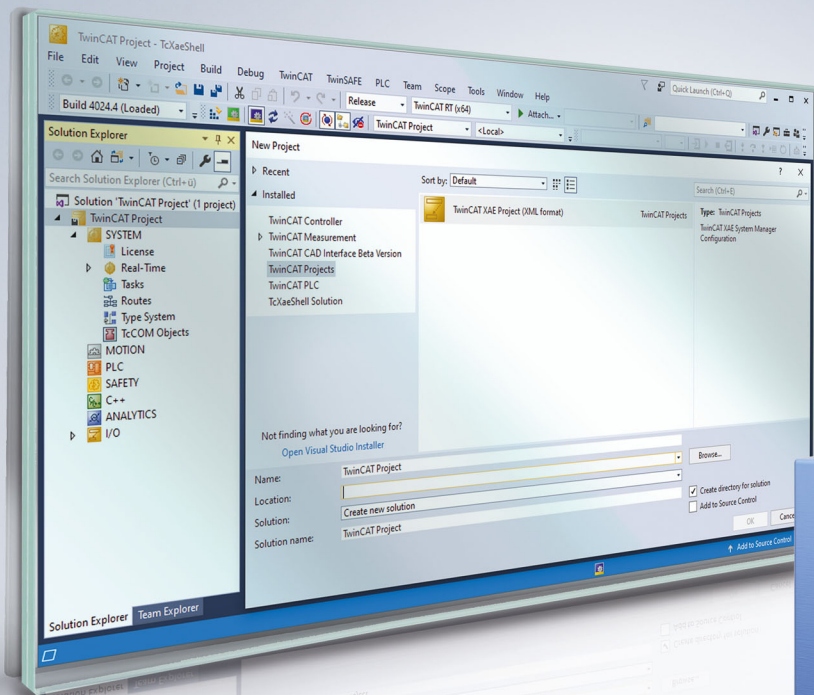


BECKHOFF New Automation Technology

Handbuch | DE

TF3500

TwinCAT 3 | Analytics Logger



Inhaltsverzeichnis

1	Vorwort	5
1.1	Hinweise zur Dokumentation	5
1.2	Zu Ihrer Sicherheit.....	6
1.3	Hinweise zur Informationssicherheit	7
2	Übersicht	8
3	Installation	9
3.1	Systemvoraussetzungen	9
3.2	Lizenzierung	9
4	Analytics Workflow – Erste Schritte	12
4.1	Aufzeichnung von Maschinendaten	12
4.2	Kommunikation	15
4.3	Historisierung von Daten	16
4.4	Analyse der Daten.....	23
4.5	24-stündige Anwendung von Analytics	28
5	Technische Einführung	38
5.1	Grundkonzepte.....	38
5.2	MQTT Grundlagen	39
5.3	Datenkompression	45
6	Konfiguration	46
6.1	Basiseinstellungen	46
6.1.1	TLS.....	48
6.1.2	Zeitstempelkorrektur	51
6.1.3	Gerätespezifische Informationen	53
6.2	Daten-Streams	54
6.2.1	Data Handling	57
7	API	60
7.1	SPS	60
7.1.1	Analytics Communication Library.....	60
7.1.2	Obsolete.....	84
7.2	Automation Interface	84
8	Beispiele	85
9	Anhang	86
9.1	FAQ – Häufig gestellte Fragen und Antworten	86

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

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

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

Disclaimer

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

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

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

Marken

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

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

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702

mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.



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

Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

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

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

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

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

Qualifikation des Personals

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

Signalwörter

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

Warnungen vor Personenschäden

GEFAHR

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

WARNUNG

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

VORSICHT

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

Warnung vor Umwelt- oder Sachschäden

HINWEIS

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

Information zum Umgang mit dem Produkt



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

1.3 Hinweise zur Informationssicherheit

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

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

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

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

2 Übersicht

Mit dem TwinCAT Analytics Logger ist die zyklussynchrone Erfassung von Prozess- und Applikationsdaten der Maschinensteuerung möglich. Charakteristisch für den Logger ist seine hohe Leistung, da er direkt im Echtzeitkontext der TwinCAT-Steuerung arbeitet.

