird.

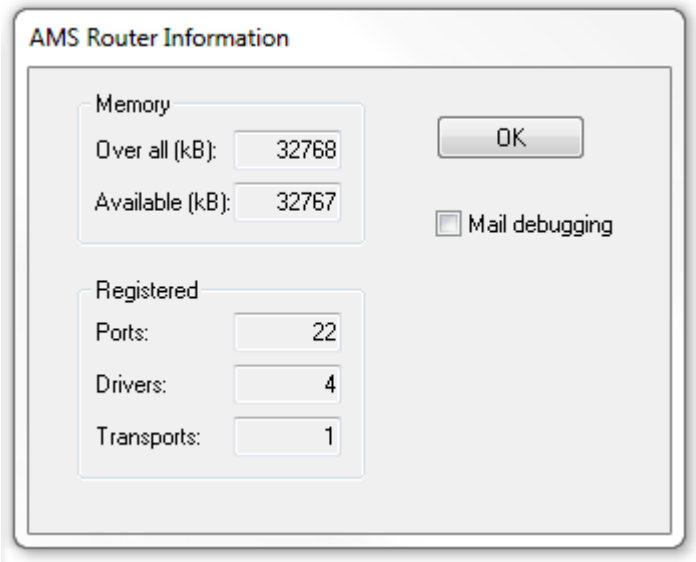
TwinCAT-Routerspeicher für dynamisch erzeugte Objekte

Die Puffer, welche die Condition-Monitoring-Bibliothek reserviert, werden bei der Initialisierung von Funktionsbausteinen im TwinCAT-AMS-Routerspeicher angelegt, sodass sie für eine Ausführung unter Echtzeitbedingungen zur Verfügung stehen. Bestimmte Funktionen wie z. B. hochauflösende Histogramme und Quantile, aber auch die Berechnung von Spektren mit sehr hoher Auflösung, erfordern wesentlich mehr Routerspeicher als herkömmliche Steuerungsprogramme. Deswegen muss der Routerspeicher möglicherweise vergrößert werden.

Routerspeicher anpassen

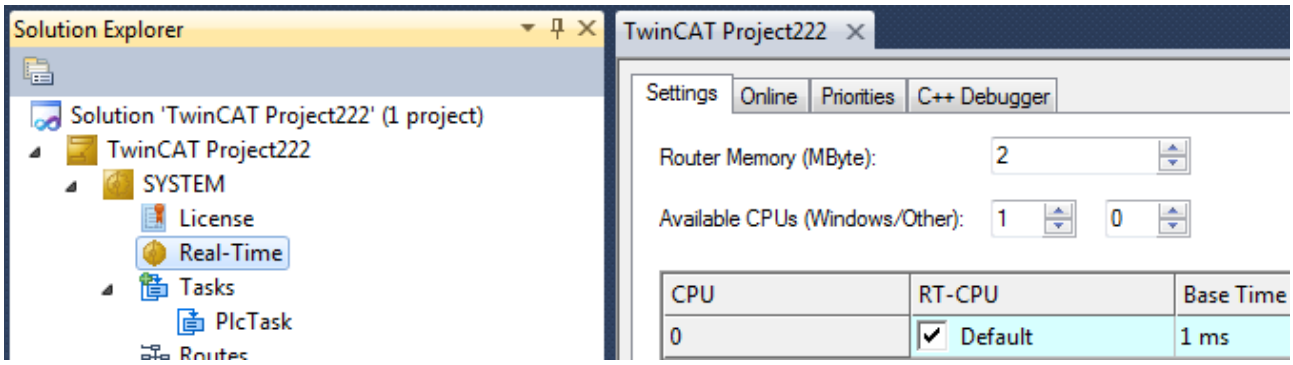
Die Standardgröße des Routerspeichers beträgt 32 MB. Die aktuelle Einstellung wird im Dialog **AMS Router Information** angezeigt.

Um den Dialog zu öffnen, klicken Sie mit der rechten Maustaste auf das TwinCAT-System-Service-Symbol im Informationsbereich der Taskleiste und wählen Sie in dem sich öffnenden Systemmenü den Befehl **Router > Info.**



Um den Routerspeicher zu vergrößern, öffnen Sie die Echtzeit-Einstellungen im TwinCAT-Engineering und tragen Sie in der TwinCAT-Konfiguration einen Wert in MB ein (**TwinCAT-Projektbaum > SYSTEM > Real-Time > Registerkarte Settings > Router Memory**). Aktivieren Sie anschließend die Konfiguration.

Die Anpassung des Routerspeichers erfordert einen Reboot des Zielgerätes.



5 SPS API

5.1 Funktionsbausteine Allgemein

5.1.1 FB_PMA_Scaling

Der Funktionsbaustein FB_PMA_Scaling dient der Skalierung von Rohwerten. Die Rohwerte können sowohl einzeln als auch als Array, beispielsweise als Oversampling-Werte, skaliert werden. Des Weiteren ist es möglich, sowohl einphasige als auch dreiphasige Eingangssignale zu verwenden.

Alternativ können die spezialisierten Funktionsbausteine FB_PMA_Scaling_EL3773 [▶ 22] für Eingänge der EtherCAT-Klemme EL3773 und FB_PMA_Scaling_EL3783 [▶ 26] für Eingänge der EtherCAT-Klemme EL3783 genutzt werden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Scaling
VAR_INPUT
    stInitPars      : ST_PMA_Scaling_InitPars
END_VAR
VAR_OUTPUT
    bError          : BOOL;
    ipResultMessage : I_TcMessage;
    bNewResult      : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 21]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist durch das erneute Aufrufen der Init [▶ 21]-Methode möglich.

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.

Methoden

Mithilfe von Methoden können unterschiedliche Systeme (einphasig bzw. dreiphasig) sowie unterschiedliche Auflösungen (16 Bit bzw. 32 Bit) skaliert werden.

Name	Beschreibung
Call_1Ph_16Bit [▶ 18]	Die Methode wird aufgerufen, um die 16-Bit-Eingangsdaten vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.

Name	Beschreibung
Call_1Ph_32Bit [▶ 19]	Die Methode wird aufgerufen, um die 32-Bit-Eingangsdaten vom Typ DINT entsprechend der konfigurierten Parameter zu skalieren.
Call_3Ph_16Bit [▶ 19]	Die Methode wird aufgerufen, um die 16-Bit-Eingangsdaten vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Call_3Ph_32Bit [▶ 20]	Die Methode wird aufgerufen, um die 32-Bit-Eingangsdaten vom Typ DINT entsprechend der konfigurierten Parameter zu skalieren.
Init [▶ 21]	Alternative zur Bausteininitialisierung
Reconfigure [▶ 21]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.1.1.1 Call_1Ph_16Bit

Die Methode wird aufgerufen, um die 16-Bit-Eingangsdaten vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.

Syntax

```
METHOD Call_1Ph_16Bit : BOOL
VAR_INPUT
    pInputBuffer_U    : POINTER TO INT;
    pInputBuffer_I    : POINTER TO INT;
    nInputBufferSize  : UDINT;
    pOutputBuffer_U   : POINTER TO LREAL;
    pOutputBuffer_I   : POINTER TO LREAL;
    nOutputBufferSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_U	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_I	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_U	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_I	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_1Ph_16Bit	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.1.2 Call_1Ph_32Bit

Die Methode wird aufgerufen, um die 32-Bit-Eingangsdaten vom Typ DINT entsprechend der konfigurierten Parameter zu skalieren.

Syntax

```
METHOD Call_1Ph_32Bit : BOOL
VAR_INPUT
    pInputBuffer_U    : POINTER TO DINT;
    pInputBuffer_I    : POINTER TO DINT
    nInputBufferSize  : UDINT;
    pOutputBuffer_U   : POINTER TO LREAL;
    pOutputBuffer_I   : POINTER TO LREAL;
    nOutputBufferSize : UDINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_U	POINTER TO DINT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_I	POINTER TO DINT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_U	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_I	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_1Ph_32Bit	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.1.3 Call_3Ph_16Bit

Die Methode wird aufgerufen, um die 16-Bit-Eingangsdaten vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.

Syntax

```
METHOD Call_3Ph_16Bit : BOOL
VAR_INPUT
    pInputBuffer_UL1   : POINTER TO INT;
    pInputBuffer_UL2   : POINTER TO INT;
    pInputBuffer_UL3   : POINTER TO INT;
    pInputBuffer_IL1   : POINTER TO INT
    pInputBuffer_IL2   : POINTER TO INT
    pInputBuffer_IL3   : POINTER TO INT
    nInputBufferSize   : UDINT;
    pOutputBuffer_UL1  : POINTER TO LREAL;
    pOutputBuffer_UL2  : POINTER TO LREAL;
    pOutputBuffer_UL3  : POINTER TO LREAL;
```

```
    pOutputBuffer_IL1  : POINTER TO LREAL;
    pOutputBuffer_IL2  : POINTER TO LREAL;
    pOutputBuffer_IL3  : POINTER TO LREAL;
    nOutputBufferSize  : UDINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_UL1 .. UL3	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_IL1 .. IL3	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_3Ph_16Bit	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.1.4 Call_3Ph_32Bit

Die Methode wird aufgerufen, um die 32-Bit-Eingangsdaten vom Typ DINT entsprechend der konfigurierten Parameter zu skalieren.

Syntax

```
METHOD Call_3Ph_32Bit : BOOL
VAR_INPUT
    pInputBuffer_UL1   : POINTER TO DINT;
    pInputBuffer_UL2   : POINTER TO DINT;
    pInputBuffer_UL3   : POINTER TO DINT;
    pInputBuffer_IL1   : POINTER TO DINT
    pInputBuffer_IL2   : POINTER TO DINT
    pInputBuffer_IL3   : POINTER TO DINT
    nInputBufferSize   : UDINT;
    pOutputBuffer_UL1  : POINTER TO LREAL;
    pOutputBuffer_UL2  : POINTER TO LREAL;
    pOutputBuffer_UL3  : POINTER TO LREAL;
    pOutputBuffer_IL1  : POINTER TO LREAL;
    pOutputBuffer_IL2  : POINTER TO LREAL;
    pOutputBuffer_IL3  : POINTER TO LREAL;
    nOutputBufferSize  : UDINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_UL1 .. UL3	POINTER TO DINT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_IL1 .. IL3	POINTER TO DINT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines Einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_3Ph_32Bit	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.1.5 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung).

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    stInitPars : ST_PMA_Scaling_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.1.6 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fOffsetVoltage  : LREAL  := 0.0;
    fGainVoltage    : LREAL  := 1.0;
    fOffsetCurrent  : LREAL  := 0.0;
    fGainCurrent    : LREAL  := 1.0;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.2 FB_PMA_Scaling_EL3773

Der Funktionsbaustein FB_PMA_Scaling_EL3773 ist eine Spezialisierung des Funktionsbausteins FB_PMA_Scaling [▶ 17] und dient der Skalierung von Rohwerten, die von der EtherCAT-Klemme EL3773 bereitgestellt werden. Die Rohwerte können sowohl einzeln als auch als Array, beispielsweise als Oversampling-Werte, skaliert werden. Des Weiteren ist es möglich, sowohl einphasige als auch dreiphasige Eingangssignale zu verwenden.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Scaling_EL3773
VAR_INPUT
    stInitPars       : ST_PMA_Scaling_EL3773_InitPars;
END_VAR
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
    bNewResult       : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 25]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist durch das erneute Aufrufen der Init [▶ 25]-Methode möglich.

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_EL3773_InitPars [▶ 129]	Bausteinspezifische Struktur mit Initialisierungsparametern

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.

Methoden

Mithilfe von Methoden können unterschiedliche Systeme (einphasig bzw. dreiphasig) skaliert werden.

Name	Beschreibung
Call_1Ph [▶ 23]	Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Call_3Ph [▶ 24]	Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Init [▶ 25]	Alternative zur Bausteininitialisierung
Reconfigure [▶ 25]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.1.2.1 Call_1Ph

Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben.

Syntax

```
METHOD Call_1Ph : BOOL
VAR_INPUT
    pInputBuffer_U    : POINTER TO INT;
    pInputBuffer_I    : POINTER TO INT;
    nInputBufferSize  : UDINT;
    pOutputBuffer_U   : POINTER TO LREAL;
    pOutputBuffer_I   : POINTER TO LREAL;
    nOutputBufferSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_U	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_I	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.

Name	Typ	Beschreibung
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_U	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_I	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_1Ph	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.2.2 Call_3Ph

Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben.

Syntax

```
METHOD Call_3Ph : BOOL
VAR_INPUT
    pInputBuffer_UL1   : POINTER TO INT;
    pInputBuffer_UL2   : POINTER TO INT;
    pInputBuffer_UL3   : POINTER TO INT;
    pInputBuffer_IL1   : POINTER TO INT
    pInputBuffer_IL2   : POINTER TO INT
    pInputBuffer_IL3   : POINTER TO INT
    nInputBufferSize   : UDINT;
    pOutputBuffer_UL1  : POINTER TO LREAL;
    pOutputBuffer_UL2  : POINTER TO LREAL;
    pOutputBuffer_UL3  : POINTER TO LREAL;
    pOutputBuffer_IL1  : POINTER TO LREAL;
    pOutputBuffer_IL2  : POINTER TO LREAL;
    pOutputBuffer_IL3  : POINTER TO LREAL;
    nOutputBufferSize  : UDINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_UL1 .. UL3	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_IL1 .. IL3	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_UL1 .. UL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_IL1 .. IL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call_3Ph	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.2.3 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung).

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    stInitPars : ST_PMA_Scaling_EL3773_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_EL3773_InitPars [▶ 129]	Bausteinspezifische Struktur mit Initialisierungsparametern

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.2.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fOffsetVoltage  : LREAL  := 0.0;
    fGainVoltage    : LREAL  := 1.0;
    fOffsetCurrent  : LREAL  := 0.0;
    fGainCurrent    : LREAL  := 1.0;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.

Name	Typ	Beschreibung
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3 FB_PMA_Scaling_EL3783

Der Funktionsbaustein FB_PMA_Scaling_EL3783 ist eine Spezialisierung des Funktionsbausteins FB_PMA_Scaling [▶ 17] und dient der Skalierung von Rohwerten, die von der EtherCAT-Klemme EL3783 bereitgestellt werden. Die Rohwerte können sowohl einzeln als auch als Array, beispielsweise als Oversampling-Werte, skaliert werden. Des Weiteren ist es möglich, sowohl einphasige als auch dreiphasige Eingangssignale zu verwenden. Der Funktionsbaustein unterstützt die Autorange-Funktionalität der EL3783, die in zwei Strommessbereichen arbeiten kann.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Scaling_EL3783
VAR_INPUT
    stInitPars       : ST_PMA_Scaling_EL3783_InitPars;
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
    bNewResult       : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 30]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist durch erneutes Aufrufen der Init [▶ 30]-Methode möglich.

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_EL3783_InitPars [▶ 130]	Bausteinspezifische Struktur mit Initialisierungsparametern

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.

Methoden

Mithilfe von Methoden können unterschiedliche Systeme (einphasig bzw. dreiphasig) skaliert werden.

Name	Beschreibung
Call_1Ph [▶ 27]	Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Call_1Ph_Autorange [▶ 28]	Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Call_3Ph [▶ 29]	Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Call_3Ph_Autorange [▶ 29]	Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren.
Init [▶ 30]	Alternative zur Bausteininitialisierung
Reconfigure [▶ 31]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.1.3.1 Call_1Ph

Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben.

Syntax

```
METHOD Call_1Ph : BOOL
VAR_INPUT
    pInputBuffer_U     : POINTER TO INT;
    pInputBuffer_I     : POINTER TO INT;
    nInputBufferSize   : UDINT;
    pOutputBuffer_U    : POINTER TO LREAL;
    pOutputBuffer_I    : POINTER TO LREAL;
    nOutputBufferSize  : UDINT;
    bUse_5A_Range      : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_U	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_I	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_U	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_I	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.
bUse_5A_Range	BOOL	Wenn der Wert TRUE ist, wird der 5A-Messbereich der EL3783 genutzt. Wenn er FALSE ist, wird der 1A-Messbereich verwendet.

Rückgabewert

Name	Typ	Beschreibung
Call_1Ph	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3.2 Call_1Ph_Autorange

Die Methode wird aufgerufen, um die Eingangsdaten in einem einphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben. Die EL3783 wird im Auto-Range-Modus betrieben.

Syntax

```
METHOD Call_1Ph_Autorange : BOOL
VAR_INPUT
    pInputBuffer_U          : POINTER TO INT;
    pInputBuffer_I          : POINTER TO INT;
    nInputBufferSize        : UDINT;
    pOutputBuffer_U         : POINTER TO LREAL;
    pOutputBuffer_I         : POINTER TO LREAL;
    nOutputBufferSize       : UDINT;
    bEL3783_HcRangeActive   : BOOL;
    aEL3783_HcRange         : ARRAY [0..3] OF USINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_U	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_I	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_U	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_I	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.
bEL3783_HcRangeActive	BOOL	TRUE, wenn der Auto-Range-Modus an der Klemme aktiv ist.
aEL3783_HcRange	ARRAY [0..3] OF USINT	Die aktuellen Messbereichsinformationen der EL3783.

Rückgabewert

Name	Typ	Beschreibung
Call_1Ph_Autorange	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3.3 Call_3Ph

Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben.

Syntax

```
METHOD Call_3Ph : BOOL
VAR_INPUT
    pInputBuffer_UL1    : POINTER TO INT;
    pInputBuffer_UL2    : POINTER TO INT;
    pInputBuffer_UL3    : POINTER TO INT;
    pInputBuffer_IL1    : POINTER TO INT
    pInputBuffer_IL2    : POINTER TO INT
    pInputBuffer_IL3    : POINTER TO INT
    nInputBufferSize    : UDINT;
    pOutputBuffer_UL1   : POINTER TO LREAL;
    pOutputBuffer_UL2   : POINTER TO LREAL;
    pOutputBuffer_UL3   : POINTER TO LREAL;
    pOutputBuffer_IL1   : POINTER TO LREAL;
    pOutputBuffer_IL2   : POINTER TO LREAL;
    pOutputBuffer_IL3   : POINTER TO LREAL;
    nOutputBufferSize   : UDINT;
    bUse_5A_Range       : BOOL;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_UL1 .. UL3	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_IL1 .. IL3	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_UL1 .. UL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_IL1 .. IL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.
bUse_5A_Range	BOOL	Wenn der Wert TRUE ist, wird der 5A-Messbereich der EL3783 genutzt. Wenn er FALSE ist, wird der 1A-Messbereich verwendet.

Rückgabewert

Name	Typ	Beschreibung
Call_3Ph	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3.4 Call_3Ph_Autorange

Die Methode wird aufgerufen, um die Eingangsdaten in einem dreiphasigen System vom Typ INT entsprechend der konfigurierten Parameter zu skalieren. Es wird ein LREAL ausgegeben. Die EL3783 wird im Auto-Range-Modus betrieben.

Syntax

```
METHOD Call_3Ph : BOOL
VAR_INPUT
    pInputBuffer_UL1        : POINTER TO INT;
    pInputBuffer_UL2        : POINTER TO INT;
    pInputBuffer_UL3        : POINTER TO INT;
    pInputBuffer_IL1        : POINTER TO INT
    pInputBuffer_IL2        : POINTER TO INT
    pInputBuffer_IL3        : POINTER TO INT
    nInputBufferSize        : UDINT;
    pOutputBuffer_UL1       : POINTER TO LREAL;
    pOutputBuffer_UL2       : POINTER TO LREAL;
    pOutputBuffer_UL3       : POINTER TO LREAL;
    pOutputBuffer_IL1       : POINTER TO LREAL;
    pOutputBuffer_IL2       : POINTER TO LREAL;
    pOutputBuffer_IL3       : POINTER TO LREAL;
    nOutputBufferSize       : UDINT;
    bEL3783_HcRangeActive   : BOOL;
    aEL3783_HcRange         : ARRAY [0..3] OF USINT;
END_VAR
VAR_OUTPUT
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputBuffer_UL1 .. UL3	POINTER TO INT	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pInputBuffer_IL1 .. IL3	POINTER TO INT	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nInputBufferSize	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.
pOutputBuffer_UL1 .. UL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Spannungswerte gespeichert werden sollen.
pOutputBuffer_IL1 .. IL3	POINTER_TO_LREAL	Zeiger auf ein Array, in das die skalierten Stromwerte gespeichert werden sollen.
nOutputBufferSize	UDINT	Gibt die Größe eines einzelnen Ausgangspuffers in Bytes an.
bEL3783_HcRangeActive	BOOL	TRUE, wenn der Auto-Range-Modus an der Klemme aktiv ist.
aEL3783_HcRange	ARRAY [0..3] OF USINT	Die aktuellen Messbereichsinformationen der EL3783.

Rückgabewert

Name	Typ	Beschreibung
Call_3Ph_Autorange	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3.5 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung).