Der TwinCAT Analytics Logger kann entweder als MQTT-Client fungieren und die Daten regelmäßig an einen MQTT-Message-Broker übertragen (MQTT-Modus) oder die Daten lokal in einer Datei auf der Festplatte der Maschinensteuerung speichern (Dateimodus). Die erforderliche Konfiguration erfolgt in Microsoft Visual Studio®. Alle Variablen des Prozessabbaus und der SPS-Anwendung können über Kontrollkästchen einfach der Konfiguration hinzugefügt werden, ohne dass Programmierung erforderlich ist.

Bei Verwendung als MQTT-Client kann der Logger kurze Unterbrechungen der Verbindung zum Message-Broker mit Hilfe einer Ringpuffer-Funktionalität überbrücken, um den Verlust von Daten kurzzeitig zu verhindern. Bei Verwendung im Dateimodus kann ebenfalls ein Ringpuffer konfiguriert werden, der hilfreich sein kann, wenn die Speicherkapazität begrenzt ist oder wenn keine Daten dauerhaft aufgezeichnet (im Folgenden als „geloggt“ bezeichnet) werden müssen, sondern der Fokus auf einem festen Zeitintervall liegt.

Die geloggt Daten können auf verschiedene Weise genutzt werden, aber ihr Hauptzweck ist die Datenanalyse mit TwinCAT Analytics sowie die Datenvisualisierung mit TwinCAT Scope.

Komponenten

- Konfigurationsoberfläche im TwinCAT-Projektbaum
- Beschreibungsdateien TcAnalytics.tmc und TcIotBase.tmc
- Treiber TcAnalytics.sys und TcIotDrivers.sys

Liste der wesentlichen Merkmale

Funktionalität	TC3 Analytics Logger als MQTT-Client	TC3 Analytics Logger für lokale Speicherung
Programmierbare Aufzeichnungssteuerung	Ja	Ja
Konfigurationsschnittstelle	Ja	Ja
RT-Kontext	Ja	Ja
MQTT	Ja	Nein
Analytik-Binärformat	Ja	Ja
JSON-Format	Nein	Nein
Dateispeicherung	Nein	Ja
Ringpuffer	Ja	Ja
Authentifizierung	Ja	Nein
Verschlüsselung	Ja	Ja
Kompression	Ja	Ja

3 Installation

Der TwinCAT Analytics Logger wird mit TwinCAT XAE und XAR installiert. Daher sollte der Logger immer verfügbar sein. Um ihn zu benutzen, ist eine „TC3 Analytics Logger“-Lizenz pro Zielgerät erforderlich, wobei es sich entweder um eine unbefristete Lizenz oder eine 7-tägige Testlizenz handeln kann.

Allgemeine Informationen über die Lizenzierung finden Sie im Abschnitt „Lizenzierung“ weiter unten.

3.1 Systemvoraussetzungen

Technische Daten	TF3500 TC3 Analytics Logger
Betriebssystem	Windows 7, Windows 8, Windows 10, WinCE, TwinCAT/BSD PC (x86, x64 und ARM)
Min. TwinCAT-Version	3.1.4022.31
Min. TwinCAT Level	TC1100 TC3 I/O

3.2 Lizenzierung

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

Lizenzierung der Vollversion einer TwinCAT 3 Function

Die Beschreibung der Lizenzierung einer Vollversion finden Sie im Beckhoff Information System in der Dokumentation „[TwinCAT 3 Lizenzierung](#)“.

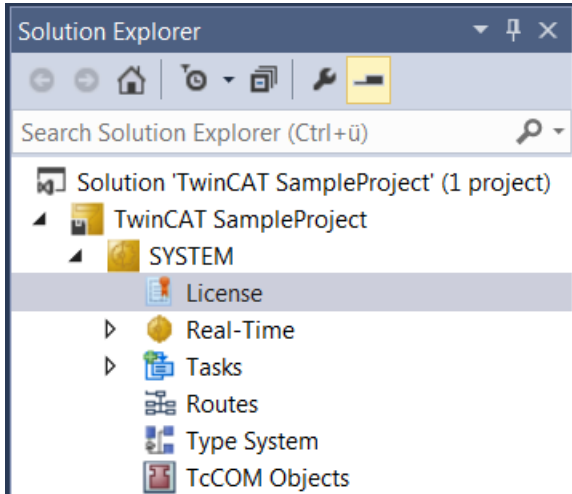
Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function



Eine 7-Tage-Testversion kann nicht für einen [TwinCAT-3-Lizenz-Dongle](#) freigeschaltet werden.

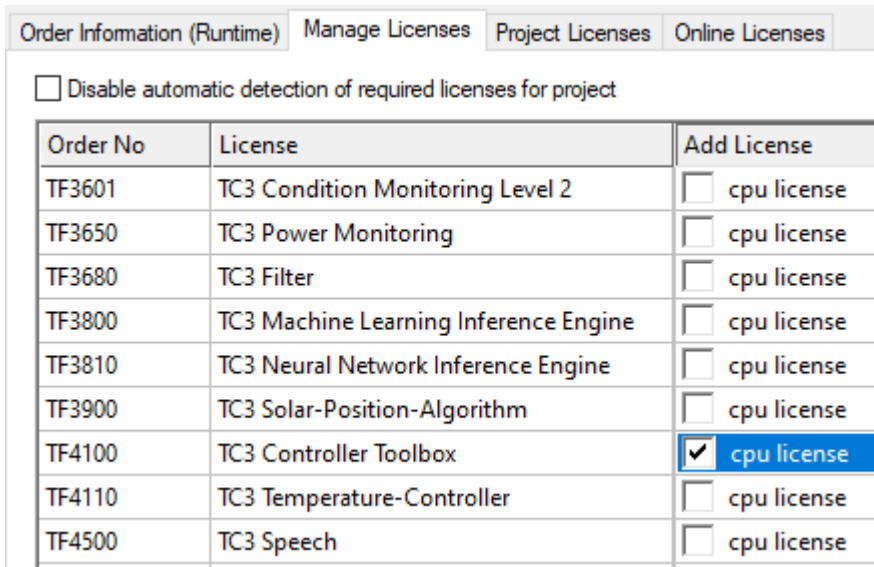
1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).
2. Öffnen Sie ein bestehendes TwinCAT-3-Projekt oder legen Sie ein neues Projekt an.
3. Wenn Sie die Lizenz für ein Remote-Gerät aktivieren wollen, stellen Sie das gewünschte Zielsystem ein. Wählen Sie dazu in der Symbolleiste in der Drop-down-Liste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

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



⇒ Der TwinCAT-3-Lizenzmanager öffnet sich.

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



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

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

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

The screenshot shows a software interface with several sections:

- Order Information (Runtime)**: Includes tabs for 'Manage Licenses', 'Project Licenses', and 'Online Licenses'. Below are fields for 'License Device' (set to 'Target (Hardware Id)'), 'System Id' (containing '2DB25408-B4CD-81DF-5488-6A3D9B49EF19'), and 'Platform' (set to 'other (91)').
- License Request**: Includes a 'Provider' dropdown set to 'Beckhoff Automation', a 'Generate File...' button, and input fields for 'License Id', 'Customer Id', and 'Comment'.
- License Activation**: This section is highlighted with a red box and contains two buttons: '7 Days Trial License...' and 'License Response File...'.

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

The dialog box is titled 'Enter Security Code' and contains the following elements:

- A prompt: 'Please type the following 5 characters:'
- A text box containing the code 'Kg8T4'.
- An input field with a red border, currently empty.
- 'OK' and 'Cancel' buttons.