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    stInitPars  : ST_PMA_Scaling_EL3783_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stInitPars	ST_PMA_Scaling_EL3783_InitPars [▶ 130]	Bausteinspezifische Struktur mit Initialisierungsparametern

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.3.6 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fOffsetVoltage  : LREAL  := 0.0;
    fGainVoltage    : LREAL  := 1.0;
    fOffsetCurrent  : LREAL  := 0.0;
    fGainCurrent    : LREAL  := 1.0;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.4 FB_PMA_TaskTransfer_Send

Der Funktionsbaustein FB_PMA_TaskTransfer_Send kann verwendet werden, um Daten von einer langsamen Task zurück in eine schnelle Task zu übertragen. Dies kann erforderlich sein, wenn rechenintensive Algorithmen in eine langsame Task ausgelagert wurden und die Ergebnisse in einer anderen Task benötigt werden.

Zusammen mit dem Funktionsbaustein FB_PMA_TaskTransfer_Receive [▶ 34] bildet er ein Bausteinpaar zum Senden und Empfangen von Daten. Die Daten werden nach jedem Aufruf der Call [▶ 33]-Methode an einen Funktionsbaustein mit der Ziel-Analyse-ID übergeben.

Die Ausgangspuffer werden für die Funktionsbausteine bereitgestellt, deren ID in dem Array von Ziel-IDs eingetragen ist. Der zu initialisierende Puffer muss mindestens so groß sein wie die zu übermittelnden Daten.

Syntax

Definition:

```
FUNCTION BLOCK FB_PMA_TaskTransfer_Send
VAR_INPUT
    nOwnID            : UDINT;
    aDestIDs          : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers    : UDINT := 4;
    tTransferTimeout  : LTIME := LTIME#40US;
    stInitPars        : ST_PMA_TaskTransfer_InitPars;
END_VAR
VAR_OUTPUT
    bError            : BOOL;
    ipResultMessage   : I_TcMessage;
    bNewResult        : BOOL;
    nCntResults       : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 34]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Die Daten des Bausteins werden an die hier als Array angegebenen IDs der Bausteininstanzen vom Typ FB_PMA_TaskTransfer_Receive [▶ 34] weitergeleitet.
nResultBuffers	UDINT	Anzahl von MultiArray-Puffern, die für die Ergebnisse initialisiert werden.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_TaskTransfer_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse bereitgestellt wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 33]	Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben.
Init [▶ 34]	Alternative zur Bausteininitialisierung

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.1.4.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben. Der Ausgangspuffer wird zyklisch verschickt.

Syntax

```
METHOD CALL : HRESULT
VAR_INPUT
    pInputData    : POINTER TO LREAL,
    nDataInSize   : UDINT;
    nOptionPars   : DWORD;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pInputData	POINTER TO LREAL	Zeiger auf einen Eingangspuffer der zu versendenden Daten.
nDataInSize	UDINT	Gibt die Größe des Eingangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.4.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : HRESULT
VAR_INPUT
    nOwnID          : UDINT;
    aDestIDs        : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers  : UDINT := 4;
    stInitPars      : ST_PMA_TaskTransfer_InitPars,
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Die Daten des Bausteins werden an die hier als Array angegebenen IDs der Bausteininstanzen vom Typ FB_PMA_TaskTransfer_Receive [▶ 34] weitergeleitet.
nResultBuffers	UDINT	Anzahl der zur Verfügung stehenden MultiArrays
stInitPars	ST_PMA_TaskTransfer_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.5 FB_PMA_TaskTransfer_Receive

Der Funktionsbaustein FB_PMA_TaskTransfer_Receive kann verwendet werden, um Daten von einer langsamen Task zurück in eine schnelle Task zu übertragen. Dies kann erforderlich sein, wenn rechenintensive Algorithmen in eine langsame Task ausgelagert wurden und die Ergebnisse in einer anderen Task benötigt werden.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_TaskTransfer_Send [▶ 32] bereitgestellt. Der zu initialisierende Puffer muss mindestens so groß sein wie die zu übermittelnden Daten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_TaskTransfer_Receive
VAR_INPUT
    nOwnID            : UDINT
    tTransferTimeout  : LTIME := LTIME#500US
    stInitPars        : ST_PMA_TaskTransfer_InitPars;
END_VAR
VAR_OUTPUT
    bError            : BOOL;
    ipResultMessage   : I_TcMessage;
    bNewResult        : BOOL;
    nCntResults       : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 36]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
nResultBuffers	UDINT	Anzahl von MultiArray-Puffern, die für die Ergebnisse initialisiert werden.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_TaskTransfer_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse bereitgestellt wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 36]	Die Methode wird in jedem Zyklus aufgerufen, um die Werte aus den Eingangspuffer zu lesen, wenn neue Daten vorhanden sind.
Init [▶ 36]	Alternative zur Bausteininitialisierung

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.1.5.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Eingangspuffer zu lesen, wenn neue Daten vorhanden sind. Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD CALL : HRESULT
VAR_INPUT
    pOutputData    : POINTER TO LREAL,
    nDataOutSize   : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pOutputData	POINTER TO LREAL	Zeiger auf den Ausgangspuffer.
nDataOutSize	UDINT	Gibt die Größe des Ausgangspuffers in Bytes an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.1.5.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : HRESULT
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Source_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
stInitPars	ST_PMA_TaskTransfer_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2 Funktionsbausteine Einphasig

5.2.1 FB_PMA_Source_1Ph

Der Funktionsbaustein FB_PMA_Source_1Ph schreibt Daten aus einem externen SPS-Datenpuffer in einen MultiArray-Puffer.

Er häuft ununterbrochen Eingangsdaten an, bis die Größe des MultiArrays erreicht ist. Wenn das MultiArray komplett gefüllt ist, wird es an einen Funktionsbaustein mit der Ziel-Analyse-ID übergeben.

Die Ausgangspuffer werden für die Funktionsbausteine bereitgestellt, deren ID in dem Array von Ziel-IDs eingetragen ist. Sie enthalten die Werte von Strom und Spannung. Die Anforderung an die Größe der Ausgangspuffer kann je nach verwendetem Analysebaustein unterschiedlich sein. Sie ist entweder abhängig von der verwendeten FFT-Länge oder der Puffergröße.

Syntax

Definition:

```
FUNCTION BLOCK FB_PMA_Source_1Ph
VAR_INPUT
    nOwnID            : UDINT;
    aDestIDs          : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers    : UDINT := 4;
    tTransferTimeout  : LTIME := LTIME#40US;
    stInitPars        : ST_PMA_Source_InitPars;
END_VAR
VAR_OUTPUT
    bError            : BOOL;
    ipResultMessage   : I_TcMessage;
    bNewResults       : BOOL;
    nCntResults       : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 39]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
nResultBuffers	UDINT	Anzahl von MultiArray-Puffern, die für die Ergebnisse initialisiert werden.
tTransferTimeout	LTIME	Einstellung des synchronen Timeouts für interne MultiArray-Weiterleitungen.
stInitPars	ST_PMA_Source_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse bereitgestellt wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 38]	Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben.
ResetData [▶ 39]	Mit dieser Methode können die Daten, die sich aktuell im Puffer befinden, zurückgesetzt werden.
ResetAnalysisChain [▶ 40]	Durch Aufruf dieser Methode wird automatisch ein Reset aller Algorithmen der vollständigen Analysekette durchgeführt.
Init [▶ 39]	Alternative zur Bausteininitialisierung

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben. Der Ausgangspuffer wird verschickt, sobald dieser gefüllt ist.

Syntax

```
METHOD CALL : BOOL
VAR_INPUT
    pBuffer_U         : POINTER TO LREAL;
    pBuffer_I         : POINTER TO LREAL;
    nDataInSizePerCh  : UDINT;
    nOptionPars       : DWORD;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pBuffer_U	POINTER TO LREAL	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pBuffer_I	POINTER TO LREAL	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nDataInSizePerCh	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.

5.2.1.2 ResetData

Mit dieser Methode können die Daten, die sich aktuell im Puffer befinden, zurückgesetzt werden.

Syntax

```
METHOD ResetData : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
ResetData	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.1.3 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID          : UDINT;
    aDestIDs        : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers  : UDINT := 4;
    stInitPars      : ST_PMA_Source_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Die Ergebnisdaten werden an die hier als Array angegebenen IDs anderer Bausteininstanzen weitergeleitet.
nResultBuffers	UDINT	Anzahl der zur Verfügung stehenden MultiArrays.
stInitPars	ST_PMA_Source_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.1.4 ResetAnalysisChain

Durch Aufruf dieser Methode wird automatisch ein Reset aller Algorithmen der vollständigen Analysekette durchgeführt. Intern wird jeweils ein ResetData() vor der Übernahme des neuen Datensatzes ausgeführt.

Soll die Analysekette nur für einen bestimmten Zeitraum aktiv sein, so bietet diese Methode die Möglichkeit alle Algorithmen vor der nächsten Ausführung zurückzusetzen.

Es können Fehler beim Aufruf einer Input Methode auftreten und Unterbrechungen der Zeitreihensammlung verursachen. Falls die folgenden Algorithmen der Analysekette Spektren berechnen, so kann im Falle eines Fehlers beim Aufruf einer Input Methode die ResetAnalysisChain() Methode aufgerufen werden. Denn es ist nicht möglich korrekte Spektren anhand zerstückelter Zeitreihen zu berechnen.

Syntax

```
METHOD ResetAnalysisChain : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
ResetAnalysisChain	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2 Basierend auf der Signalperiode

5.2.2.1 FB_PMA_Frequency_Period_1Ph

Der Funktionsbaustein FB_PMA_Frequency_Period_1Ph berechnet die Grundfrequenz des gegebenen Eingangssignals. Dafür wird das Signal zunächst mit einem Butterworth-Tiefpassfilter gefiltert. Anschließend werden aus den gefilterten Werten die Nulldurchgänge des Eingangssignals ermittelt und aus deren Differenz die Frequenz berechnet. Die Ergebnisse beziehen sich, abhängig von der Konfiguration, auf eine oder mehrere Perioden. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Arrays.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Frequency_Period_1Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Frequency_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
    fFreq               : LREAL;
    fFreq_Min           : LREAL;
    fFreq_Max           : LREAL;
    fRocof              : LREAL;
    bValidStatistics    : BOOL;
    bOutOfRange         : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 43]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Frequency_Period_InitPars [▶ 133]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fFreq	LREAL	Frequenz, die durch zwei oder mehr Nulldurchgänge ermittelt wurde.
fFreq_Min	LREAL	Kleinster aufgetretener Wert von fFreq. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fFreq_Max	LREAL	Größter aufgetretener Wert von fFreq. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fRocof	LREAL	Rate of change of frequency (ROCOF).

Name	Typ	Beschreibung
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
bOutOfRange	BOOL	TRUE, sobald der Eingangswert oder die Frequenz nicht innerhalb der konfigurierten Grenzen ist.

Methoden

Name	Beschreibung
Call [▶ 42]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 43]	Alternative zur Bausteininitialisierung
PassInputs [▶ 44]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 44]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 45]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_1Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithm
fbFrequency.Call(FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.2.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.1.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Frequency_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Frequency_Period_InitPars [▶ 133]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.1.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.1.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinFreq       : LREAL;
    fMaxFreq       : LREAL;
    nPeriods       : UDINT;
    nFilterOrder   : UINT;
    fCutoff        : LREAL;
    eInputSelect   : E_PMA_InputSelect;
    fMinInput      : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinFreq	LREAL	Minimal zu erwartende Messfrequenz
fMaxFreq	LREAL	Maximal zu erwartende Messfrequenz
nPeriods	UDINT	Anzahl an Perioden, die Einfluss auf die Berechnung haben. (Periodenlänge = Samplerate/Frequenz)
nFilterOrder	UINT	Gibt die Ordnung des Tiefpassfilters an. Bei der Einstellung muss die Stabilität des Filters berücksichtigt werden. Nur Werte bis zur zehnten Ordnung sind zulässig.
fCutoff	LREAL	Gibt die Grenzfrequenz des Tiefpassfilters an.
eInputSelect	E_PMA_InputSelect [▶ 146]	Hier kann konfiguriert werden, ob die Frequenz der Spannung oder des Stroms berechnet werden soll.
fMinInput	LREAL	Mindesteingangsgröße (RMS) über eine Periode. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.1.5 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.2 FB_PMA_BasicValues_Period_1Ph

Der Funktionsbaustein FB_PMA_BasicValues_Period_1Ph berechnet Analysewerte für den zeitlichen Signalverlauf von Strom und Spannung in einem einphasigen System. Dazu gehören Mittelwert, RMS-Wert, Spitzenwert, Gleichrichtwert, Crest-Faktor sowie Formfaktor jeweils für Strom und Spannung. Die Ergebnisse beziehen sich auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 48]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Arrays.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_BasicValues_Period_1Ph
VAR_INPUT
    nOwnID                : UDINT;
    tTransferTimeout      : LTIME := LTIME#500US;
    stInitPars            : ST_PMA_BasicValues_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
    fMeanValue_U        : LREAL;
    fRMS_U              : LREAL;
    fRMS_U_Min          : LREAL;
    fRMS_U_Max          : LREAL;
    fPeakValue_U        : LREAL;
    fPeakHold_U         : LREAL;
    fRectifiedValue_U   : LREAL;
    fCrestFactor_U      : LREAL;
    fFormFactor_U       : LREAL;
    fMeanValue_I        : LREAL,
    fRMS_I              : LREAL,
```

```
    fRMS_I_Min          : LREAL,
    fRMS_I_Max          : LREAL,
    fPeakValue_I        : LREAL,
    fPeakHold_I         : LREAL;
    fRectifiedValue_I   : LREAL;
    fCrestFactor_I      : LREAL;
    fFormFactor_I       : LREAL;
    bValidStatistics    : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 94]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_BasicValues_Period_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fMeanValue_U	LREAL	Mittelwert der Spannung über n Perioden
fRMS_U	LREAL	Effektivwert der Spannung über n Perioden
fRMS_U_Min	LREAL	Kleinster aufgetretener Wert von fRMS_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fRMS_U_Max	LREAL	Größter aufgetretener Wert von fRMS_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fPeakValue_U	LREAL	Spitzenwert der Spannung über n Perioden
fPeakHold_U	LREAL	Allzeit-Spitzenwert der Spannung über n Perioden. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fRectifiedValue_U	LREAL	Gleichrichtwert der Spannung über n Perioden
fCrestFactor_U	LREAL	Scheitelfaktor (Crest-Faktor) der Spannung (Spitzenwert/Effektivwert)
fFormFactor_U	LREAL	Formfaktor der Spannung (Effektivwert/Gleichrichtwert)
fMeanValue_I	LREAL	Mittelwert des Stroms über n Perioden
fRMS_I	LREAL	Effektivwert des Stroms über n Perioden

Name	Typ	Beschreibung
fRMS_I_Min	LREAL	Kleinster aufgetretener Wert von fRMS_I. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fRMS_I_Max	LREAL	Größter aufgetretener Wert von fRMS_I. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fPeakValue_I	LREAL	Spitzenwert des Stroms über n Perioden
fPeakHold_I	LREAL	Allzeit-Spitzenwert des Stroms. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fRectifiedValue_I	LREAL	Gleichrichtwert des Stroms über n Perioden
fCrestFactor_I	LREAL	Scheitelfaktor (Crest-Faktor) des Stroms (Spitzenwert/Effektivwert)
fFormFactor_I	LREAL	Formfaktor des Stroms (Effektivwert/ Gleichrichtwert)
bValidStatistics	BOOL	TRUE, wenn die Min-, Max- und Hold-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 48]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 94]	Alternative zur Bausteininitialisierung
PassInputs [▶ 95]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 49]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 50]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cBasicValuesInitPars : ST_PMA_BasicValues_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinInputCurrent := 0.01,
        nPeriods := 1);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_1Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbBasicValues : FB_PMA_BasicValues_Period_1Ph := (nOwnID := 3, stInitPars := cBasicValuesInitPar
s);
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithms
fbFrequency.Call(FALSE);
fbBasicValues.Call(fbFrequency.fFreq, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.2.2.1 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_BasicValues_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_BasicValues_Period_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.2.2 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq             : LREAL;
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet, um zu Beginn einer Periode die Periodenlänge zu ermitteln. Der Ausgang des Funktionsbausteins FB_CMA_Frequency_Period_1Ph [▶ 40] kann verwendet werden.
bResetStatistics	BOOL	TRUE setzt die Min-, Max-, und Hold-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.2.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.2.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinInputCurrent   : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.2.5 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.3 FB_PMA_PowerValues_Period_1Ph

Der Funktionsbaustein FB_PMA_PowerValues_Period_1Ph berechnet die Leistungswerte des angeschlossenen Verbrauchers. Dazu gehören auch die Grundschwingungskomponenten und der Phasenverschiebungswinkel. Hierfür werden neben dem Signalverlauf im Zeitbereich nur die ersten Harmonischen des Eingangssignals zur Berechnung herangezogen. Der Vorteil dieser Algorithmen ist die hohe Dynamik der Berechnungen. Die Ergebnisse beziehen sich auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 53]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Alternativ kann der Funktionsbaustein FB_PMA_PowerValues_1Ph [▶ 66] verwendet werden. Dieser nutzt intern die einzelnen Harmonischen zur Berechnung der Leistungswerte.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Werten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_PowerValues_Period_1Ph
VAR_INPUT
    nOwnID                : UDINT;
    tTransferTimeout      : LTIME := LTIME#500US;
    stInitPars            : ST_PMA_PowerValues_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError                : BOOL;
    ipResultMessage       : I_TcMessage;
    bNewResult            : BOOL;
    nCntResults           : ULINT;
    fApparentPower        : LREAL;
    fApparentPower_1      : LREAL;
    fApparentPower_1_Min  : LREAL;
    fApparentPower_1_Max  : LREAL;
    fActivePower          : LREAL;
    fActivePower_Min      : LREAL;
    fActivePower_Max      : LREAL;
    fReactivePower_d      : LREAL;
    fReactivePower_1      : LREAL;
    fReactivePower_1_Min  : LREAL;
    fReactivePower_1_Max  : LREAL;
    fTotalReactivePower   : LREAL;
    fPhi                  : LREAL;
    fCosPhi               : LREAL;
    fPowerFactor          : LREAL;
    fPowerQualityFactor   : LREAL;
    bValidStatistics      : BOOL;
END_VAR
VAR_OUTPUT PERSISTENT
    stEnergy_Pos          : ST_PMA_Energy;
    stEnergy_Neg          : ST_PMA_Energy;
    stEnergy_Res          : ST_PMA_Energy;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 54]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_PowerValues_Period_InitPars [▶ 134]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.

Name	Typ	Beschreibung
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fApparentPower	LREAL	Gesamt-Scheinleistung
fApparentPower_1	LREAL	Grundschwingungs-Scheinleistung
fApparentPower_1_Min	LREAL	Kleinster aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fApparentPower_1_Max	LREAL	Größter aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fActivePower	LREAL	Wirkleistung
fActivePower_Min	LREAL	Kleinster aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fActivePower_Max	LREAL	Größter aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fReactivePower_d	LREAL	Verzerrungsblindleistung
fReactivePower_1	LREAL	Grundschwingungs-Verschiebungs-Blindleistung
fReactivePower_1_Min	LREAL	Kleinster aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fReactivePower_1_Max	LREAL	Größter aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTotalReactivePower	LREAL	Gesamt-Blindleistung
fPhi	LREAL	Phasenverschiebungswinkel
fCosPhi	LREAL	CosPhi (Wirkleistung/Grundschwingungs-Scheinleistung)
fPowerFactor	LREAL	Leistungsfaktor (Wirkleistung/Gesamt-Scheinleistung)
fPowerQualityFactor	LREAL	Power Quality Factor. Stellt die Qualität der Spannungsversorgung vereinfacht in einem Wertebereich zwischen 0 und 1 dar. Miteinbezogen wird die Frequenz, der Effektivwert der Spannung sowie der THD der Spannung.
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
stEnergy_Pos	ST_PMA_Energy [▶ 146]	Energie in positiver Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
stEnergy_Neg	ST_PMA_Energy [▶ 146]	Energie in negativer Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
stEnergy_Res	ST_PMA_Energy [▶ 146]	Resultierende Energie. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.