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 Analytics Workflow – Erste Schritte

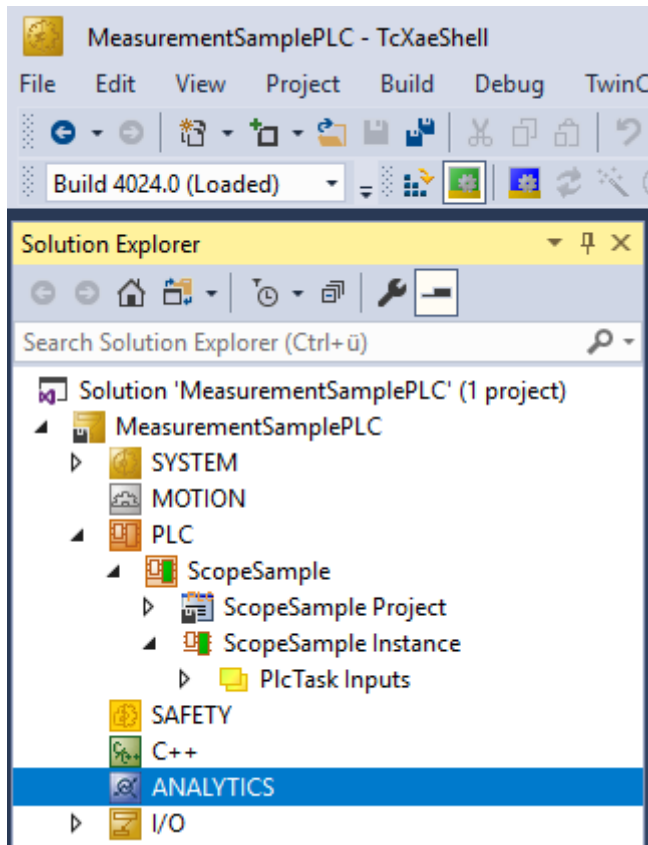
Diese Dokumentation präsentiert Schritt für Schritt den vollständigen TwinCAT Analytics Workflow. Von der Datenerfassung über die Kommunikation und Historisierung bis hin zur Auswertung und Analyse der Daten sowie zur Präsentation der Daten im webbasierten Dashboard.

4.1 Aufzeichnung von Maschinendaten

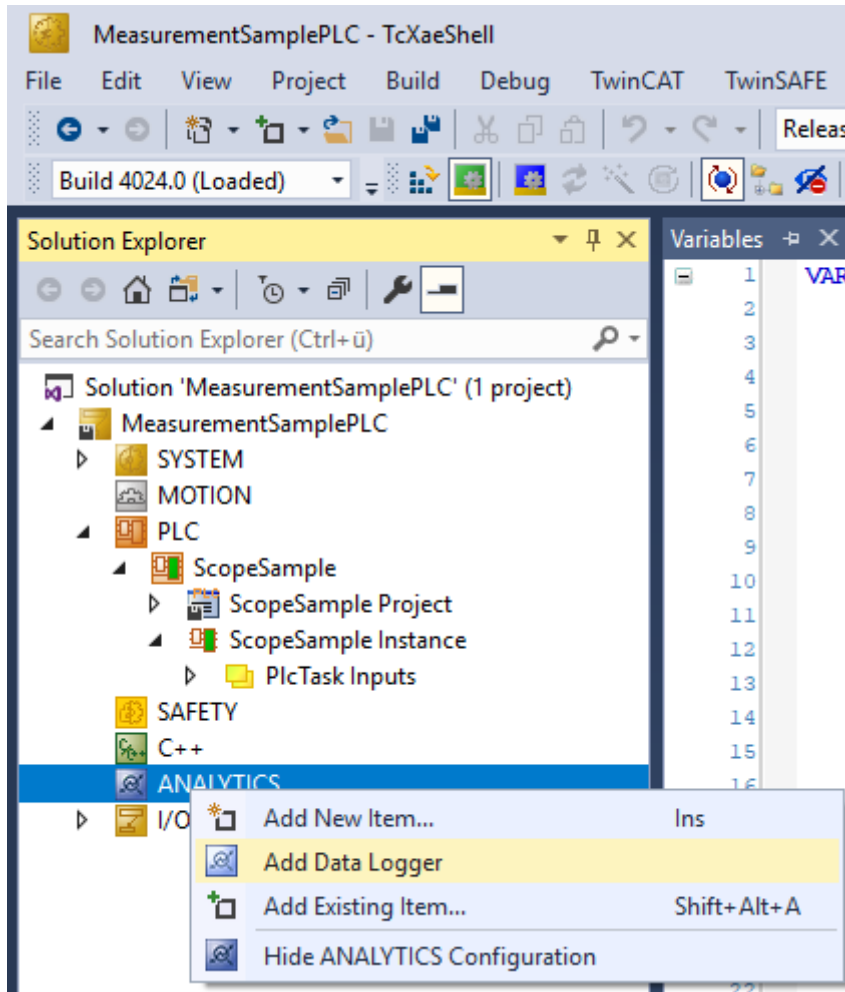
Maschinenseitig ist der Analytics Logger die Aufzeichnungseinrichtung von Prozessdaten des Maschinenabbaus, SPS, NC usw. Der Logger arbeitet im Echtzeitkontext von TwinCAT.