Methoden

Name	Beschreibung
Call [▶ 53]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 54]	Alternative zur Bausteininitialisierung
PassInputs [▶ 55]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 55]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 56]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cPowerValuesInitPars : ST_PMA_PowerValues_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinInputCurrent := 0.01,
        nPeriods := 1);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_1Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbPowerValues : FB_PMA_PowerValues_Period_1Ph := (nOwnID := 3, stInitPars := cPowerValuesInitPar
s);
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithms
fbFrequency.Call(FALSE);
fbPowerValues.Call(fbFrequency.fFreq, FALSE, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.2.3.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq               : LREAL;
    bResetEnergyCalc    : LREAL;
    bResetStatistics    : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet, um zu Beginn einer Periode die Periodenlänge zu ermitteln. Der Ausgang des Funktionsbausteins FB_CMA_Frequency_Period_1Ph [▶ 40] kann verwendet werden.
bResetEnergyCalc	LREAL	TRUE setzt die berechneten Werte der Energiemessung zurück.
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.3.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT
    stInitPars  : ST_PMA_PowerValues_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_PowerValues_Period_InitPars [▶ 134]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.3.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.3.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinInputCurrent : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.3.5 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.4 FB_PMA_Harmonics_Period_1Ph

Der Funktionsbaustein FB_PMA_Harmonics_Period_1Ph berechnet die Harmonischen von Strom und Spannung. Zusätzlich wird aus den Harmonischen der THD der Eingangsgrößen berechnet. Im Gegensatz zum Funktionsbaustein FB_PMA_Harmonics_1Ph [▶ 60] beziehen sich die Ergebnisse auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 58]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Werten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Harmonics_Period_1Ph
VAR_INPUT
    nOwnID                : UDINT;
    tTransferTimeout      : LTIME := LTIME#500US;
    stInitPars            : ST_PMA_Harmonics_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError                : BOOL;
    ipResultMessage       : I_TcMessage;
    bNewResult            : BOOL;
    nCntResults           : ULINT;
    fTHD_U                : LREAL;
    fTHD_U_Min            : LREAL;
    fTHD_U_Max            : LREAL;
    fTHD_I                : LREAL;
    fTHD_I_Min            : LREAL;
    fTHD_I_Max            : LREAL;
    bValidStatistics      : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 59]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Harmonics_Period_Init Pars [▶ 135]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fTHD_U	LREAL	THD der Spannung. Die Ausgabe erfolgt in Prozent.
fTHD_U_Min	LREAL	Kleinster aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_U_Max	LREAL	Größter aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_I	LREAL	THD des Stroms. Die Ausgabe erfolgt in Prozent.
fTHD_U_Min	LREAL	Kleinster aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_U_Max	LREAL	Größter aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 58]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 59]	Alternative zur Bausteininitialisierung
PassInputs [▶ 60]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reset [▶ 60]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cHarmonicsInitPars : ST_PMA_Harmonics_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        nNumHarmonics := 20,
        nPeriods := 10,
        bTransformToPercent := TRUE);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_1Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbHarmonics : FB_PMA_Harmonics_Period_1Ph := (nOwnID := 3, stInitPars := cHarmonicsInitPars);
    aHarmonicsVoltage : ARRAY[1..20] OF LREAL;
    aHarmonicsCurrent : ARRAY[1..20] OF LREAL;
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithms
fbFrequency.Call(FALSE);
fbHarmonics.Call(fbFrequency.fFreq, ADR(aHarmonicsVoltage), ADR(aHarmonicsCurrent), SIZEOF(aHarmonic
sVoltage), FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.2.4.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq               : LREAL;
    pHarmonicsRMS_U     : POINTER TO LREAL;
    pHarmonicsRMS_I     : POINTER TO LREAL;
    nHarmonicsRMSSize   : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet, um zu Beginn einer Periode die Periodenlänge zu ermitteln. Der Ausgang des

Name	Typ	Beschreibung
		Funktionsbausteins FB_CMA_Frequency_Period_1Ph [▶ 40] kann verwendet werden.
pHarmonicsRMS_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pHarmonicsRMS_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nHarmonicsRMSSize	UDINT	Gibt die Größe eines Ausgangsarrays für die Harmonischen an.
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.4.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT
    stInitPars  : ST_PMA_Harmonics_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Harmonics_Period_InitPars [▶ 135]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.4.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.2.4.4 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3 Basierend auf dem Frequenzbereich

5.2.3.1 FB_PMA_Harmonics_1Ph

Der Funktionsbaustein FB_PMA_Harmonics_1Ph berechnet die RMS-Bänder der einzelnen Harmonischen von Strom und Spannung. Zusätzlich wird aus den berechneten RMS-Bändern der THD der Eingangsgrößen berechnet.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUCNTION_BLOCK FB_PMA_Harmonics_1Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Harmonics_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
    fTHD_U              : LREAL;
    fTHD_U_Min          : LREAL;
    fTHD_U_Max          : LREAL;
    fTHD_I              : LREAL;
    fTHD_I_Min          : LREAL;
    fTHD_I_Max          : LREAL;
    bValidStatistics    : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 64]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Harmonics_InitPars [▶ 136]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fTHD_U	LREAL	THD der Spannung. Die Ausgabe erfolgt in Prozent.
fTHD_U_Min	LREAL	Kleinster aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_U_Max	LREAL	Größter aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_I	LREAL	THD des Stroms. Die Ausgabe erfolgt in Prozent.
fTHD_U_Min	LREAL	Kleinster aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTHD_U_Max	LREAL	Größter aufgetretener Wert von fTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
bValidStatistics	BOOL	TRUE, wenn die Min-, Max- und Hold-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 63]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 64]	Alternative zur Bausteininitialisierung
PassInputs [▶ 64]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 65]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 65]	Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cHarmonicsInitPars : ST_PMA_Harmonics_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        fBaseFreq := 50.0,
        nNumBands := 40,
        fBandwidth := 20.0,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
```

```
        fDecibelThreshold := GVL_PMA.cMinArgLog10,
        bTransformToPercent := TRUE);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbHarmonics : FB_PMA_Harmonics_1Ph := (nOwnID := 2, stInitPars := cHarmonicsInitPars);
    aHarmonicsVoltage : ARRAY[1..40] OF LREAL;
    aHarmonicsCurrent : ARRAY[1..40] OF LREAL;
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithm
fbHarmonics.Call(ADR(aHarmonicsVoltage), ADR(aHarmonicsCurrent), SIZEOF(aHarmonicsVoltage), FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.3.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pHarmonicsRMS_U   : POINTER TO LREAL;
    pHarmonicsRMS_I   : POINTER TO LREAL;
    nHarmonicsRMSSize : UDINT;
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHarmonicsRMS_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pHarmonicsRMS_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nHarmonicsRMSSize	UDINT	Gibt die Größe eines Ausgangsarrays für die Harmonischen an.
bResetStatistics	BOOL	TRUE setzt die Min-, Max-, und Hold- Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.1.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Harmonics_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Harmonics_InitPars [▶ 136]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.1.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.1.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fBaseFreq   : LREAL;
    fBandwidth  : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fBaseFreq	LREAL	Frequenz der ersten Harmonischen.
fBandwidth	LREAL	Gesamte Bandweite der einzelnen RMS-Bänder.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.1.5 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.2 FB_PMA_PowerValues_1Ph

Der Funktionsbaustein FB_PMA_PowerValues_1Ph berechnet die Leistungswerte des angeschlossenen Verbrauchers. Dazu gehören auch die Grundschwingungskomponenten und der Phasenverschiebungswinkel. Intern werden die einzelnen Harmonischen und deren Phasenlage für die Berechnungen ermittelt.

Alternativ kann der Funktionsbaustein FB_PMA_PowerValues_Period_1Ph [▶ 50] für die Berechnung eingesetzt werden. Dieser verwendet einfachere Berechnungsverfahren für eine erhöhte Dynamik.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_PowerValues_1Ph
VAR_INPUT
    nOwnID                : UDINT;
    tTransferTimeout      : LTIME := LTIME#500US;
    stInitPars            : ST_PMA_PowerValues_InitPars;
END_VAR
VAR_OUTPUT
    bError                : BOOL;
    ipResultMessage       : I_TcMessage;
    bNewResult            : BOOL;
    nCntResults           : ULINT;
    fApparentPower        : LREAL;
    fApparentPower_1      : LREAL;
    fApparentPower_1_Min  : LREAL;
    fApparentPower_1_Max  : LREAL;
    fActivePower          : LREAL;
    fActivePower_Min      : LREAL;
    fActivePower_Max      : LREAL;
    fReactivePower_d      : LREAL;
    fReactivePower_1      : LREAL;
    fReactivePower_1_Min  : LREAL;
    fReactivePower_1_Max  : LREAL;
    fTotalReactivePower   : LREAL;
    fPhi                  : LREAL;
    fCosPhi               : LREAL;
    fPowerFactor          : LREAL;
    bValidStatistics      : BOOL;
END_VAR
VAR_OUTPUT PERSISTENT
    stEnergy_Pos          : ST_PMA_Energy;
    stEnergy_Neg          : ST_PMA_Energy;
    stEnergy_Res          : ST_PMA_Energy;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 69]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_PowerValues_InitPars [▶ 137]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
fApparentPower	LREAL	Gesamt-Scheinleistung
fApparentPower_1	LREAL	Grundschwingungs-Scheinleistung
fApparentPower_1_Min	LREAL	Kleinster aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fApparentPower_1_Max	LREAL	Größter aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fActivePower	LREAL	Wirkleistung
fActivePower_Min	LREAL	Kleinster aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fActivePower_Max	LREAL	Größter aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fReactivePower_d	LREAL	Verzerrungsblindleistung
fReactivePower_1	LREAL	Grundschwingungs-Verschiebungs-Blindleistung
fReactivePower_1_Min	LREAL	Kleinster aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fReactivePower_1_Max	LREAL	Größter aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
fTotalReactivePower	LREAL	Gesamt-Blindleistung
fPhi	LREAL	Phasenverschiebungswinkel

Name	Typ	Beschreibung
fCosPhi	LREAL	CosPhi (Wirkleistung/Grundschwingungs-Scheinleistung)
fPowerFactor	LREAL	Leistungsfaktor (Wirkleistung/Gesamt-Scheinleistung)
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
stEnergy_Pos	ST_PMA_Energy [▶ 146]	Energie in positiver Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
stEnergy_Neg	ST_PMA_Energy [▶ 146]	Energie in negativer Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
stEnergy_Res	ST_PMA_Energy [▶ 146]	Resultierende Energie. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.

Methoden

Name	Beschreibung
Call [▶ 69]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 69]	Alternative zur Bausteininitialisierung
PassInputs [▶ 70]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 71]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 70]	Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cPowerValuesInitPars : ST_PMA_PowerValues_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        fBaseFreq := 50.0,
        nNumBands := 40,
        fBandwidth := 20.0,
        eWindowType := E_PMA_WindowType.HannWindow,
        fTimeLagCurrentTransformer := 0.0,
        fMinInputCurrent := 0.01);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbPowerValues : FB_PMA_PowerValues_1Ph := (nOwnID := 2, stInitPars := cPowerValuesInitPars);
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithm
fbPowerValues.Call(FALSE, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.3.2.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    bResetEnergyCalc    : BOOL;
    bResetStatistics    : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetEnergyCalc	BOOL	TRUE setzt die berechneten Werte der Energiemessung zurück.
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.2.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_PowerValues_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_PowerValues_InitPars [▶ 137]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.2.3 PassInputs

Solange eine FB_PMA_Source_1Ph [▶ 37] Instanz aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.2.4 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.2.5 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fBaseFreq        : LREAL;
    fBandwidth       : LREAL;
    fMinInputCurrent : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fBaseFreq	LREAL	Frequenz der ersten Harmonischen.
fBandwidth	LREAL	Gesamte Bandweite der einzelnen RMS-Bänder.
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.3 FB_PMA_Spectrum_1Ph

Der Funktionsbaustein FB_PMA_Spectrum_1Ph berechnet die Betragsspektren der Strom- und Spannungswerte. Diese sind für eine Analyse der Eingangssignale im Frequenzbereich geeignet.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Spectrum_1Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Spectrum_InitPars;
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 74]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Spectrum_InitPars [▶ 138]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 73]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 74]	Alternative zur Bausteininitialisierung

Name	Beschreibung
PassInputs [▶ 75]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reset [▶ 75]	Die Methode löscht alle bereits hinzugefügten Datensätze.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cSpectrumInitPars : ST_PMA_Spectrum_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        eScalingType := E_PMA_ScalingType.PeakAmplitude,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
        fDecibelThreshold := GVL_PMA.cMinArgLog10);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbSpectrum : FB_PMA_Spectrum_1Ph := (nOwnID := 2, stInitPars := cSpectrumInitPars);
    aSpectrumVoltage : ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumCurrent : ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithm
fbSpectrum.Call(ADR(aSpectrumVoltage), ADR(aSpectrumCurrent), SIZEOF(aSpectrumVoltage));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.3.3.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pMagnitudeSpectrum_U    : POINTER TO LREAL;
    pMagnitudeSpectrum_I    : POINTER TO LREAL;
    nMagnitudeSpectrumSize  : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pMagnitudeSpectrum_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.

Name	Typ	Beschreibung
pMagnitudeSpectrum_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.3.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Spectrum_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Spectrum_InitPars [▶ 138]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.3.3 PassInputs

Solange eine FB_PMA_Source_1Ph [▶ 37] Instanz aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.3.4 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4 FB_PMA_Spectrum_Quantiles_1Ph

Der Funktionsbaustein FB_PMA_Spectrum_Quantiles_1Ph berechnet die Betragsspektren der Strom- und Spannungswerte wie der Funktionsbaustein FB_PMA_Spectrum_1Ph [▶ 71]. Zusätzlich können p-Quantile der Verteilung des Spektrums berechnet werden. Die Quantile sowie deren Anzahl lassen sich individuell konfigurieren.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_1Ph [▶ 37] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600

FFT-Länge		Fensterlänge	Pufferlänge
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Der Baustein berücksichtigt alle Eingangswerte seit der Instanziierung. Wenn seit dem Start die Reset [▶ 81]-Methode aufgerufen wurde, werden alle Eingangswerte seit deren letzten Aufruf berücksichtigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Spectrum_Quantiles_1Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Spectrum_Quantiles_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 79]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Spectrum_Quantiles_InitPars [▶ 139]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 78]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
CallEx [▶ 80]	Um die CPU-Auslastung zu minimieren, kann es erforderlich sein auf die CallEx-Methode zurückzugreifen. Es werden, im Gegensatz zur Call-Methode, nicht nach jeder Spektrum-Berechnung die Quantile berechnet, sondern erst nach einer konfigurierbaren Anzahl von Berechnungen.
Init [▶ 79]	Alternative zur Bausteininitialisierung
PassInputs [▶ 79]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 81]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 81]	Die Methode löscht alle bereits hinzugefügten Datensätze. Alternativ kann das automatische Zurücksetzen an der CallEx-Methode verwendet werden.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cSpectrumQuantilesInitPars : ST_PMA_Spectrum_Quantiles_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        eScalingType := E_PMA_ScalingType.PeakAmplitude,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
        fDecibelThreshold := GVL_PMA.cMinArgLog10,
        fMinBinnedVoltage := 0.0,
        fMaxBinnedVoltage := 300,
        fMinBinnedCurrent := 0.0,
        fMaxBinnedCurrent := 2,
        nBins := 10,
        nNumQuantiles := 2,
        aQuantiles := [0.5, 0.9]);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_1Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbSpectrumQuantiles : FB_PMA_Spectrum_Quantiles_1Ph := (nOwnID := 2, stInitPars := cSpectrumQuan
tilesInitPars);
    aSpectrumVoltage : ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumCurrent : ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumQuantilesVoltage : ARRAY[1..cFFT_Length/2 + 1, 1..2] OF LREAL;
    aSpectrumQuantilesCurrent : ARRAY[1..cFFT_Length/2 + 1, 1..2] OF LREAL;
    bNewResult_Spectrum : BOOL;
    bNewResult_Quantiles : BOOL;
END_VAR
```

```
// Call source
fbSource.Call(ADR(aVoltage), ADR(aCurrent), SIZEOF(aVoltage), 0);

// Call algorithm
fbSpectrumQuantiles.CallEx(
    5,
    FALSE,
    ADR(aSpectrumVoltage),
    ADR(aSpectrumCurrent),
    SIZEOF(aSpectrumVoltage),
    ADR(aSpectrumQuantilesVoltage),
    ADR(aSpectrumQuantilesCurrent),
    SIZEOF(aSpectrumQuantilesVoltage),
    ADR(bNewResult_Spectrum),
    ADR(bNewResult_Quantiles));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.2.3.4.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Eine Alternative stellt die CallEx [▶ 80]-Methode dar. Sie berechnet die Quantile erst nach einer definierten Zahl von Ergebnissen aus der Spektrum-Berechnung, um die CPU-Auslastung zu minimieren.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pMagnitudeSpectrum_U   : POINTER TO LREAL;
    pMagnitudeSpectrum_I   : POINTER TO LREAL;
    nMagnitudeSpectrumSize : UDINT
    pSpectrumQuantiles_U   : POINTER TO LREAL;
    pSpectrumQuantiles_I   : POINTER TO LREAL;
    nSpectrumQuantilesSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pMagnitudeSpectrum_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
pMagnitudeSpectrum_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.
pSpectrumQuantiles_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pSpectrumQuantiles_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nSpectrumQuantilesSize	UDINT	Gibt die Größe des Ausgangsarrays für eine Quantilberechnung an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Spectrum_Quantiles_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Spectrum_Quantiles_InitPars [▶ 139]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_1Ph [▶ 37] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4.4 CallEx

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Im Gegensatz zur Call [▶ 78]-Methode werden bei der CallEx-Methode eine variable Anzahl von Ergebnissen aus der Spektrum-Berechnung gesammelt und erst anschließend die Quantile berechnet. Dies kann erforderlich sein, um die CPU-Auslastung zu minimieren.

Syntax

```
METHOD CallEx : BOOL
VAR_INPUT
    nAppendData            : UDINT;
    bResetData             : BOOL;
    pMagnitudeSpectrum_U   : POINTER TO LREAL;
    pMagnitudeSpectrum_I   : POINTER TO LREAL;
    nMagnitudeSpectrumSize : UDINT
    pSpectrumQuantiles_U   : POINTER TO LREAL;
    pSpectrumQuantiles_I   : POINTER TO LREAL;
    nSpectrumQuantilesSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nAppendData	UDINT	Anzahl der zu berechnenden Spektren bis eine Berechnung der Quantile erfolgt. Ein Wert von 1 bedeutet dass nach jedem Ergebnis der Spektrum-Berechnung die Quantile berechnet werden.
bResetData	BOOL	Automatisches Zurücksetzen der Datensätze nach jeder Quantil-Berechnung
pMagnitudeSpectrum_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
pMagnitudeSpectrum_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.
pSpectrumQuantiles_U	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.

Name	Typ	Beschreibung
pSpectrumQuantiles_I	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nSpectrumQuantilesSize	UDINT	Gibt die Größe des Ausgangsarrays für eine Quantilberechnung an.

Rückgabewert

Name	Typ	Beschreibung
CallEx	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4.5 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Alternativ kann das automatische Zurücksetzen an der CallEx [▶ 80]-Methode verwendet werden.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.2.3.4.6 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    aQuantiles      : ARRAY[0..GVL_PMA.cMaxQuantiles - 1] OF LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
aQuantiles	ARRAY[0..GVL_PMA.cMaxQuantiles-1] OF LREAL	Gibt die Quantilgrenze an. Sie muss zwischen 0.0 und 1.0 liegen. Beispielsweise entspricht 0.2 dem 20%-Quantil.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3 Funktionsbausteine Dreiphasig

5.3.1 FB_PMA_Source_3Ph

Der Funktionsbaustein FB_PMA_Source_3Ph schreibt Daten aus einem externen SPS-Datenpuffer in einen MultiArray-Puffer.

Er häuft ununterbrochen Eingangsdaten an, bis die Größe des MultiArrays erreicht ist. Wenn das MultiArray komplett gefüllt ist, wird es an einen Funktionsbaustein mit der Ziel-Analyse-ID übergeben.

Die Ausgangspuffer werden für die Funktionsbausteine bereitgestellt, deren ID in dem Array von Ziel-IDs eingetragen ist. Sie enthalten die Werte von Strom und Spannung. Die Anforderungen an die Größe der Ausgangspuffer können je nach verwendeten Analysebaustein unterschiedlich sein. Sie sind entweder von der verwendeten FFT-Länge oder der Puffergröße abhängig.

Syntax

Definition:

```
FUNCTION BLOCK FB_PMA_Source_3Ph
VAR_INPUT
    nOwnID            : UDINT;
    aDestIDs          : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers    : UDINT := 4;
    tTransferTimeout  : LTIME := LTIME#40US;
    stInitPars        : ST_PMA_Source_InitPars;
END_VAR
VAR_OUTPUT
    bError            : BOOL;
    ipResultMessage   : I_TcMessage;
    bNewResults       : BOOL;
    nCntResults       : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 84]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
nResultBuffers	UDINT	Anzahl von MultiArray-Puffern, die für die Ergebnisse initialisiert werden.

Name	Typ	Beschreibung
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray).
stInitPars	ST_PMA_Source_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse bereitgestellt wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 83]	Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben.
ResetData [▶ 84]	Mit dieser Methode können die Daten, die sich aktuell im Puffer befinden, zurückgesetzt werden.
ResetAnalysisChain [▶ 85]	Durch Aufruf dieser Methode wird automatisch ein Reset aller Algorithmen der vollständigen Analysekette durchgeführt.
Init [▶ 84]	Alternative zur Bausteininitialisierung

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Werte in den Ausgangspuffer zu schreiben. Der Ausgangspuffer wird verschickt, sobald dieser gefüllt ist.

Syntax

```
METHOD CALL : BOOL
VAR_INPUT
    pBuffer_UL1         : POINTER TO LREAL;
    pBuffer_UL2         : POINTER TO LREAL;
    pBuffer_UL3         : POINTER TO LREAL;
    pBuffer_IL1         : POINTER TO LREAL;
    pBuffer_IL2         : POINTER TO LREAL;
    pBuffer_IL3         : POINTER TO LREAL;
    nDataInSizePerCh    : UDINT;
    nOptionPars         : DWORD;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pBuffer_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array von Spannungswerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
pBuffer_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array von Stromwerten. Diese können einzeln oder als Oversampling-Array hinzugefügt werden.
nDataInSizePerCh	UDINT	Gibt die Größe eines einzelnen Eingangspuffers in Bytes an.

5.3.1.2 ResetData

Mit dieser Methode können die Daten, die sich aktuell im Puffer befinden, zurückgesetzt werden.

Syntax

```
METHOD ResetData : BOOL
VAR_INPUT
ENC_VAR
```

Rückgabewert

Name	Typ	Beschreibung
ResetData	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.1.3 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID          : UDINT;
    aDestIDs        : ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT;
    nResultBuffers  : UDINT := 4;
    stInitPars      : ST_PMA_Source_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
aDestIDs	ARRAY[1..GVL_PMA.cMA_MaxDest] OF UDINT	Die Ergebnisdaten werden an die hier als Array angegebenen IDs anderer Bausteininstanzen weitergeleitet.
nResultBuffers	UDINT	Anzahl der zur Verfügung stehenden Multi Arrays.
stInitPars	ST_PMA_Source_InitPars [▶ 131]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.1.4 ResetAnalysisChain

Durch Aufruf dieser Methode wird automatisch ein Reset aller Algorithmen der vollständigen Analysekette durchgeführt. Intern wird jeweils ein ResetData() vor der Übernahme des neuen Datensatzes ausgeführt.

Soll die Analysekette nur für einen bestimmten Zeitraum aktiv sein, so bietet diese Methode die Möglichkeit alle Algorithmen vor der nächsten Ausführung zurückzusetzen.

Es können Fehler beim Aufruf einer Input Methode auftreten und Unterbrechungen der Zeitreihensammlung verursachen. Falls die folgenden Algorithmen der Analysekette Spektren berechnen, so kann im Falle eines Fehlers beim Aufruf einer Input Methode die ResetAnalysisChain() Methode aufgerufen werden. Denn es ist nicht möglich korrekte Spektren anhand zerstückelter Zeitreihen zu berechnen.

Syntax

```
METHOD ResetAnalysisChain : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
ResetAnalysisChain	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2 Basierend auf der Signalperiode

5.3.2.1 FB_PMA_Frequency_Period_3Ph

Der Funktionsbaustein FB_PMA_Frequency_Period_3Ph berechnet die Grundfrequenz des gegebenen Eingangssignals für alle drei Phasen. Dafür wird das Signal zunächst mit einem Butterworth-Tiefpassfilter gefiltert. Anschließend werden aus den gefilterten Werten die Nulldurchgänge des Eingangssignals ermittelt und aus deren Differenz die Frequenz berechnet. Die Ergebnisse beziehen sich, abhängig von der Konfiguration, auf eine oder mehrere Perioden. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Arrays.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Frequency_Period_3Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Frequency_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
    aFreq               : ARRAY[0..2] OF LREAL;
    aFreq_Min           : ARRAY[0..2] OF LREAL;
    aFreq_Max           : ARRAY[0..2] OF LREAL;
    aRocof              : ARRAY[0..2] OF LREAL;
    eRotDirection       : E_PMA_RotationalDirection_3Ph;
    bValidStatistics    : BOOL;
    aOutOfRange         : ARRAY[0..2] OF  BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 88]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Frequency_Period_InitPars [▶ 133]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aFreq	ARRAY[0..2] OF LREAL	Frequenz, die durch zwei oder mehr Nulldurchgänge ermittelt wurde.
aFreq_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fFreq. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.

Name	Typ	Beschreibung
aFreq_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fFreq. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRocof	ARRAY[0..2] OF LREAL	Rate of change of frequency (ROCOF)
eRotDirection	E_PMA_RotationalDirection_3Ph [▶ 143]	Drehrichtung
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
aOutOfRange	ARRAY[0..2] OF	TRUE, sobald der Eingangswert oder die Frequenz nicht innerhalb der konfigurierten Grenzen ist.

Methoden

Name	Beschreibung
Call [▶ 88]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 88]	Alternative zur Bausteininitialisierung
PassInputs [▶ 89]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 89]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 89]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
END_VAR
VAR
    aVoltage AT%I* : ARRAY [0..2] OF ARRAY[1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY [0..2] OF ARRAY[1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_3Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithms
fbFrequency.Call(FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.2.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.1.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Frequency_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
stInitPars	ST_PMA_Frequency_Period_InitPars [▶ 133]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.1.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.1.4 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.1.5 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinFreq          : LREAL;
    fMaxFreq          : LREAL;
    nPeriods          : UDINT;
    nFilterOrder      : UINT;
    fCutoff           : LREAL;
    eInputSelect      : E_PMA_InputSelect;
    fMinInput         : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinFreq	LREAL	Minimal zu erwartende Messfrequenz
fMaxFreq	LREAL	Maximal zu erwartende Messfrequenz
nPeriods	UDINT	Anzahl an Perioden, die Einfluss auf die Berechnung haben. (Periodenlänge = Samplerate/Frequenz)
nFilterOrder	UINT	Gibt die Ordnung des Tiefpassfilters an. Bei der Einstellung muss die Stabilität des Filters berücksichtigt werden. Nur Werte bis zur zehnten Ordnung sind zulässig.
fCutoff	LREAL	Gibt die Grenzfrequenz des Tiefpassfilters an.
eInputSelect	E_PMA_InputSelect [▶ 146]	Hier kann konfiguriert werden, ob die Frequenz der Spannung oder des Stroms berechnet werden soll.
fMinInput	LREAL	Mindesteingangsgröße (RMS) über eine Periode. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.2 FB_PMA_BasicValues_Period_3Ph

Der Funktionsbaustein FB_PMA_BasicValues_Period_3Ph berechnet Analysewerte für den zeitlichen Signalverlauf von Strom und Spannung in einem dreiphasigen System. Dazu gehören Mittelwert, RMS-Wert, Spitzenwert, Gleichrichtwert, Crest-Faktor sowie Formfaktor jeweils für die einzelnen Ströme und Spannungen. Zusätzlich werden die Spannungswerte zwischen den einzelnen Phasen berechnet. Die Ergebnisse beziehen sich auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 93]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Arrays.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_BasicValues_Period_3Ph
VAR_INPUT
    nOwnID                : UDINT;
    tTransferTimeout      : LTIME := LTIME#500US;
    stInitPars            : ST_PMA_BasicValues_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError                : BOOL;
    ipResultMessage       : I_TcMessage;
    bNewResult            : BOOL;
    nCntResults           : ULINT;
    aMeanValue_U          : ARRAY[0..2] OF LREAL;
    aRMS_U                : ARRAY[0..2] OF LREAL;
    aRMS_U_Min            : ARRAY[0..2] OF LREAL;
    aRMS_U_Max            : ARRAY[0..2] OF LREAL;
    aRMS_U_PP             : ARRAY[0..2] OF LREAL
    aPeakValue_U          : ARRAY[0..2] OF LREAL;
    aPeakHold_U           : ARRAY[0..2] OF LREAL;
    aRectifiedValue_U     : ARRAY[0..2] OF LREAL;
    aCrestFactor_U        : ARRAY[0..2] OF LREAL;
    aFormFactor_U         : ARRAY[0..2] OF LREAL;
    aMeanValue_I          : ARRAY[0..2] OF LREAL,
    aRMS_I                : ARRAY[0..2] OF LREAL,
    aRMS_I_Min            : ARRAY[0..2] OF LREAL,
    aRMS_I_Max            : ARRAY[0..2] OF LREAL,
    aPeakValue_I          : ARRAY[0..2] OF LREAL,
    aPeakHold_I           : ARRAY[0..2] OF LREAL;
    aRectifiedValue_I     : ARRAY[0..2] OF LREAL;
    aCrestFactor_I        : ARRAY[0..2] OF LREAL;
    aFormFactor_I         : ARRAY[0..2] OF LREAL;
    bValidStatistics      : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 94]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_BasicValues_Period_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aMeanValue_U	ARRAY[0..2] OF LREAL	Mittelwert der Spannung über n Perioden
aRMS_U	ARRAY[0..2] OF LREAL	Effektivwert der Spannung über n Perioden

Name	Typ	Beschreibung
aRMS_U_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fRMS_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRMS_U_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fRMS_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRMS_UPP	ARRAY[0..2] OF LREAL	Effektivwert der Spannung zwischen zwei Phasen. Index 0: L1-L2, Index 1: L2-L3, Index 2: L3-L1
aPeakValue_U	ARRAY[0..2] OF LREAL	Spitzenwert der Spannung über n Perioden
aPeakHold_U	ARRAY[0..2] OF LREAL	Allzeit-Spitzenwert der Spannung über n Perioden. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRectifiedValue_U	ARRAY[0..2] OF LREAL	Gleichrichtwert der Spannung über n Perioden
aCrestFactor_U	ARRAY[0..2] OF LREAL	Scheitelfaktor (Crest-Faktor) der Spannung (Spitzenwert/Effektivwert)
aFormFactor_U	ARRAY[0..2] OF LREAL	Formfaktor der Spannung (Effektivwert/ Gleichrichtwert)
aMeanValue_I	ARRAY[0..2] OF LREAL	Mittelwert des Stroms über n Perioden
aRMS_I	ARRAY[0..2] OF LREAL	Effektivwert des Stroms über n Perioden
aRMS_I_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fRMS_I. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRMS_I_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fRMS_I. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aPeakValue_I	ARRAY[0..2] OF LREAL	Spitzenwert des Stroms über n Perioden
aPeakHold_I	ARRAY[0..2] OF LREAL	Allzeit-Spitzenwert des Stroms. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aRectifiedValue_I	ARRAY[0..2] OF LREAL	Gleichrichtwert des Stroms über n Perioden
aCrestFactor_I	ARRAY[0..2] OF LREAL	Scheitelfaktor (Crest-Faktor) des Stroms (Spitzenwert / Effektivwert)
aFormFactor_I	ARRAY[0..2] OF LREAL	Formfaktor des Stroms (Effektivwert / Gleichrichtwert)
bValidStatistics	BOOL	TRUE, wenn die Min-, Max- und Hold-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 93]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 94]	Alternative zur Bausteininitialisierung
PassInputs [▶ 95]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 95]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 95]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cBasicValuesInitPars : ST_PMA_BasicValues_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinInputCurrent := 0.01,
        nPeriods := 1);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_3Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbBasicValues : FB_PMA_BasicValues_Period_3Ph := (nOwnID := 3, stInitPars := cBasicValuesInitPar
s);
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithms
fbFrequency.Call(FALSE);
fbBasicValues.Call(fbFrequency.aFreq[0], FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.2.2.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq             : LREAL;
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet um zu Beginn einer Periode die Periodenlänge zu ermitteln. Es kann der Ausgang des Funktionsbausteins FB_PMA_Frequency_Period_3Ph [▶ 85] verwendet werden.
bResetStatistics	BOOL	TRUE setzt die Min-, Max-, und Hold- Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.2.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_BasicValues_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_BasicValues_Period_InitPars [▶ 132]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.2.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.2.4 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.2.5 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinInputCurrent   : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.3 FB_PMA_PowerValues_Period_3Ph

Der Funktionsbaustein FB_PMA_PowerValues_Period_3Ph berechnet die Leistungswerte des angeschlossenen Verbrauchers in einem dreiphasigen Netz. Dazu gehören auch die Grundschwingungskomponenten und der Phasenverschiebungswinkel. Hierfür werden neben dem Signalverlauf im Zeitbereich nur die ersten Harmonischen des Eingangssignals zur Berechnung herangezogen. Der Vorteil dieser Algorithmen ist die hohe Dynamik der Berechnungen. Die Ergebnisse beziehen sich auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 88]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Alternativ kann der Funktionsbaustein FB_PMA_PowerValues_3Ph [▶ 112] verwendet werden. Dieser nutzt intern die einzelnen Harmonischen zur Berechnung der Leistungswerte.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Werten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_PowerValues_Period_3Ph
VAR_INPUT
    nOwnID                  : UDINT;
    tTransferTimeout        : LTIME := LTIME#500US;
    stInitPars              : ST_PMA_PowerValues_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError                  : BOOL;
    ipResultMessage         : I_TcMessage;
    bNewResult              : BOOL;
    nCntResults             : ULINT;
    aApparentPower          : ARRAY[0..2] OF LREAL;
    aApparentPower_1        : ARRAY[0..2] OF LREAL;
    aApparentPower_1_Min    : ARRAY[0..2] OF LREAL;
    aApparentPower_1_Max    : ARRAY[0..2] OF LREAL;
    aActivePower            : ARRAY[0..2] OF LREAL;
    aActivePower_Min        : ARRAY[0..2] OF LREAL;
    aActivePower_Max        : ARRAY[0..2] OF LREAL;
    aReactivePower_d        : ARRAY[0..2] OF LREAL;
    aReactivePower_1        : ARRAY[0..2] OF LREAL;
    aReactivePower_1_Min    : ARRAY[0..2] OF LREAL;
    aReactivePower_1_Max    : ARRAY[0..2] OF LREAL;
    aTotalReactivePower     : ARRAY[0..2] OF LREAL;
    aPhi                    : ARRAY[0..2] OF LREAL;
    aCosPhi                 : ARRAY[0..2] OF LREAL;
    aPowerFactor            : ARRAY[0..2] OF LREAL;
    fPowerQualityFactor     : LREAL;
    fSumApparentPower       : LREAL;
    fSumActivePower         : LREAL;
    fSumTotalReactivePower  : LREAL;
    fSumReactivePower_1     : LREAL;
    bValidStatistics        : BOOL;
END_VAR
VAR_OUTPUT PERSISTENT
    aEnergy_Pos             : ARRAY[0..2] OF ST_PMA_Energy;
    aEnergy_Neg             : ARRAY[0..2] OF ST_PMA_Energy;
    aEnergy_Res             : ARRAY[0..2] OF ST_PMA_Energy;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 100]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_PowerValues_Period_InitPars [▶ 134]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aApparentPower	ARRAY[0..2] OF LREAL	Gesamt-Scheinleistung
aApparentPower_1	ARRAY[0..2] OF LREAL	Grundschwingungs-Scheinleistung
aApparentPower_1_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aApparentPower_1_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aActivePower	ARRAY[0..2] OF LREAL	Wirkleistung
aActivePower_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aActivePower_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aReactivePower_d	ARRAY[0..2] OF LREAL	Verzerrungsblindleistung
aReactivePower_1	ARRAY[0..2] OF LREAL	Grundschwingungs-Verschiebungs-Blindleistung
aReactivePower_1_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aReactivePower_1_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTotalReactivePower	ARRAY[0..2] OF LREAL	Gesamt- Blindleistung
aPhi	ARRAY[0..2] OF LREAL	Phasenverschiebungswinkel

Name	Typ	Beschreibung
aCosPhi	ARRAY[0..2] OF LREAL	CosPhi (Wirkleistung/Grundschwingungs-Scheinleistung)
aPowerFactor	ARRAY[0..2] OF LREAL	Leistungsfaktor (Wirkleistung/Gesamt-Scheinleistung)
fPowerQualityFactor	LREAL	Power Quality Factor. Stellt die Qualität der Spannungsversorgung vereinfacht in einem Wertebereich zwischen 0 und 1 dar. Miteinbezogen wird die Frequenz, die Effektivwerte der Spannungen, der THD der Spannungen sowie optional die Spannungsunsymmetrie.
fSumApparentPower	LREAL	Summe der Gesamt-Scheinleistung aller Phasen.
fSumActivePower	LREAL	Summe der Wirkleistung aller Phasen.
fSumTotalReactivePower	LREAL	Summe der Gesamt- Blindleistung aller Phasen.
fSumReactivePower_1	LREAL	Summe der Beträge der Grundschwingungs-Verschiebungs-Blindleistung aller Phasen.
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
aEnergy_Pos	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Energie in positiver Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
aEnergy_Neg	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Energie in negativer Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
aEnergy_Res	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Resultierende Energie. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.

Methoden

Name	Beschreibung
Call [▶ 99]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 100]	Alternative zur Bausteininitialisierung
PassInputs [▶ 100]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 101]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 101]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
```

```
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cPowerValuesInitPars : ST_PMA_PowerValues_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinInputCurrent := 0.01,
        nPeriods := 1);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_3Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbPowerValues : FB_PMA_PowerValues_Period_3Ph := (nOwnID := 3, stInitPars := cPowerValuesInitPar
s);
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithms
fbFrequency.Call(FALSE);
fbPowerValues.Call(fbFrequency.aFreq[0], FALSE, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.2.3.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq             : LREAL;
    bResetEnergyCalc  : LREAL;
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet, um zu Beginn einer Periode die Periodenlänge zu ermitteln. Es kann der Ausgang des Funktionsbausteins FB_PMA_Frequency_Period_3Ph [▶ 85] verwendet werden.
bResetEnergyCalc	LREAL	TRUE setzt die berechneten Werte der Energiemessung zurück.
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.3.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT
    stInitPars  : ST_PMA_PowerValues_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_PowerValues_Period_InitPars [▶ 134]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.3.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.3.4 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.3.5 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fMinInputCurrent  : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.4 FB_PMA_Harmonics_Period_3Ph

Der Funktionsbaustein FB_PMA_Harmonics_Period_3Ph berechnet die Harmonischen von Strom und Spannung. Zusätzlich wird aus den Harmonischen der THD der Eingangsgrößen berechnet. Im Gegensatz zum Funktionsbaustein FB_PMA_Harmonics_3Ph [▶ 106] beziehen sich die Ergebnisse auf eine konfigurierbare Anzahl von Signalperioden. Die Periodendauer bezieht sich auf die zu Periodenbeginn angegebene Frequenz am Eingang der Call [▶ 88]-Methode. Die statistischen Ergebnisse beziehen sich auf die gesamte Laufzeit bzw. den Zeitpunkt, an dem zuletzt das Zurücksetzen der statistischen Ergebnisse erfolgt ist.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Dieser kann sowohl eine oder mehrere Signalperioden beinhalten oder auch einzelne Fragmente aus Oversampling-Werten.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_PowerValues_Period_3Ph
VAR_INPUT
    nOwnID                  : UDINT;
    tTransferTimeout        : LTIME := LTIME#500US;
    stInitPars              : ST_PMA_Harmonics_Period_InitPars;
END_VAR
VAR_OUTPUT
    bError                  : BOOL;
    ipResultMessage         : I_TcMessage;
    bNewResult              : BOOL;
    nCntResults             : ULINT;
    aTHD_U                  : ARRAY[0..2] OF LREAL;
    aTHD_U_Min              : ARRAY[0..2] OF LREAL;
    aTHD_U_Max              : ARRAY[0..2] OF LREAL;
    aTHD_I                  : ARRAY[0..2] OF LREAL;
    aTHD_I_Min              : ARRAY[0..2] OF LREAL;
    aTHD_I_Max              : ARRAY[0..2] OF LREAL;
    bValidStatistics        : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 105]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Harmonics_Period_InitPars [▶ 135]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der obigen Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.

Name	Typ	Beschreibung
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aTHD_U	ARRAY[0..2] OF LREAL	THD der Spannung. Die Ausgabe erfolgt in Prozent.
aTHD_U_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_U_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_I	ARRAY[0..2] OF LREAL	THD des Stroms. Die Ausgabe erfolgt in Prozent.
aTHD_U_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_U_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 104]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 105]	Alternative zur Bausteininitialisierung
PassInputs [▶ 105]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reset [▶ 106]	Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cOversamples);
    cFrequencyInitPars : ST_PMA_Frequency_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinFreq := 45.0,
        fMaxFreq := 55.0,
        nPeriods := 1,
        nFilterOrder := 2,
        fCutOff := 70.0,
        eInputSelect := E_PMA_InputSelect.Voltage,
        fMinInput := 200.0);
    cPowerValuesInitPars : ST_PMA_PowerValues_Period_InitPars := (
        nBufferLength := cOversamples,
        fSampleRate := cOversamples * 1000,
        fMinInputCurrent := 0.01,
        nPeriods := 1);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2,3], stInitPars := cSourceInitPars);
    fbFrequency : FB_PMA_Frequency_Period_3Ph := (nOwnID := 2, stInitPars := cFrequencyInitPars);
    fbPowerValues : FB_PMA_PowerValues_Period_3Ph := (nOwnID := 3, stInitPars := cPowerValuesInitPar
s);
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithms
fbFrequency.Call(FALSE);
fbPowerValues.Call(fbFrequency.aFreq[0], FALSE, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.2.4.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    fFreq              : LREAL;
    pHarmonicsRMS_UL1  : POINTER TO LREAL;
    pHarmonicsRMS_UL2  : POINTER TO LREAL;
    pHarmonicsRMS_UL3  : POINTER TO LREAL;
    pHarmonicsRMS_IL1  : POINTER TO LREAL;
    pHarmonicsRMS_IL2  : POINTER TO LREAL;
    pHarmonicsRMS_IL3  : POINTER TO LREAL;
    nHarmonicsRMSSize  : UDINT;
    bResetStatistics   : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fFreq	LREAL	Aktuelle Frequenz des Eingangssignals. Wird verwendet, um zu Beginn einer Periode die Periodenlänge zu ermitteln. Es kann der Ausgang des Funktionsbausteins FB_PMA_Frequency_Period_3Ph [▶ 85] verwendet werden.
pHarmonicsRMS_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pHarmonicsRMS_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nHarmonicsRMSSize	UDINT	Gibt die Größe eines Ausgangsarrays für die Harmonischen an.

Name	Typ	Beschreibung
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.4.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT
    stInitPars  : ST_PMA_Harmonics_Period_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Harmonics_Period_InitPars [▶ 135]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.4.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend, wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.2.4.4 Reset

Mit der Methode werden die aktuellen Berechnungen zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3 Basierend auf dem Frequenzbereich

5.3.3.1 FB_PMA_Harmonics_3Ph

Der Funktionsbaustein FB_PMA_Harmonics_3Ph berechnet die RMS-Bänder der einzelnen Harmonischen von Strom und Spannung in einem dreiphasigen System. Zusätzlich wird aus den berechneten RMS-Bändern der THD der Eingangsgrößen berechnet.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUCNTION BLOCK FB_PMA_Harmonics_3Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Harmonics_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
    aTHD_U              : ARRAY[0..2] OF LREAL;
    aTHD_U_Min          : ARRAY[0..2] OF LREAL;
    aTHD_U_Max          : ARRAY[0..2] OF LREAL;
    aTHD_I              : ARRAY[0..2] OF LREAL;
    aTHD_I_Min          : ARRAY[0..2] OF LREAL;
    aTHD_I_Max          : ARRAY[0..2] OF LREAL;
    bValidStatistics    : BOOL;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 110]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Harmonics_InitPars [▶ 136]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aTHD_U	ARRAY[0..2] OF LREAL	THD der Spannung. Die Ausgabe erfolgt in Prozent.

Name	Typ	Beschreibung
aTHD_U_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_U_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_I	ARRAY[0..2] OF LREAL	THD des Stroms. Die Ausgabe erfolgt in Prozent.
aTHD_U_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aTHD_U_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von aTHD_U. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
bValidStatistics	BOOL	TRUE, wenn die Min-, Max- und Hold-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.

Methoden

Name	Beschreibung
Call [▶ 109]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 110]	Alternative zur Bausteininitialisierung
PassInputs [▶ 110]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 111]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 111]	Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cHarmonicsInitPars : ST_PMA_Harmonics_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        fBaseFreq := 50.0,
        nNumBands := 40,
        fBandwidth := 20.0,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
        fDecibelThreshold := GVL_PMA.cMinArgLog10,
        bTransformToPercent := TRUE);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbHarmonics  : FB_PMA_Harmonics_3Ph := (nOwnID := 2, stInitPars := cHarmonicsInitPars);
    aHarmonicsVoltage : ARRAY[0..2] OF ARRAY[1..40] OF LREAL;
    aHarmonicsCurrent : ARRAY[0..2] OF ARRAY[1..40] OF LREAL;
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
```

```
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithm
fbHarmonics.Call(
    ADR(aHarmonicsVoltage[0]),
    ADR(aHarmonicsVoltage[1]),
    ADR(aHarmonicsVoltage[2]),
    ADR(aHarmonicsCurrent[0]),
    ADR(aHarmonicsCurrent[1]),
    ADR(aHarmonicsCurrent[2]),
    SIZEOF(aHarmonicsVoltage[0]),
    FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.3.1.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pHarmonicsRMS_UL1  : POINTER TO LREAL;
    pHarmonicsRMS_UL2  : POINTER TO LREAL;
    pHarmonicsRMS_UL3  : POINTER TO LREAL;
    pHarmonicsRMS_IL1  : POINTER TO LREAL;
    pHarmonicsRMS_IL2  : POINTER TO LREAL;
    pHarmonicsRMS_IL3  : POINTER TO LREAL;
    nHarmonicsRMSSize  : UDINT;
    bResetStatistics   : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pHarmonicsRMS_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pHarmonicsRMS_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: Anzahl Harmonische. Wenn die einzelnen Harmonischen nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nHarmonicsRMSSize	UDINT	Gibt die Größe eines Ausgangsarrays für die Harmonischen an.
bResetStatistics	BOOL	TRUE setzt die Min-, Max-, und Hold- Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.1.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Harmonics_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Harmonics_InitPars [▶ 136]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.1.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.1.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fBaseFreq      : LREAL;
    fBandwidth     : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fBaseFreq	LREAL	Frequenz der ersten Harmonischen.
fBandwidth	LREAL	Gesamte Bandweite der einzelnen RMS-Bänder.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.1.5 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.2 FB_PMA_PowerValues_3Ph

Der Funktionsbaustein FB_PMA_PowerValues_3Ph berechnet die Leistungswerte des angeschlossenen Verbrauchers in einem dreiphasigen Netz. Dazu gehören auch die Grundschwingungskomponenten und der Phasenverschiebungswinkel. Intern werden die einzelnen Harmonischen und deren Phasenlage für die Berechnungen ermittelt.

Alternativ kann der Funktionsbaustein FB_PMA_PowerValues_Period_3Ph [▶ 96] für die Berechnung eingesetzt werden. Dieser verwendet einfachere Berechnungsverfahren für eine erhöhte Dynamik.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen des wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_PowerValues_3Ph
VAR_INPUT
    nOwnID                  : UDINT;
    tTransferTimeout        : LTIME := LTIME#500US;
    stInitPars              : ST_PMA_PowerValues_InitPars;
END_VAR
VAR_OUTPUT
    bError                  : BOOL;
    ipResultMessage         : I_TcMessage;
    bNewResult              : BOOL;
    nCntResults             : ULINT;
    aApparentPower          : ARRAY[0..2] OF LREAL;
    aApparentPower_1        : ARRAY[0..2] OF LREAL;
    aApparentPower_1_Min    : ARRAY[0..2] OF LREAL;
    aApparentPower_1_Max    : ARRAY[0..2] OF LREAL;
    aActivePower            : ARRAY[0..2] OF LREAL;
    aActivePower_Min        : ARRAY[0..2] OF LREAL;
    aActivePower_Max        : ARRAY[0..2] OF LREAL;
    aReactivePower_d        : ARRAY[0..2] OF LREAL;
    aReactivePower_1        : ARRAY[0..2] OF LREAL;
    aReactivePower_1_Min    : ARRAY[0..2] OF LREAL;
    aReactivePower_1_Max    : ARRAY[0..2] OF LREAL;
    aTotalReactivePower     : ARRAY[0..2] OF LREAL;
    aPhi                    : ARRAY[0..2] OF LREAL;
    aCosPhi                 : ARRAY[0..2] OF LREAL;
    aPowerFactor            : ARRAY[0..2] OF LREAL;
    fSumApparentPower       : LREAL;
    fSumActivePower         : LREAL;
    fSumTotalReactivePower  : LREAL;
    fSumReactivePower_1     : LREAL;
    bValidStatistics        : BOOL;
END_VAR
VAR_OUTPUT PERSISTENT
    aEnergy_Pos             : ARRAY[0..2] OF ST_PMA_Energy;
```

```
    aEnergy_Neg             : ARRAY[0..2] OF ST_PMA_Energy;
    aEnergy_Res             : ARRAY[0..2] OF ST_PMA_Energy;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 115]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_PowerValues_InitPars [▶ 137]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.
aApparentPower	ARRAY[0..2] OF LREAL	Gesamt-Scheinleistung
aApparentPower_1	ARRAY[0..2] OF LREAL	Grundschwingungs-Scheinleistung
aApparentPower_1_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aApparentPower_1_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fApparentPower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aActivePower	ARRAY[0..2] OF LREAL	Wirkleistung
aActivePower_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aActivePower_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fActivePower. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aReactivePower_d	ARRAY[0..2] OF LREAL	Verzerrungsblindleistung
aReactivePower_1	ARRAY[0..2] OF LREAL	Grundschwingungs-Verschiebungs-Blindleistung
aReactivePower_1_Min	ARRAY[0..2] OF LREAL	Kleinster aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.
aReactivePower_1_Max	ARRAY[0..2] OF LREAL	Größter aufgetretener Wert von fReactivePower_1. Kann über bResetStatistics der Call-Methode zurückgesetzt werden.

Name	Typ	Beschreibung
aTotalReactivePower	ARRAY[0..2] OF LREAL	Gesamt- Blindleistung
aPhi	ARRAY[0..2] OF LREAL	Phasenverschiebungswinkel
aCosPhi	ARRAY[0..2] OF LREAL	CosPhi (Wirkleistung/Grundschwingungs-Scheinleistung)
aPowerFactor	ARRAY[0..2] OF LREAL	Leistungsfaktor (Wirkleistung/Gesamt-Scheinleistung)
fSumApparentPower	LREAL	Summe der Gesamt-Scheinleistung aller Phasen.
fSumActivePower	LREAL	Summe der Wirkleistung aller Phasen.
fSumTotalReactivePower	LREAL	Summe der Gesamt- Blindleistung aller Phasen.
fSumReactivePower_1	LREAL	Summe der Beträge der Grundschwingungs-Verschiebungs-Blindleistung aller Phasen.
bValidStatistics	BOOL	TRUE, wenn die Min- und Max-Werteberechnung durchgeführt wurde. Diese Werte sind gültig.
aEnergy_Pos	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Energie in positiver Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
aEnergy_Neg	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Energie in negativer Richtung. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.
aEnergy_Res	ARRAY[0..2] OF ST_PMA_Energy [▶ 146]	Resultierende Energie. Der Ausgang wird persistent gespeichert und kann über bResetEnergyCalc der Call-Methode zurückgesetzt werden.

Methoden

Name	Beschreibung
Call [▶ 115]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 115]	Alternative zur Bausteininitialisierung
PassInputs [▶ 116]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 116]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 117]	Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cPowerValuesInitPars : ST_PMA_PowerValues_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        fBaseFreq := 50.0,
        nNumBands := 40,
        fBandwidth := 20.0,
        eWindowType := E_PMA_WindowType.HannWindow,
        fTimeLagCurrentTransformer := 0.0,
        fMinInputCurrent := 0.01);
END_VAR
VAR
```

```
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbPowerValues : FB_PMA_PowerValues_3Ph := (nOwnID := 2, stInitPars := cPowerValuesInitPars);
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithm
fbPowerValues.Call(FALSE, FALSE);
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.3.2.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    bResetEnergyCalc  : BOOL;
    bResetStatistics  : BOOL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
bResetEnergyCalc	BOOL	TRUE setzt die berechneten Werte der Energiemessung zurück.
bResetStatistics	BOOL	TRUE setzt die Min/Max-Werte der Ausgänge zurück.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.2.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_PowerValues_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_PowerValues_InitPars [▶ 137]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.2.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.2.4 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    fBaseFreq        : LREAL;
    fBandwidth       : LREAL;
    fMinInputCurrent : LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
fBaseFreq	LREAL	Frequenz der ersten Harmonischen.
fBandwidth	LREAL	Gesamte Bandweite der einzelnen RMS-Bänder.
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.2.5 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Zusätzlich werden die berechneten Ausgangswerte zurückgesetzt.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.3 FB_PMA_Spectrum_3Ph

Der Funktionsbaustein FB_PMA_Spectrum_3Ph berechnet die Betragsspektren der Strom- und Spannungswerte. Diese sind für eine Analyse der Eingangssignale im Frequenzbereich geeignet.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600

FFT-Länge		Fensterlänge	Pufferlänge
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Aufgrund der Verwendung der Welch-Methode innerhalb der Berechnungen wird jeweils der aktuelle Eingangspuffer zusammen mit dem zuletzt übergebenen Puffer zur Berechnung genutzt.

Die Frequenzanalyse berücksichtigt Sprünge in der Zeitreihe. Um ein korrektes Ergebnis zu erzielen, müssen sich deswegen die letzten zwei Eingangspuffer lückenlos und ohne Sprünge aneinanderreihen.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Spectrum_3Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Spectrum_InitPars;
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 120]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Spectrum_InitPars [▶ 138]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 119]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
Init [▶ 120]	Alternative zur Bausteininitialisierung
PassInputs [▶ 121]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reset [▶ 121]	Die Methode löscht alle bereits hinzugefügten Datensätze.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cSpectrumInitPars : ST_PMA_Spectrum_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        eScalingType := E_PMA_ScalingType.PeakAmplitude,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
        fDecibelThreshold := GVL_PMA.cMinArgLog10);
END_VAR
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbSpectrum : FB_PMA_Spectrum_3Ph := (nOwnID := 2, stInitPars := cSpectrumInitPars);
    aSpectrumVoltage : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumCurrent : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithm
fbSpectrum.Call(
    ADR(aSpectrumVoltage[0]),
    ADR(aSpectrumVoltage[1]),
    ADR(aSpectrumVoltage[2]),
    ADR(aSpectrumCurrent[0]),
    ADR(aSpectrumCurrent[1]),
    ADR(aSpectrumCurrent[2]),
    SIZEOF(aSpectrumVoltage[0]));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.3.3.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pMagnitudeSpectrum_UL1  : POINTER TO LREAL;
    pMagnitudeSpectrum_UL2  : POINTER TO LREAL;
    pMagnitudeSpectrum_UL3  : POINTER TO LREAL;
    pMagnitudeSpectrum_IL1  : POINTER TO LREAL;
    pMagnitudeSpectrum_IL2  : POINTER TO LREAL;
    pMagnitudeSpectrum_IL3  : POINTER TO LREAL;
    nMagnitudeSpectrumSize  : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pMagnitudeSpectrum_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
pMagnitudeSpectrum_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.3.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Spectrum_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Spectrum_InitPars [▶ 138]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.3.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werde (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.3.4 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4 FB_PMA_Spectrum_Quantiles_3Ph

Der Funktionsbaustein FB_PMA_Spectrum_Quantiles_3Ph berechnet die Betragsspektren der Strom- und Spannungswerte wie der Funktionsbaustein FB_PMA_Spectrum_3Ph [▶ 117]. Zusätzlich können p-Quantile der Verteilung des Spektrums berechnet werden. Die Quantile sowie deren Anzahl lassen sich individuell konfigurieren.

Der Eingangspuffer wird über den Funktionsbaustein FB_PMA_Source_3Ph [▶ 82] bereitgestellt. Die Größe des Eingangspuffers entspricht der halben Fensterlänge.

Beispielhaft sind mögliche FFT- und Fensterlängen in der nachfolgenden Tabelle dargestellt:

FFT-Länge		Fensterlänge	Pufferlänge
512	2^9	400	200
1024	2^{10}	800	400
2048	2^{11}	1600	800
4096	2^{12}	3200	1600
8192	2^{13}	6400	3200
16384	2^{14}	12800	6400

Gedächtniseigenschaften

Der Baustein berücksichtigt alle Eingangswerte seit der Instanziierung. Wenn seit dem Start die Reset [▶ 128]-Methode aufgerufen wurde, werden alle Eingangswerte seit deren letzten Aufruf berücksichtigt.

Syntax

Definition:

```
FUNCTION_BLOCK FB_PMA_Spectrum_Quantiles_3Ph
VAR_INPUT
    nOwnID              : UDINT;
    tTransferTimeout    : LTIME := LTIME#500US;
    stInitPars          : ST_PMA_Spectrum_Quantiles_InitPars;
END_VAR
VAR_OUTPUT
    bError              : BOOL;
    ipResultMessage     : I_TcMessage;
    bNewResult          : BOOL;
    nCntResults         : ULINT;
END_VAR
```

Eingänge

Die Eingangsparameter dieses Bausteins repräsentieren Initialisierungsparameter und müssen bereits bei der Deklaration der Funktionsbausteininstanz zugewiesen werden (alternativ: Init [▶ 125]-Methode). Sie dürfen nur einmal zugewiesen werden. Eine Änderung zur Laufzeit ist nicht möglich.

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.

Name	Typ	Beschreibung
tTransferTimeout	LTIME	Einstellung des synchronen Timeout für interne MultiArray-Weiterleitungen. Siehe Parallelverarbeitung im Transfer Tray.
stInitPars	ST_PMA_Spectrum_Quantiles_InitPars [▶ 139]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	TRUE, falls ein Fehler auftritt.
ipResultMessage	I_TcMessage	Das Interface bietet detaillierte Informationen über den Rückgabewert.
bNewResult	BOOL	TRUE, sobald neue Ergebnisse berechnet wurden.
nCntResults	ULINT	Zählwert wird bei neuen Ausgangsdaten inkrementiert.

Methoden

Name	Beschreibung
Call [▶ 124]	Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.
CallEx [▶ 126]	Um die CPU-Auslastung zu minimieren, kann es erforderlich sein auf die CallEx-Methode zurückzugreifen. Es werden, im Gegensatz zur Call-Methode, nicht nach jeder Spektrum-Berechnung die Quantile berechnet, sondern erst nach einer konfigurierbaren Anzahl von Berechnungen.
Init [▶ 125]	Alternative zur Bausteininitialisierung.
PassInputs [▶ 126]	Die Methode kann alternativ zur Call-Methode in jedem Zyklus aufgerufen werden, falls keine Berechnung erfolgen soll. Der ankommende Eingangspuffer wird dann entsprechend weitergeleitet.
Reconfigure [▶ 128]	Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.
Reset [▶ 128]	Die Methode löscht alle bereits hinzugefügten Datensätze. Alternativ kann das automatische Zurücksetzen an der CallEx-Methode verwendet werden.

Beispiel

```
VAR CONSTANT
    cOversamples : UDINT := 10;
    cFFT_Length : UDINT := 4096;
    cWindowLength : UDINT := 3200;
    cSourceInitPars: ST_PMA_Source_InitPars := (
        nBufferLength := cWindowLength/2);
    cSpectrumQuantilesInitPars : ST_PMA_Spectrum_Quantiles_InitPars := (
        nFFT_Length := cFFT_Length,
        nWindowLength := cWindowLength,
        fSampleRate := cOversamples * 1000,
        eScalingType := E_PMA_ScalingType.PeakAmplitude,
        eWindowType := E_PMA_WindowType.HannWindow,
        bTransformToDecibel := FALSE,
        fDecibelThreshold := GVL_PMA.cMinArgLog10,
        fMinBinnedVoltage := 0.0,
        fMaxBinnedVoltage := 300,
        fMinBinnedCurrent := 0.0,
        fMaxBinnedCurrent := 2,
        nBins := 10,
        nNumQuantiles := 2,
        aQuantiles := [0.5, 0.9]);
END_VAR
```

```
VAR
    aVoltage AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    aCurrent AT%I* : ARRAY[0..2] OF ARRAY [1..cOversamples] OF LREAL;
    fbSource : FB_PMA_Source_3Ph := (nOwnID := 1, aDestIDs := [2], stInitPars := cSourceInitPars);
    fbSpectrumQuantiles : FB_PMA_Spectrum_Quantiles_3Ph := (nOwnID := 2, stInitPars := cSpectrumQuan
tilesInitPars);
    aSpectrumVoltage : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumCurrent : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1] OF LREAL;
    aSpectrumQuantilesVoltage : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1, 1..2] OF LREAL;
    aSpectrumQuantilesCurrent : ARRAY[0..2] OF ARRAY[1..cFFT_Length/2 + 1, 1..2] OF LREAL;
    bNewResult_Spectrum : BOOL;
    bNewResult_Quantiles : BOOL;
END_VAR
```

```
// Call source
fbSource.Call(
    ADR(aVoltage[0]),
    ADR(aVoltage[1]),
    ADR(aVoltage[2]),
    ADR(aCurrent[0]),
    ADR(aCurrent[1]),
    ADR(aCurrent[2]),
    SIZEOF(aVoltage[0]),
    0);

// Call algorithm
fbSpectrumQuantiles.CallEx(
    5,
    FALSE,
    ADR(aSpectrumVoltage[0]),
    ADR(aSpectrumVoltage[1]),
    ADR(aSpectrumVoltage[2]),
    ADR(aSpectrumCurrent[0]),
    ADR(aSpectrumCurrent[1]),
    ADR(aSpectrumCurrent[2]),
    SIZEOF(aSpectrumVoltage[0]),
    ADR(aSpectrumQuantilesVoltage[0]),
    ADR(aSpectrumQuantilesVoltage[1]),
    ADR(aSpectrumQuantilesVoltage[2]),
    ADR(aSpectrumQuantilesCurrent[0]),
    ADR(aSpectrumQuantilesCurrent[1]),
    ADR(aSpectrumQuantilesCurrent[2]),
    SIZEOF(aSpectrumQuantilesVoltage[0]),
    ADR(bNewResult_Spectrum),
    ADR(bNewResult_Quantiles));
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.3.3.4.1 Call

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Eine Alternative stellt die CallEx [▶ 126]-Methode dar. Sie berechnet die Quantile erst nach einer definierten Zahl von Ergebnissen aus der Spektrum-Berechnung, um die CPU-Auslastung zu minimieren.

Syntax

```
METHOD Call : BOOL
VAR_INPUT
    pMagnitudeSpectrum_UL1 : POINTER TO LREAL;
    pMagnitudeSpectrum_UL2 : POINTER TO LREAL;
    pMagnitudeSpectrum_UL3 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL1 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL2 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL3 : POINTER TO LREAL;
    nMagnitudeSpectrumSize : UDINT
    pSpectrumQuantiles_UL1 : POINTER TO LREAL;
```

```
    pSpectrumQuantiles_UL2 : POINTER TO LREAL;
    pSpectrumQuantiles_UL3 : POINTER TO LREAL;
    pSpectrumQuantiles_IL1 : POINTER TO LREAL;
    pSpectrumQuantiles_IL2 : POINTER TO LREAL;
    pSpectrumQuantiles_IL3 : POINTER TO LREAL;
    nSpectrumQuantilesSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
pMagnitudeSpectrum_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
pMagnitudeSpectrum_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.
pSpectrumQuantiles_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pSpectrumQuantiles_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nSpectrumQuantilesSize	UDINT	Gibt die Größe des Ausgangsarrays für eine Quantilberechnung an.

Rückgabewert

Name	Typ	Beschreibung
Call	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4.2 Init

Üblicherweise ist die Init-Methode nicht notwendig in einer Power-Monitoring-Applikation. Sie bietet eine Alternative zur Bausteininitialisierung, mit der eine Kapselung des Bausteins möglich ist. Es müssen dafür die Methode FB_init oder das Attribut 'call_after_init' verwendet werden (siehe TwinCAT 3 PLC > Referenz Programmierung). Die Init-Methode darf nur während der Initialisierungsphase der SPS aufgerufen werden. Sie kann nicht während der Laufzeit verwendet werden.

Die Eingangsparameter der Bausteininstanz dürfen nicht bei der Deklaration zugewiesen werden, falls die Initialisierung mit der Init-Methode erfolgen soll.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    nOwnID      : UDINT;
    stInitPars  : ST_PMA_Spectrum_Quantiles_InitPars;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nOwnID	UDINT	Identifiziert die Bausteininstanz mit einer eindeutigen ID. Diese muss immer größer als null sein. Eine bewährte Vorgehensweise ist die Definition einer Enumeration für diesen Zweck.
stInitPars	ST_PMA_Spectrum_Quantiles_InitPars [▶ 139]	Bausteinspezifische Struktur mit Initialisierungsparametern. Die Parameter müssen mit der Definition der Ein- und Ausgangspuffer übereinstimmen.

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4.3 PassInputs

Solange eine Instanz des Funktionsbausteins FB_PMA_Source_3Ph [▶ 82] aufgerufen wird und somit Signaldaten zu einem Zielblock übertragen werden, müssen alle weiteren Blöcke der Analysekette zyklisch aufgerufen werden (siehe Parallelverarbeitung im Transfer Tray).

Manchmal ist es sinnvoll, einen Algorithmus für eine bestimmte Zeit nicht auszuführen. Zwar muss der Funktionsbaustein dennoch zyklisch aufgerufen werden, aber es ist ausreichend wenn die ankommenden Eingangsdaten weitergeleitet werden. Dies geschieht mit der PassInputs-Methode anstelle der Call-Methode. Hierbei wird kein Ergebnis generiert.

Syntax

```
METHOD PassInputs : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PassInputs	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4.4 CallEx

Die Methode wird in jedem Zyklus aufgerufen, um die Berechnungen aus dem Eingangspuffer durchzuführen, wenn neue Daten vorhanden sind.

Der Baustein wartet auf Eingangsdaten, sofern die Methode weder neue Ergebnisse noch einen Fehler ausgibt. Dies ist ein reguläres Verhalten im Ablauf der Analysekette.

Im Gegensatz zur Call [▶ 124]-Methode werden bei der CallEx-Methode eine variable Anzahl von Ergebnissen aus der Spektrum-Berechnung gesammelt und erst anschließend die Quantile berechnet. Dies kann erforderlich sein, um die CPU-Auslastung zu minimieren.

Syntax

```
METHOD CallEx : BOOL
VAR_INPUT
    nAppendData            : UDINT;
    bResetData             : BOOL;
    pMagnitudeSpectrum_UL1 : POINTER TO LREAL;
    pMagnitudeSpectrum_UL2 : POINTER TO LREAL;
    pMagnitudeSpectrum_UL3 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL1 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL2 : POINTER TO LREAL;
    pMagnitudeSpectrum_IL3 : POINTER TO LREAL;
    nMagnitudeSpectrumSize : UDINT
    pSpectrumQuantiles_UL1 : POINTER TO LREAL;
    pSpectrumQuantiles_UL2 : POINTER TO LREAL;
    pSpectrumQuantiles_UL3 : POINTER TO LREAL;
    pSpectrumQuantiles_IL1 : POINTER TO LREAL;
    pSpectrumQuantiles_IL2 : POINTER TO LREAL;
    pSpectrumQuantiles_IL3 : POINTER TO LREAL;
    nSpectrumQuantilesSize : UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nAppendData	UDINT	Anzahl der zu berechnenden Spektren bis eine Berechnung der Quantile erfolgt. Ein Wert von 1 bedeutet dass nach jedem Ergebnis der Spektrum-Berechnung die Quantile berechnet werden.
bResetData	BOOL	Automatisches Zurücksetzen der Datensätze nach jeder Quantil-Berechnung
pMagnitudeSpectrum_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
pMagnitudeSpectrum_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1. Wenn das Spektrum nicht ausgegeben werden soll, kann der Eingang auf 0 gesetzt werden.
nMagnitudeSpectrumSize	UDINT	Gibt die Größe des Ausgangsarrays eines Spektrums an.
pSpectrumQuantiles_UL1 .. UL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
pSpectrumQuantiles_IL1 .. IL3	POINTER TO LREAL	Zeiger auf ein Array vom Typ LREAL mit der Dimension: FFT-Länge/2+1 x Quantile. Wenn diese Werte nicht ausgegeben werden sollen, kann der Eingang auf 0 gesetzt werden.
nSpectrumQuantilesSize	UDINT	Gibt die Größe des Ausgangsarrays für eine Quantilberechnung an.

Rückgabewert

Name	Typ	Beschreibung
CallEx	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4.5 Reset

Die Methode löscht alle bereits hinzugefügten Datensätze. Alternativ kann das automatische Zurücksetzen an der CallEx [▶ 126]-Methode verwendet werden.

Syntax

```
METHOD Reset : BOOL
VAR_INPUT
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.3.3.4.6 Reconfigure

Die Methode wird aufgerufen, um den Funktionsbaustein während der Laufzeit neu zu konfigurieren.

Syntax

```
METHOD Reconfigure : BOOL
VAR_INPUT
    aQuantiles      : ARRAY[0..GVL_PMA.cMaxQuantiles - 1] OF LREAL;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
aQuantiles	ARRAY[0..GVL_PMA.cMaxQuantiles-1] OF LREAL	Gibt die Quantilgrenze an. Sie muss zwischen 0.0 und 1.0 liegen. Beispielsweise entspricht 0.2 dem 20%-Quantil.

Rückgabewert

Name	Typ	Beschreibung
Reconfigure	BOOL	Gibt an, ob die Methode erfolgreich ausgeführt worden ist. Weitere Informationen im Event Interface des Bausteins.

5.4 Datentypen

5.4.1 E_PMA_ScalingType

Mit der Enumeration Enumeration E_PMA_ScalingType werden alle Skalierungsmöglichkeiten für die Spektralberechnungen aufgeführt. Sie wird in den Strukturen der Initialisierungsparameter verwendet. Weitere Details dazu finden Sie im Anhang der Dokumentation „TF3600 Condition Monitoring“.

Syntax

Definition:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_PMA_ScalingType :
(
     NoScaling            := 0,
     DiracScaling         := 1,
     PeakAmplitude        := 2,
     RootPowerSum         := 3,
     RMS                  := 4,
     GainCorrection       := 5,
     PowerSpectralDensity := 6,
     UnitaryScaling       := 7
) UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.2 E_PMA_WindowType

Mit der Enumeration E_PMA_WindowType kann der Fenstertyp für die Spektralberechnungen ausgewählt werden. Sie wird in den Strukturen der Initialisierungsparameter verwendet. Weitere Details dazu finden Sie im Anhang der Dokumentation „TF3600 Condition Monitoring“.

Syntax

Definition:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_PMA_WindowType :
(
    HannWindow          := 16#05300901,
    RectangularWindow   := 16#05300902
) UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3 InitParameter

5.4.3.1 Allgemein

5.4.3.1.1 ST_PMA_Scaling_EL3773_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung des Bausteins FB_PMA_Scaling_EL3773 [▶ 22] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Scaling_EL3773_InitPars :
STRUCT
    nOversamples                : UDINT  := 10;   // Oversampling factor
    fOffsetVoltage              : LREAL  := 0.0;  // Output = (input * gain) + offset
    fGainVoltage                : LREAL  := 1.0;  //  ||
    fOffsetCurrent              : LREAL  := 0.0;  // Output = (input * gain) + offset
```

```
    fGainCurrent                : LREAL  := 1.0;  //  ||
    fFactorCurrentTransformer   : LREAL  := 1.0;  // Factor of current tranformer (input / output)
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
nOversamples	UDINT	Oversampling-Faktor
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.
fFactorCurrentTransformer	LREAL	Faktor des Stromwandlers. Dieser berechnet sich aus Eingangsstrom und Ausgangsstrom.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.1.2 ST_PMA_Scaling_EL3783_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung des Bausteins FB_PMA_Scaling_EL3783 [▶ 26] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Scaling_EL3783_InitPars :
STRUCT
    nOversamples                : UDINT  := 20;   // Oversampling factor
    fOffsetVoltage              : LREAL  := 0.0;  // Output = (input * gain) + offset
    fGainVoltage                : LREAL  := 1.0;  //  ||
    fOffsetCurrent              : LREAL  := 0.0;  // Output = (input * gain) + offset
    fGainCurrent                : LREAL  := 1.0;  //  ||
    fFactorCurrentTransformer   : LREAL  := 1.0;  // Factor of current tranformer (input / output)
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
nOversamples	UDINT	Oversampling-Faktor
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.
fFactorCurrentTransformer	LREAL	Faktor des Stromwandlers. Dieser berechnet sich aus Eingangsstrom und Ausgangsstrom.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.1.3 ST_PMA_Scaling_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung des Bausteins FB_PMA_Scaling [▶ 17] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Scaling_InitPars :
STRUCT
    nOversamples              : UDINT := 10;      // Oversampling factor
    nResolutionVoltage        : UDINT := (1..32); // in bit
    fMaxVoltage               : LREAL;            // Max input amplitude
    fOffsetVoltage            : LREAL := 0;       // Output = (input * gain) + offset
    fGainVoltage              : LREAL := 1;       //  ||
    nResolutionCurrent        : UDINT := (1..32); // in bit
    fMaxCurrent               : LREAL;            // Max input amplitude
    fOffsetCurrent            : LREAL := 0;       // Output = (input * gain) + offset
    fGainCurrent              : LREAL := 1;       //  ||
    fFactorCurrentTransformer : LREAL := 1;       // Factor of current transformer (input / output)
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nOversamples	UDINT	Oversampling-Faktor
nResolutionVoltage	UDINT	Auflösung der Spannung in Bit. Wertebereich: 0-32
fMaxVoltage	LREAL	Gibt die maximale Amplitude der Eingangsspannung an.
fOffsetVoltage	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung der Spannung an.
fGainVoltage	UDINT	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung der Spannung an.
nResolutionCurrent	LREAL	Auflösung des Stroms in Bit. Wertebereich: 0-32
fMaxCurrent	LREAL	Gibt die maximale Amplitude des Eingangsstroms an.
fOffsetCurrent	LREAL	Gibt einen benutzerdefinierten Offset für die Skalierung des Stroms an.
fGainCurrent	LREAL	Gibt einen benutzerdefinierten Verstärkungsfaktor für die Skalierung des Stroms an.
fFactorCurrentTransformer	LREAL	Faktor des Stromwandlers. Dieser berechnet sich aus Eingangsstrom und Ausgangsstrom.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.1.4 ST_PMA_Source_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Source_1Ph [▶ 37] und FB_PMA_Source_3Ph [▶ 82] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Source_InitPars :
STRUCT
    nBufferLength  : UDINT := 200;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nBufferlength	UDINT	Länge des Ausgangspuffers

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.1.5 ST_PMA_TaskTransfer_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_TaskTransfer_Send [▶ 32] und FB_PMA_TaskTransfer_Receive [▶ 34] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_TaskTransfer_InitPars :
STRUCT
    nInputSize  : UDINT;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nInputSize	UDINT	Anzahl an Elementen, die über die Funktionsbausteine zum Task-Transfer ausgetauscht werden

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.2 Basierend auf der Signalperiode

5.4.3.2.1 ST_PMA_BasicValues_Period_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_BasicValues_Period_1Ph [▶ 45] und FB_PMA_BasicValues_Period_3Ph [▶ 90] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_BasicValues_Period_InitPars :
STRUCT
    nBufferLength    : UDINT := 200;   // Length of input buffer
    fSampleRate      : LREAL := 1000;  // in Hz
    fMinInputCurrent : LREAL := 0.0;   // Minimal input of current (RMS) to calculate outputs
    nPeriods         : UDINT := 1;     // Amount of signal periods to calculate outputs
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nBufferLength	UDINT	Länge des Eingangspuffers
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.
nPeriods	UDINT	Anzahl an Signalperioden zur Berechnung des RMS

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.2.2 ST_PMA_Frequency_Period_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Frequency_Period_1Ph [▶ 40] und FB_PMA_Frequency_Period_3Ph [▶ 85] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Frequency_Period_InitPars :
STRUCT
    nBufferLength      : UDINT := 200;   // Length of input buffer
    fSampleRate        : LREAL := 1_000; // in Hz
    fMinFreq           : LREAL := 45;    // Min measured Freq
    fMaxFreq           : LREAL := 65;    // Max measured Freq
    nPeriods           : UDINT := 2;     // Number of periods to be considered
    nFilterOrder       : UINT  := 3;     // Filter order of butterworth lowpass filter
    fCutoff            : LREAL := 70.0;  // Cutoff frequency of filter
    eInputSelect       : E_PMA_InputSelect := E_PMA_InputSelect.Voltage; // Select input: Voltage |
 Current
    fMinInput          : LREAL := 200.0; // Minimal input (RMS) over one period to calculate outputs
    nRocofAvgWindow    : UDINT := 25;    // Window length of elements to calculate moving average of
 ROCOF
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nBufferLength	UDINT	Länge des Eingangspuffers
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
fMinFreq	LREAL	Minimal zu erwartende Messfrequenz
fMaxFreq	LREAL	Maximal zu erwartende Messfrequenz
nPeriods	UDINT	Anzahl an Perioden, die Einfluss auf die Berechnung haben. (Periodenlänge = Samplerate/Frequenz)

Name	Typ	Parameter
nFilterOrder	UINT	Gibt die Ordnung des Tiefpassfilters an. Bei der Einstellung muss die Stabilität des Filters berücksichtigt werden. Nur Werte bis zur zehnten Ordnung sind zulässig.
fCutoff	LREAL	Gibt die Grenzfrequenz des Tiefpassfilters an.
eInputSelect	E_PMA_InputSelect [▶ 146]	*Voltage*: Die Spannung wird als Basis für die Frequenzberechnung genutzt *Current*: Der Strom wird als Basis für die Frequenzberechnung genutzt
fMinInput	LREAL	Mindesteingangsgröße (RMS) über eine Periode. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.
nRocofAvgWindow	UDINT	Fenstergröße des gleitenden Mittelwertes für die Berechnung des ROCOF Resultierende Zeitspanne: nPeriods * nRocofAvgWindow / avgeragedFreq)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.2.3 ST_PMA_PowerValues_Period_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_PowerValues_Period_1Ph [▶ 50] und FB_PMA_PowerValues_Period_3Ph [▶ 96] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_PowerValues_Period_InitPars :
STRUCT
    nBufferLength              : UDINT := 200;    // Length of input buffer
    fSampleRate                : LREAL := 1000;   // in Hz
    fMinInputCurrent           : LREAL := 0.0;    // Minimal input of current (RMS) to calculate outp
uts
    nPeriods                   : UDINT := 1;      // Amount of signal periods to calculate outputs
    fNominalVoltage            : LREAL := 230.0; // Nominal voltage, required for PQF calculation
    fNominalFreq               : LREAL := 50.0;   // Nominal frequency, required for PQF calculation
    ePqfMode                   : E_PMA_PqfMode := E_PMA_PqfMode.Default; // Mode of PQF calculation
    uTimeLagCurrentTransformer : U_PMA_Timelag := (fTimeLag := 0);       // Timelag caused by induct
ivity of current transformer ( in s
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nBufferLength	UDINT	Länge des Eingangspuffers
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.
nPeriods	UDINT	Anzahl an Signalperioden zur Berechnung der Ausgangswerte
fNominalVoltage	LREAL	Nominaler Spannungswert. Wird für die Berechnung des Power Quality Factors benötigt.

Name	Typ	Parameter
fNominalFreq	LREAL	Nominale Frequenz. Wird für die Berechnung des Power Quality Factors benötigt.
ePqfMode	E_PMA_PqfMode [▶ 147]	Modus für die Berechnung des Power Quality Factors. *Default*: Berechnung des Power Quality Factors mit der Frequenz, dem Effektivwert der Spannung sowie dem THD der Spannung. *DefaultAndUnbalance*: Wie Default, jedoch das in einem dreiphasigen System zusätzlich noch die Spannungsunsymmetrie mit in die Berechnung einfließt.
uTimeLagCurrentTransformer	U_PMA_Timelag	Hier kann die mögliche Verzögerung durch die Induktivität des Stromwandlers in Sekunden angegeben werden. *U_PMA_Timelag.fTimeLag*: Identisch für alle Phasen *U_PMA_Timelag.aTimeLag*: Individuell pro Phase

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.2.4 ST_PMA_Harmonics_Period_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Harmonics_Period_1Ph [▶ 56] und FB_PMA_Harmonics_Period_3Ph [▶ 102] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Harmonics_Period_InitPars :
STRUCT
    nBufferLength       : UDINT := 200;   // Length of input buffer
    fSampleRate         : LREAL := 1000;  // in Hz
    nNumHarmonics       : UDINT := 20;    // Number of harmonics
    nPeriods            : UDINT := 10;    // Amount of signal periods to calculate outputs
    bTransformToPercent : BOOL := TRUE;   // transform results to percent (1st harmonic=100%)
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nBufferLength	UDINT	Länge des Eingangspuffers
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
nNumHarmonics	UDINT	Anzahl zu berechnender Harmonischen
nPeriods	UDINT	Anzahl an Signalperioden zur Berechnung der Ausgangswerte.
bTransformToPercent	BOOL	Boolescher Wert der angibt, ob das Ergebnis in Prozent ausgegeben werden soll. Dabei entspricht die erste Harmonische 100%.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.3 Basierend auf dem Frequenzbereich

5.4.3.3.1 ST_PMA_Harmonics_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Harmonics_1Ph [▶ 60] und FB_PMA_Harmonics_3Ph [▶ 106] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_HarmonicAnalysis_InitPars :
STRUCT
    fSampleRate          : LREAL := 1000;    // Sample rate
    fBaseFreq            : LREAL := 50.0;    // Multiple of base frequence is used for band limit de
finition
    nFFT_Length          : UDINT := 512;     // Length of FFT
    nWindowLength        : UDINT := 400;     // Length of FFT window
    nNumBands            : UDINT := 10;      // Number of bands
    fBandwidth           : LREAL := 4.0;     // Whole bandwidth for each frequency band, min 1.0
    eWindowType          : E_PMA_WindowType := E_PMA_WindowType.HannWindow; // Window  function used
    bTransformToDecibel  : BOOL  := TRUE;    // Transform result to decibel
    fDecibelThreshold    : LREAL := 1.3E-308;// Logarithm threshold for decibel transformation
    bTransformToPercent  : BOOL := FALSE;    // Transform result to percent (1st harmonic=100%)
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
fBaseFreq	LREAL	Frequenz der ersten Harmonischen.
nFFT_Length	UDINT	Länge der FFT. Sie muss größer als Eins und eine ganzzahlige Potenz von Zwei sein.
nWindowLength	UDINT	Länge des Analysefensters in Samples. Die Länge muss größer als eins und eine gerade Zahl sein.
nNumBands	UDINT	Gibt die Anzahl an Bändern an, für die der RMS berechnet wird.
fBandwidth	LREAL	Gesamte Bandweite der einzelnen RMS-Bänder
eWindowType	E_PMA_WindowType [▶ 129]	Definiert die verwendete Fensterfunktion. Ein guter Standardwert ist der Fenstertyp „HannWindow“. Die Fensterung kann durch die Verwendung des Fenstertyps „RectangularWindow“ abgeschaltet werden. Weitere Erläuterungen und die Liste von möglichen Fensterfunktionen finden sich im einleitenden Teil im Abschnitt „Fensterfunktionen“.
bTransformToDecibel	BOOL	Boolescher Wert, der angibt, ob das Ergebnis der FFT in die Dezibel-Skala transformiert werden soll, entsprechend der Transformation x → 20 * log10(x).

Name	Typ	Parameter
fDecibelThreshold	LREAL	Sehr kleiner Fließkommawert größer als Null. Werte, die kleiner als diese Zahl sind, werden vor einer Transformation in die Dezibel-Skala durch diesen Wert ersetzt, da der Logarithmus von Null mathematisch nicht definiert ist. Der kleinste mögliche Wert ist 3.75e-324.
bTransformToPercent	BOOL	Boolescher Wert, der angibt, ob das Ergebnis in Prozent ausgegeben werden soll. Dabei entspricht die erste Harmonische 100%.

Fensterlänge

Der Wert von nWindowLength muss kleiner oder gleich dem Wert von nFFT_Length sein. Die Länge der FFT kann sich an der benötigten Frequenzauflösung orientieren. Typischerweise wird ein Wert von ca. 3/4 der FFT-Länge als Fensterlänge verwendet.

Wenn nFFT_Length größer ist als nWindowLength, wird die Frequenzauflösung der FFT (und damit auch die Länge des Vektors der Rückgabewerte) vergrößert. Die Differenz der Länge wird vor der Fourier-Transformation mit Nullen aufgefüllt. Dies kann sinnvoll sein, um eine höhere Frequenzauflösung zu erreichen oder um, z. B. bei der Berechnung mit Rücktransformation in den Zeitbereich, zirkuläres Aliasing zu vermeiden. Das Ergebnis enthält trotz der höheren Frequenzauflösung allerdings nicht mehr Informationen.

5.4.3.3.2 ST_PMA_PowerValues_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_PowerValues_1Ph [▶ 66] und FB_PMA_PowerValues_3Ph [▶ 112] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_PowerValues_InitPars :
STRUCT
    fSampleRate                  : LREAL := 1000;     // in Hz
    fBaseFreq                    : LREAL := 50.0;     // in HZ
    nFFT_Length                  : UDINT := 512;      // Length of FFT
    nWindowLength                : UDINT := 400;      // Length of FFT window
    nNumBands                    : UDINT := 10,       // Number of bands
    fBandwidth                   : LREAL := 4.0       // Whole bandwidth of each frequency band
    eWindowType                  : E_PMA_WindowType := E_PMA_WindowType.HannWindow; // Window funct
ion used
    fMinInputCurrent             : LREAL := 0.0;      // Minimal input of current (RMS) to calculate
 outputs
    uTimeLagCurrentTransformer   : U_PMA_Timelag    := (fTimeLag := 0); // Timelag caused by induct
ivity of current transformer ( in s )
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
fFreq	LREAL	Frequenz der ersten Harmonischen (Grundschwingung)
nFFT_Length	UDINT	Länge der FFT. Sie muss größer als Eins und eine ganzzahlige Potenz von Zwei sein.
nWindowLength	UDINT	Länge des Analysefensters in Samples. Die Länge muss größer als Eins und eine gerade Zahl sein. Sie darf nicht größer als nFFT_Length sein.

Name	Typ	Parameter
nNumBands	UDINT	Gibt die Anzahl der Bänder an, für die der RMS berechnet wird.
fBandwidth	LREAL	Gesamte Bandweite eines einzelnen RMS-Bandes
eWindowType	E_PMA_WindowType [▶ 129]	Definiert die verwendete Fensterfunktion. Ein guter Standardwert ist der Fenstertyp „HannWindow“. Die Fensterung kann durch die Verwendung des Fenstertyps „RectangularWindow“ abgeschaltet werden. Weitere Erläuterungen und die Liste von möglichen Fensterfunktionen finden sich im einleitenden Teil im Abschnitt Fensterfunktionen.
fMinInputCurrent	LREAL	Mindesteingangsgröße (RMS) des Stroms. Diese verhindert die Berechnung bei zu kleinen Eingangswerten.
uTimeLagCurrentTransformer	U_PMA_Timelag	Hier kann die mögliche Verzögerung durch die Induktivität des Stromwandlers in Sekunden angegeben werden. *U_PMA_Timelag.fTimeLag*: Identisch für alle Phasen *U_PMA_Timelag.aTimeLag*: Individuell pro Phase

Fensterlänge

Der Wert von nWindowLength muss kleiner oder gleich dem Wert von nFFT_Length sein. Die Länge der FFT kann sich an der benötigten Frequenzauflösung orientieren. Typischerweise wird ein Wert von ca. 3/4 der FFT-Länge als Fensterlänge verwendet.

Wenn nFFT_Length größer ist als nWindowLength, wird die Frequenzauflösung der FFT (und damit auch die Länge des Vektors der Rückgabewerte) vergrößert. Die Differenz der Länge wird vor der Fourier-Transformation mit Nullen aufgefüllt. Dies kann sinnvoll sein, um eine höhere Frequenzauflösung zu erreichen oder um, z. B. bei der Berechnung mit Rücktransformation in den Zeitbereich, zirkuläres Aliasing zu vermeiden. Das Ergebnis enthält trotz der höheren Frequenzauflösung allerdings nicht mehr Informationen.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.3.3 ST_PMA_Spectrum_InitPars

Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Spectrum_1Ph [▶ 71] und FB_PMA_Spectrum_3Ph [▶ 117] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Spectrum_InitPars :
STRUCT
    nFFT_Length         : UDINT := 512;      // Length of FFT
    nWindowLength       : UDINT := 400;      // Length of FFT window
    fSampleRate         : LREAL := 1000;     // in Hz
    eScalingType        : E_PMA_ScalingType := E_PMA_ScalingType.NoScaling; // Scaling type used
    eWindowType         : E_PMA_WindowType  := E_PMA_WindowType.HannWindow; // Window function used
    bTransformToDecibel : BOOL := TRUE;      // Transform result to decibel
    fDecibelThreshold   : LREAL := 1.3E-308; // Logarithm threshold for decibel transformation
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nFFT_Length	UDINT	Länge der FFT. Sie muss größer als Eins und eine ganzzahlige Potenz von Zwei sein.
nWindowLength	UDINT	Länge des Analysefensters in Samples. Die Länge muss größer als eins und eine gerade Zahl sein. Sie darf nicht größer als nFFT_Length sein.
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
eScalingType	E_PMA_ScalingType [▶ 128]	Ermöglicht eine Auswahl der verwendeten Skalierung, falls eine absolute Skalierung benötigt wird.
eWindowType	E_PMA_WindowType [▶ 129]	Definiert die verwendete Fensterfunktion. Ein guter Standardwert ist der Fenstertyp „HannWindow“. Die Fensterung kann durch die Verwendung des Fenstertyps „RectangularWindow“ abgeschaltet werden. Weitere Erläuterungen und die Liste von möglichen Fensterfunktionen finden sich im einleitenden Teil im Abschnitt „Fensterfunktionen“.
bTransformToDecibel	BOOL	Boolescher Wert, der angibt, ob das Ergebnis der FFT in die Dezibel-Skala transformiert werden soll, entsprechend der Transformation x → 20 * log10(x).
fDecibelThreshold	LREAL	Sehr kleiner Fließkommawert größer als Null. Werte, die kleiner als diese Zahl sind, werden vor einer Transformation in die Dezibel-Skala durch diesen Wert ersetzt, da der Logarithmus von Null mathematisch nicht definiert ist. Der kleinste mögliche Wert ist 3.75e-324, dies entspricht der Konstanten `cCM_MinArgLog10`.

Fensterlänge

Der Wert von nWindowLength muss kleiner oder gleich dem Wert von nFFT_Length sein. Die Länge der FFT kann sich an der benötigten Frequenzauflösung orientieren. Typischerweise wird ein Wert von ca. 3/4 der FFT-Länge als Fensterlänge verwendet.

Wenn nFFT_Length größer ist als nWindowLength, wird die Frequenzauflösung der FFT (und damit auch die Länge des Vektors der Rückgabewerte) vergrößert. Die Differenz der Länge wird vor der Fourier-Transformation mit Nullen aufgefüllt. Dies kann sinnvoll sein, um eine höhere Frequenzauflösung zu erreichen oder um, z. B. bei der Berechnung mit Rücktransformation in den Zeitbereich, zirkuläres Aliasing zu vermeiden. Das Ergebnis enthält trotz der höheren Frequenzauflösung allerdings nicht mehr Informationen.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.3.3.4 ST_PMA_Spectrum_Quantiles_InitPars

Lefeld Bausteinspezifische Struktur mit Initialisierungsparametern, die bei der Initialisierung der Bausteine FB_PMA_Spectrum_Quantiles_1Ph [▶ 75] und FB_PMA_Spectrum_Quantiles_3Ph [▶ 122] ausgewertet wird.

Syntax

Definition:

```
TYPE ST_PMA_Spectrum_Quantiles_InitPars :
STRUCT
    nFFT_Length         : UDINT := 512;      // Length of FFT
    nWindowLength       : UDINT := 400;      // Length of FFT window
    fSampleRate         : LREAL := 1000;     // in Hz
    eScalingType        : E_PMA_ScalingType := E_PMA_ScalingType.NoScaling; // Scaling type used
    eWindowType         : E_PMA_WindowType  := E_PMA_WindowType.HannWindow; // Window function used
    bTransformToDecibel : BOOL := TRUE;      // Transform result to decibel
    fDecibelThreshold   : LREAL := 1.3E-308; // Log threshold for decibel transformation
    fMinBinnedVoltage    : LREAL;             // Minimum binned voltage
    fMaxBinnedVoltage    : LREAL;             // Maximum binned voltage
    fMinBinnedCurrent    : LREAL;             // Minimum binned current
    fMaxBinnedCurrent    : LREAL;             // Maximum binned current
    nBins               : UDINT := 1;        // Number of bins in interval
    nNumQuantiles       : UDINT := 1;        // Maximum number of quantile values
    aQuantiles          : ARRAY[0..GVL_PMA.cMaxQuantiles-1] OF LREAL; // 0.0 < aQuantiles[x] <1.0
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Parameter
nFFT_Length	UDINT	Länge der FFT. Sie muss größer als Eins und eine ganzzahlige Potenz von Zwei sein.
nWindowLength	UDINT	Länge des Analysefensters in Samples. Die Länge muss größer als eins und eine gerade Zahl sein. Sie darf nicht größer als nFFT_Length sein.
fSampleRate	LREAL	Gibt die Abtastrate (Samples pro Sekunde) des Eingangssignals an.
eScalingType	E_PMA_ScalingType [▶ 128]	Ermöglicht eine Auswahl der verwendeten Skalierung, falls eine absolute Skalierung benötigt wird.
eWindowType	E_PMA_WindowType [▶ 129]	Definiert die verwendete Fensterfunktion. Ein guter Standardwert ist der Fenstertyp „HannWindow“. Die Fensterung kann durch die Verwendung des Fenstertyps „RectangularWindow“ abgeschaltet werden. Weitere Erläuterungen und die Liste von möglichen Fensterfunktionen finden sich im einleitenden Teil im Abschnitt „Fensterfunktionen“.
bTransformToDecibel	BOOL	Boolescher Wert, der angibt, ob das Ergebnis der FFT in die Dezibel-Skala transformiert werden soll, entsprechend der Transformation x → 20 * log10(x).
fDecibelThreshold	LREAL	Sehr kleiner Fließkommawert größer als Null. Werte, die kleiner als diese Zahl sind, werden vor einer Transformation in die Dezibel-Skala durch diesen Wert ersetzt, da der Logarithmus von Null mathematisch nicht definiert ist. Der kleinste mögliche Wert ist 3.75e-324.
fMinBinnedVoltage	LREAL	Unterer Grenzwert, für den die Ausgangsdaten der Spektrumsberechnung der Spannung in den regulären Histogramm-Bins gezählt werden.
fMaxBinnedVoltage	LREAL	Oberer Grenzwert, für den die Ausgangsdaten der Spektrumsberechnung der Spannung in den regulären Histogramm-Bins gezählt werden. fMaxBinnedVoltage muss größer sein als fMinBinnedVoltage.

Name	Typ	Parameter
fMinBinnedCurrent	LREAL	Unterer Grenzwert, für den die Ausgangsdaten der Spektrumsberechnung des Stroms in den regulären Histogramm-Bins gezählt werden.
fMaxBinnedCurrent	LREAL	Oberer Grenzwert, für den die Ausgangsdaten der Spektrumsberechnung des Stroms in den regulären Histogramm-Bins gezählt werden. fMaxBinnedCurrent muss größer sein als fMinBinnedCurrent.
nBins	UDINT	Anzahl der Bins eines Histogramms. Sie muss mindestens eins sein. In vielen Fällen sind Werte zwischen zehn und zwanzig eine sinnvolle Wahl. Die beiden speziellen Bins für Werte, die unterhalb von fMinBinned bzw. oberhalb von fMaxBinned liegen sind in diesem Wert nicht mit enthalten.
nNumQuantiles	UDINT	Zahl der zu berechnenden Quantile für jeden Kanal. Dies muss eine ganze Zahl größer als Null sein.
aQuantiles	ARRAY[0..GVL_PMA.cMaxQuantiles-1] OF LREAL	Gibt die Quantilgrenze an. Sie muss zwischen 0.0 und 1.0 liegen. Beispielsweise entspricht 0.2 dem 20%-Quantil.

Fensterlänge

Der Wert von nWindowLength muss kleiner oder gleich dem Wert von nFFT_Length sein. Die Länge der FFT kann sich an der benötigten Frequenzauflösung orientieren. Typischerweise wird ein Wert von ca. 3/4 der FFT-Länge als Fensterlänge verwendet.

Wenn nFFT_Length größer ist als nWindowLength, wird die Frequenzauflösung der FFT (und damit auch die Länge des Vektors der Rückgabewerte) vergrößert. Die Differenz der Länge wird vor der Fourier-Transformation mit Nullen aufgefüllt. Dies kann sinnvoll sein, um eine höhere Frequenzauflösung zu erreichen oder um, z. B. bei der Berechnung mit Rücktransformation in den Zeitbereich, zirkuläres Aliasing zu vermeiden. Das Ergebnis enthält trotz der höheren Frequenzauflösung allerdings nicht mehr Informationen.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.4 Einphasig

5.4.4.1 ST_PMA_BasicValues_Period_1Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst und welche über das Property stResults abgefragt werden kann. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_BasicValues_Period_1Ph [▶ 45].

Syntax

Definition:

```
TYPE ST_PMA_BasicValues_Period_1Ph :
STRUCT
    fMeanValue_U      : LREAL; // [V] | Mean value over n periods
    fRMS_U            : LREAL; // [V] | Root mean square over n periods
    fRMS_U_Min        : LREAL; // [V] | Min value of fRMS_U
    fRMS_U_Max        : LREAL; // [V] | Max value of fRMS_U
    fPeakValue_U      : LREAL; // [V] | Peak value of U over n periods
```

```
    fPeakHold_U       : LREAL; // [V] | All time peak of U
    fRectifiedValue_U : LREAL; // [V] | Rectified value over n periods
    fCrestFactor_U    : LREAL; // []  | Peak_value / RMS
    fFormFactor_U     : LREAL; // []  | RMS / rectified_value
    fMeanValue_I      : LREAL; // [A] | Mean value over n periods
    fRMS_I            : LREAL; // [A] | Root mean square over n periods
    fRMS_I_Min        : LREAL; // [A] | Min value of fRMS_I
    fRMS_I_Max        : LREAL; // [A] | Max value of fRMS_I
    fPeakValue_I      : LREAL; // [A] | Peak value of I over n periods
    fPeakHold_I       : LREAL; // [A] | All time peak of I
    fRectifiedValue_I : LREAL; // [A] | Rectified value over n periods
    fCrestFactor_I    : LREAL; // []  | Peak_value / RMS
    fFormFactor_I     : LREAL; // []  | RMS / rectified_value
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.4.2 ST_PMA_Frequency_Period_1Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_Frequency_Period_1Ph [▶ 40].

Syntax

Definition:

```
TYPE ST_PMA_Frequency_Period_1Ph :
STRUCT
    fFreq     : LREAL; // [Hz]    | f     | Frequency calculated by zero crossings
    fFreq_Min : LREAL; // [Hz]    | f_min | Min value of fFreq
    fFreq_Max : LREAL; // [Hz]    | f_max | Max value of fFreq
    fRocof    : LREAL; // [Hz/s]  |       | Rate of change of frequency (ROCOF)
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.4.3 ST_PMA_PowerValues_1Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in den Beschreibungen der Funktionsbausteine FB_PMA_PowerValues_Period_1Ph [▶ 50] oder FB_PMA_PowerValues_1Ph [▶ 66].

Syntax

Definition:

```
Type ST_PMA_PowerValues_1Ph
STRUCT
    fApparentPower       : LREAL;         // [VA]  | S       | apparent power
    fApparentPower_1     : LREAL;         // [VA]  | S_1     | fundamental apparent power
    fApparentPower_1_Min : LREAL;         // [VA]  | S_1_min | Min value of S_1
    fApparentPower_1_Max : LREAL;         // [VA]  | S_1_max | Max value of S_1
    fActivePower         : LREAL;         // [W]   | P       | active power
    fActivePower_Min     : LREAL;         // [W]   | P_min   | Min value of P
    fActivePower_Max     : LREAL;         // [W]   | P_max   | Max value of P
    fReactivePower_d     : LREAL;         // [var] | Q_d     | distortion reactive power
    fReactivePower_1     : LREAL;         // [var] | Q_1     | fundamental reactive power
```

```
    fReactivePower_1_Min : LREAL;         // [var] | Q_1_min | Min value of Q_1
    fReactivePower_1_Max : LREAL;         // [var] | Q1_max  | Max value of Q_1
    fTotalReactivePower  : LREAL;         // [var] | Q_tot   | total reactive power
    fPhi                 : LREAL;         // [°]   | phi     | phase difference
    fCosPhi              : LREAL;         // []    | COS(phi)|(P / S_1)
    fPowerFactor         : LREAL;         // []    | PF      | power factor (P / S)
    stEnergy_Pos         : ST_PMA_Energy; // [kWh] | W_pos   | Energy in positive direction
    stEnergy_Neg         : ST_PMA_Energy; // [kWh] | W_neg   | Energy in negative direction
    stEnergy_Res         : ST_PMA_Energy; // [kWh] | W       | Resulting energy
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.4.4 ST_PMA_THD_1Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_Harmonics_1Ph [▶ 60].

Syntax

Definition:

```
TYPE ST_PMA_THD_1Ph :
STRUCT
    fTHD_U     : LREAL; // [] | THD_U     | Total harmonic distortion of voltage | in percent
    fTHD_U_Min : LREAL; // [] | THD_U_min | Min value of THD_U
    fTHD_U_Max : LREAL; // [] | THD_U_max | Max value of THD_U
    fTHD_I     : LREAL; // [] | THD_I     | Total harmonic distortion of current | in percent
    fTHD_I_Min : LREAL; // [] | THD_I_min | Min value of THD_I
    fTHD_I_Max : LREAL; // [] | THD_I_max | Max value of THD_I
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.5 Dreiphasig

5.4.5.1 E_PMA_RotationalDirection_3Ph

Syntax

Definition:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_PMA_RotationalDirection_3Ph :
(
    No_Direction := 0; // no rotational direction detected
    Right        := 1;
    Left         := 2;
) UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.5.2 ST_PMA_BasicValues_Period_3Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_BasicValues_Period_3Ph [▶ 90].

Syntax

Definition:

```
TYPE ST_PMA_BasicValues_Period_3Ph :
STRUCT
    aMeanValue_U      : ARRAY[0..2] OF LREAL; // [V] | Mean value over n periods
    aRMS_U            : ARRAY[0..2] OF LREAL; // [V] | Root mean square over n periods
    aRMS_U_Min        : ARRAY[0..2] OF LREAL; // [V] | Min value of aRMS_U
    aRMS_U_Max        : ARRAY[0..2] OF LREAL; // [V] | Max value of aRMS_U
    aRMS_UPP          : ARRAY[0..2] OF LREAL; // [V] | 0: L1 - L2 | 1: L2 - L3 | 2: L3 - L1
    aPeakValue_U      : ARRAY[0..2] OF LREAL; // [V] | Peak value of U over n periods
    aPeakHold_U       : ARRAY[0..2] OF LREAL; // [V] | All time peak of U
    aRectifiedValue_U : ARRAY[0..2] OF LREAL; // [V] | Rectified value over n periods
    aCrestFactor_U    : ARRAY[0..2] OF LREAL; // []  | Peak_value / RMS
    aFormFactor_U     : ARRAY[0..2] OF LREAL; // []  | RMS / rectified_value
    aMeanValue_I      : ARRAY[0..2] OF LREAL; // [A] | Mean value over n periods [A]
    aRMS_I            : ARRAY[0..2] OF LREAL; // [A] | Root mean square over n periods [A]
    aRMS_I_Min        : ARRAY[0..2] OF LREAL; // [A] | Min value of aRMS_I [A]
    aRMS_I_Max        : ARRAY[0..2] OF LREAL; // [A] | Max value of aRMS_I [A]
    aPeakValue_I      : ARRAY[0..2] OF LREAL; // [A] | Peak value of I over n periods [A]
    aPeakHold_I       : ARRAY[0..2] OF LREAL; // [A] | All time peak of I [A]
    aRectifiedValue_I : ARRAY[0..2] OF LREAL; // [A] | Rectified value over n periods [A]
    aCrestFactor_I    : ARRAY[0..2] OF LREAL; // []  | Peak_value / RMS
    aFormFactor_I     : ARRAY[0..2] OF LREAL; // []  | RMS / rectified_value
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.5.3 ST_PMA_Frequency_Period_3Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_Frequency_Period_3Ph [▶ 85].

Syntax

Definition:

```
TYPE ST_PMA_Frequency_Period_3Ph :
STRUCT
    aFreq         : ARRAY[0..2] OF LREAL;           // [Hz]   | f     |
 Frequency calculated by zero crossings
    aFreq_Min     : ARRAY[0..2] OF LREAL;           // [Hz]   | f_min | Min value of aFreq
    aFreq_Max     : ARRAY[0..2] OF LREAL;           // [Hz]   | f_max | Max value of aFreq
    aRocof        : ARRAY[0..2] OF LREAL;           // [Hz/s] |       |
 Rate of change of frequency (ROCOF)
    eRotDirection : E_PMA_RotationalDirection_3Ph;  // Rotational direction
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.5.4 ST_PMA_PowerValues_3Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in den Beschreibungen der Funktionsbausteine FB_PMA_PowerValues_Period_3Ph [▶ 96] oder FB_PMA_PowerValues_3Ph [▶ 112].

Syntax

Definition:

```
TYPE ST_PMA_PowerValues_3Ph :
STRUCT
    aApparentPower          : ARRAY[0..2] OF LREAL;          // [VA]    | S         | apparent power
    aApparentPower_1        : ARRAY[0..2] OF LREAL;          // [VA]    | S_1       |
 fundamental apparent power
    aApparentPower_1_Min    : ARRAY[0..2] OF LREAL;          // [VA]    | S_1_min   | Min value of S_1
    aApparentPower_1_Max    : ARRAY[0..2] OF LREAL;          // [VA]    | S_1_max   | Max value of S_1
    aActivePower            : ARRAY[0..2] OF LREAL;          // [W]     | P         | active power
    aActivePower_Min        : ARRAY[0..2] OF LREAL;          // [W]     | P_min     | Min value of P
    aActivePower_Max        : ARRAY[0..2] OF LREAL;          // [W]     | P_max     | Max value of P
    aReactivePower_d        : ARRAY[0..2] OF LREAL;          // [var]   | Q_d       |
 distortion reactive power
    aReactivePower_1        : ARRAY[0..2] OF LREAL;          // [var]   | Q_1       |
 fundamental reactive power
    aReactivePower_1_Min    : ARRAY[0..2] OF LREAL;          // [var]   | Q_1_min   | Min value of Q_1
    aReactivePower_1_Max    : ARRAY[0..2] OF LREAL;          // [var]   | Q1_max    | Max value of Q_1
    aTotalReactivePower     : ARRAY[0..2] OF LREAL;          // [var]   | Q_tot     |
 total reactive power
    aPhi                    : ARRAY[0..2] OF LREAL;          // [°]     | phi       | phase difference
    aCosPhi                 : ARRAY[0..2] OF LREAL;          // []      | COS(phi)  |(P / S_1)
    aPowerFactor            : ARRAY[0..2] OF LREAL;          // []      | PF        |
 power factor (P / S)
    fSumApparentPower       : LREAL;                         // [VA]    | S_sum     |
 Sum of apparent power 1..3
    fSumActivePower         : LREAL;                         // [W]     | P_sum     |
 Sum of active power 1..3
    fSumTotalReactivePower : LREAL;                          // [var]   | Qtot_sum  |
 Sum of total reactive power 1..3
    fSumReactivePower_1     : LREAL;                         // [var]   | Q1_sum    |
 Sum of abs fundamental reactive power 1..3
    aEnergy_Pos             : ARRAY[0..2] OF ST_PMA_Energy; // [kWh]   | W_pos     |
 Energy in positive direction
    aEnergy_Neg             : ARRAY[0..2] OF ST_PMA_Energy; // [kWh]   | W_neg     |
 Energy in negative direction
    aEnergy_Res             : ARRAY[0..2] OF ST_PMA_Energy; // [kWh]   | W         | Resulting energy
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.5.5 ST_PMA_THD_3Ph

Struktur, welche die Ergebnisse des entsprechenden Funktionsbausteins zusammenfasst. Sie kann über das Property stResults abgefragt werden. Details zu den enthaltenen Werten finden Sie im Kommentar oder in der Beschreibung des Funktionsbausteins FB_PMA_Harmonics_Ph3 [▶ 106].

Syntax

Definition:

```
TYPE ST_PMA_THD_3Ph :
STRUCT
    aTHD_U      : ARRAY[0..2] OF LREAL; // [] | THD_U     | Total harmonic distortion of voltage |
 in percent
    aTHD_U_Min  : ARRAY[0..2] OF LREAL; // [] | THD_U_min | Min value of THD_U
    aTHD_U_Max  : ARRAY[0..2] OF LREAL; // [] | THD_U_max | Max value of THD_U
    aTHD_I      : ARRAY[0..2] OF LREAL; // [] | THD_I     | Total harmonic distortion of current |
 in percent
    aTHD_I_Min  : ARRAY[0..2] OF LREAL; // [] | THD_I_min | Min value of THD_I
    aTHD_I_Max  : ARRAY[0..2] OF LREAL; // [] | THD_I_max | Max value of THD_I
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.6 E_PMA_InputSelect

Mit der Enumeration E_PMA_InputSelect kann selektiert werden, ob sich die zu berechnenden Ergebnisse auf die Strom- oder die Spannungswerte beziehen sollen.

Syntax

Definition:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_PMA_InputSelect :
(
     Voltage    := 0,    // Voltage selected
     Current    := 1     // Current selected
) UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.7 ST_PMA_Energy

Struktur zur Ausgabe von Energiewerten. Dabei werden die vollen Kilowattstunden (kWh) als auch die Nachkommastellen ausgegeben.

Syntax

Definition:

```
TYPE ST_PMA_Energy :
STRUCT
    nEnergy               : LINT;   // Energy in kWh
    fEnergyFraction       : LREAL;  // Fraction of energy in kWh
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
nEnergy	LINT	Energie in Kilowattstunden (kWh) als ganzzahliger Wert.
fEnergyFraction	LREAL	Nachkommastellen der Energie in Kilowattstunden (kWh).

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.4.8 E_PMA_PqfMode

Mit der Enumeration E_PMA_PqfMode kann die Berechnungsart für den Power Quality Factor (PQF) eingestellt werden.

Syntax

Definition:

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_PMA_ScalingType :
(
     Default              := 0,
     DefaultAndUnbalance  := 1,
) UDINT;
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

5.5 Globale Konstanten

5.5.1 GVL_PMA

Die Power-Monitoring-Bibliothek enthält folgende Konstanten in der SPS:

```
{attribute 'qualified_only'}
VAR_GLOBAL
    eEventTraceLevel         : TcEventSeverity := TcEventSeverity.Info;
END_VAR
VAR_GLOBAL CONSTANT
    cMA_MaxDest              : UDINT := 20;           // maximum destinations for one analysis block
    cMA_MaxID                : UDINT := 600;          // maximum ID which can be used (=maximum numb
er of analysis blocks
    cMbRMS_MaxBands          : UDINT := 300;          // maximum number of bands usable in multiband
 rms
    cMaxQuantiles            : UDINT := 40;           // maximum number of quantiles
    cMinArgLog10             : LREAL := 2.3E-308;     // min value to calculate logarithm
END_VAR
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

6 Beispiele

Allgemeines

Für ein- bzw. dreiphasige Systeme stehen jeweils zwei Beispiele bereit. Im ersten Beispiel wird die hochdynamische Analyse elektrischer Systeme gezeigt. Im zweiten Beispiel wird eine Kombination aus der hochdynamischen Analyse mit der Analyse im Frequenzbereich verwendet. Aufgrund der höheren benötigten Rechenleistung der frequenzbasierten Algorithmen werden diese in einer separaten, langsameren Task ausgeführt.

6.1 Beispiele der Berechnungen basierend auf der Signalperiode

Die Beispiele zeigen die hochdynamische Analyse elektrischer Systeme. Hierbei wir auf die Algorithmen der Power Monitoring-Bibliothek zurückgegriffen, dessen Berechnungen auf der Signalperiode basieren.

Übersicht

In den Beispielen wird eine Analysekette genutzt. Die Baustein-IDs zur Verknüpfung der Algorithmen sind in der Struktur *E_AnalysisIDs* definiert. Die Analysekette beginnt wahlweise mit einem Eingang aus einer Busklemme, beispielsweise der EL3783, oder mit dem Signalgenerator. Die Umschaltung erfolgt mit der Variable *eInputSelect*.

Das Eingangssignal wird dem Source-Baustein (*fbSource*) übergeben, der es anschließend an die ihm zugewiesenen Analysebausteine weiterreicht. Hierzu gehören die Frequenzberechnung mit dem Baustein (*fbFrequency*), die Berechnung der Basiswerte mit dem Baustein (*fbBasicValues*), die Berechnung von Leistungen mit dem Baustein (*fbPowerValues*) sowie die Berechnung der Harmonischen mit dem Baustein (*fbHarmonics*).

Programmparameter

Die wichtigsten Parameter zur Beeinflussung des Eingangssignals werden in der folgenden Tabelle dargestellt.

Variable	Beschreibung	Standardwert
eInputSelect	Auswahl des Eingangssignals	E_InputSelect.Signal Generator
fFrequency	Grundfrequenz der generierten Signale	50 Hz
fAmplitudeVoltage	Amplitude des generierten Spannungssignals	325,27 V
fAmplitudeCurrent	Amplitude des generierten Stromsignals	1,414 A
fPhaseDifferenceCurrent	Phasenverschiebung zwischen den generierten Strom- und Spannungssignalen	5 °
bEnableHarmonics	Generierung von Harmonischen Anteilen in den Strom- und Spannungssignalen	FALSE

Globale Konstanten

Folgende globale Konstanten werden definiert:

Variable	Beschreibung	Standardwert
cOversamples	Anzahl von Oversamples der Eingangskanäle	10
cSamplerate	Samplerate der Eingangskanäle in Hz	10000
cNumHarmonics	Anzahl der zu berechnenden Harmonischen	20

Download

Beispielprogramm Einphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/5517982603.zip
Beispielprogramm Dreiphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/5517987211.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

6.2 Beispiele der Berechnungen basierend auf dem Frequenzbereich

Die Beispiele zeigen die hochdynamische Analyse elektrischer Systeme in Kombination mit der Analyse im Frequenzbereich.

Übersicht

In den Beispielen werden zwei Analyseketten genutzt. Die Baustein-IDs zur Verknüpfung der Algorithmen sind in der Struktur *E_AnalysisIDs* definiert. Die Analyseketten beginnen wahlweise mit einem Eingang aus einer Busklemme, beispielsweise der EL3783, oder mit dem Signalgenerator. Die Umschaltung erfolgt mit der Variable *eInputSelect*.

Die Analysekette für die hochdynamische Analyse wird nur im *MAIN*-Programm ausgeführt. Das Eingangssignal wird dem Source-Baustein (*fbSource_Period*) übergeben, der es anschließend an die ihm zugewiesenen Analysebausteine weiterreicht. Hierzu gehören die Frequenzberechnung mit dem Baustein (*fbFrequency*), die Berechnung der Basiswerte mit dem Baustein (*fbBasicValues*), die Berechnung von Leistungen (*fbPowerValues*) sowie die Berechnung der Harmonischen mit dem Baustein (*fbHarmonics*). Die Pufferlängen entsprechen dem Oversampling-Faktor.

Die langsamere Analysekette für die Analyse im Frequenzbereich beginnt ebenfalls im *MAIN*-Programm mit dem Source-Baustein (*fbSource*). Dieser sammelt die eingehenden Daten bis zur konfigurierten Pufferlänge *cBufferLength* und schickt diese anschießend an die frequenzbasierten Bausteine zur Leistungsberechnung (*fbPowerValues*), zur Berechnung des Spektrums (*fbSpectrum*) und zur Berechnung der Harmonischen (*fbHarmonics*) in das langsamere Programm *MAIN_SLOW*.

Programmparameter

Die wichtigsten Parameter zur Beeinflussung des Eingangssignals werden in der folgenden Tabelle dargestellt.

Variable	Beschreibung	Standardwert
eInputSelect	Auswahl des Eingangssignals	E_InputSelect.Signal Generator
fFrequency	Grundfrequenz der generierten Signale	50 Hz
fAmplitudeVoltage	Amplitude des generierten Spannungssignals	325,27 V
fAmplitudeCurrent	Amplitude des generierten Stromsignals	1,414 A
fPhaseDifferenceCurrent	Phasenverschiebung zwischen den generierten Strom- und Spannungssignalen	5 °
bEnableHarmonics	Generierung von Harmonischen Anteilen in den Strom- und Spannungssignalen	FALSE

Globale Konstanten

Folgende globale Konstanten werden definiert:

Variable	Beschreibung	Standardwert
cOversamples	Anzahl von Oversamples der Eingangskanäle	10
cSamplerate	Samplerate der Eingangskanäle in Hz	10000
cFFTLength	Länge der FFT	4096
cWindowLength	Interne Pufferlänge mit 50% Überlappung	3200
cBufferLength	Länge des Eingangspuffers für die FFT-Berechnung	1600
cHarmonicBands	Anzahl der zu berechnenden Harmonischen	20
cFreqMode	Grundfrequenz für die Berechnung der Harmonischen	50.0

Download

Beispielprogramm Einphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/5517984907.zip
Beispielprogramm Dreiphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/5517989515.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

6.3 Beispiele Wiederverwendung

Die Beispiele zeigen die hochdynamische Analyse elektrischer Systeme in Kombination mit der Analyse im Frequenzbereich. Zusätzlich wird dargestellt, wie die Analyseketten gekapselt und somit wiederverwendet werden können.

Übersicht

In diesem Beispiel werden jeweils zwei Analyseketten für die hochdynamische Analyse sowie zwei Analyseketten für die Analyse im Frequenzbereich implementiert. Die Funktionsbausteine *FB_PowerMonitoring_Fast* und *FB_PowerMonitoring_Slow* kapseln die Analyseketten und stellen die Möglichkeit bereit, die Analyseketten mehrfach zu instanziieren. Dazu wird eine zusätzliche ID benötigt, die intern einen Offset auf die Baustein-IDs addiert. Die zusätzliche ID wird in den Programmen *MAIN* sowie *MAIN_SLOW* vergeben. Die Baustein-IDs zur Verknüpfung der Algorithmen sind in den Strukturen *E_AnalysisIDs_Fast* und *E_AnalysisIDs_Slow* definiert.

Die Analyseketten beginnen wahlweise mit einem Eingang aus einer Busklemme, beispielsweise der EL3783, oder mit dem Signalgenerator. Die Umschaltung erfolgt mit der Variable *eInputSelect*.

Die Analysekette für die hochdynamische Analyse wird nur im *MAIN*-Programm ausgeführt. Das Eingangssignal wird dem Source-Baustein (*fbSource_Period*) übergeben, der es anschließend an die ihm zugewiesenen Analysebausteine weiterreicht. Hierzu gehören die Frequenzberechnung mit dem Baustein (*fbFrequency*), die Berechnung der Basiswerte mit dem Baustein (*fbBasicValues*), die Berechnung von Leistungen (*fbPowerValues*) sowie die Berechnung der Harmonischen mit dem Baustein (*fbHarmonics*). Die Pufferlängen entsprechen dem Oversampling-Faktor.

Die langsamere Analysekette für die Analyse im Frequenzbereich beginnt ebenfalls im *MAIN*-Programm mit dem Source-Baustein (*fbSource*). Dieser sammelt die eingehenden Daten bis zur konfigurierten Pufferlänge *cBufferLength* und schickt diese anschießend an die frequenzbasierten Bausteine zur Leistungsberechnung (*fbPowerValues*), zur Berechnung des Spektrums (*fbSpectrum*) und zur Berechnung der Harmonischen (*fbHarmonics*) in das langsamere Programm *MAIN_SLOW*.

Programmparameter

Die wichtigsten Parameter zur Beeinflussung des Eingangssignals werden in der folgenden Tabelle dargestellt.

Variable	Beschreibung	Standardwert
eInputSelect	Auswahl des Eingangssignals	E_InputSelect.SignalGenerator
fFrequency	Grundfrequenz der generierten Signale	50 Hz
fAmplitudeVoltage	Amplitude des generierten Spannungssignals	325,27 V
fAmplitudeCurrent	Amplitude des generierten Stromsignals	1,414 A
fPhaseDifferenceCurrent	Phasenverschiebung zwischen den generierten Strom- und Spannungssignalen	5 °
bEnableHarmonics	Generierung von Harmonischen Anteilen in den Strom- und Spannungssignalen	FALSE

Globale Konstanten

Folgende globale Konstanten werden definiert:

Variable	Beschreibung	Standardwert
cOversamples	Anzahl von Oversamples der Eingangskanäle	10
cSamplerate	Samplerate der Eingangskanäle in Hz	10000
cFFTLength	Länge der FFT	4096
cWindowLength	Interne Pufferlänge mit 50% Überlappung	3200
cBufferLength	Länge des Eingangspuffers für die FFT-Berechnung	1600
cHarmonicBands	Anzahl der zu berechnenden Harmonischen	20
cFreqMode	Grundfrequenz für die Berechnung der Harmonischen	50.0

Download

Beispielprogramm Einphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/13424000011.zip
Beispielprogramm Dreiphasig	https://infosys.beckhoff.com/content/1031/TF3650_TC3_Power_Monitoring/Resources/13424000523.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.0	PC oder CX (x86, x64)	Tc3_PowerMonitoring

7 Anhang

7.1 FAQ

In diesem Bereich werden häufig gestellte Fragen beantwortet, um Ihnen die Arbeit mit der TwinCAT-3-Power-Monitoring-Bibliothek zu erleichtern.
Wenn Sie weitere Fragen haben, kontaktieren Sie unseren Support (-157).

Welche Kennwerte können mit der TwinCAT-3-Power-Monitoring-Bibliothek berechnet werden? [▶ 152]

Welche Kennwerte können mit der TwinCAT-3-Power-Monitoring-Bibliothek berechnet werden?

Folgenden Kennwerte können berechnet werden: Effektiv-, Mittel-, Maximal- und Minimalwerte von Strom, Spannung, Wirk-, Blind-, Verzerrungsblind- und Scheinleistung sowie Frequenzen und Frequenzspektren, Total Harmonic Distortion und die Harmonische.

7.2 Support und Service

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

Beckhoff Niederlassungen und Vertretungen

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

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unseren Internetseiten: https://www.beckhoff.de

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

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

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

Hotline:	+49(0)5246 963 157
Fax:	+49(0)5246 963 9157
E-Mail:	support@beckhoff.com

Beckhoff Service

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

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

Hotline:	+49(0)5246 963 460
Fax:	+49(0)5246 963 479
E-Mail:	service@beckhoff.com

Beckhoff Firmenzentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49(0)5246 963 0
Fax:	+49(0)5246 963 198
E-Mail:	info@beckhoff.com
Internet:	https://www.beckhoff.de

Mehr Informationen:
www.beckhoff.de/tf3650

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