Der TwinCAT Analytics Logger wird mit TwinCAT XAE und XAR installiert. Der Logger kann als MQTT-Client fungieren, um die aufgezeichneten Daten an einen nativen MQTT-Message-Broker zu übermitteln, oder die Daten im selben Datenformat in einer lokalen Binärdatei zu speichern. Bei Verwendung als MQTT-Client kann der Logger kurze Unterbrechungen der Verbindung zum Message-Broker mit Hilfe einer Ringpuffer-Funktionalität überbrücken. Auch für den lokalen Binärdateispeicher kann ein Ringpuffer konfiguriert werden.

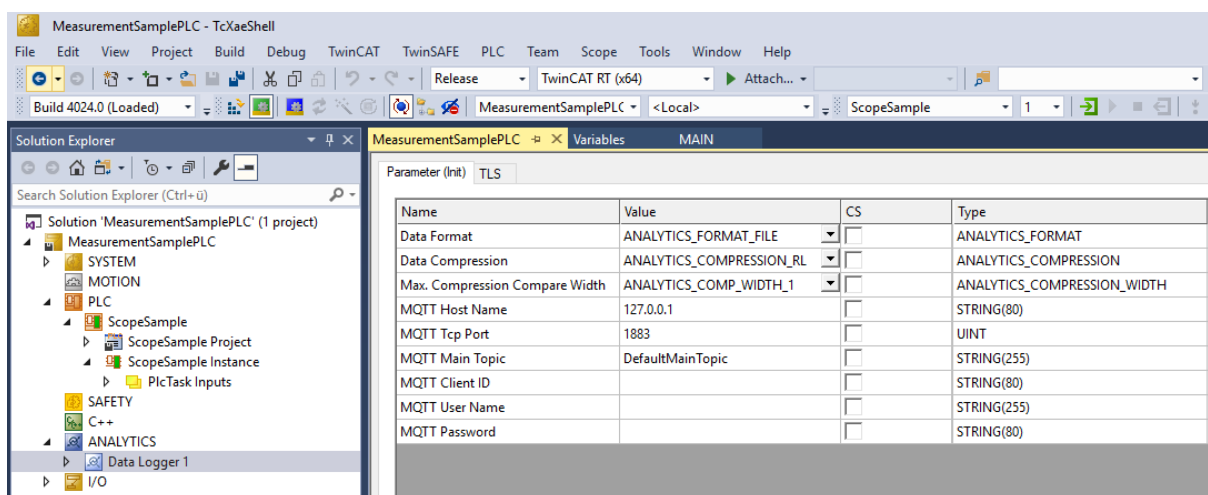
- Um den Analytics Logger zu konfigurieren, müssen Sie in Ihrem vorhandenen TwinCAT-Projekt zum Analytics-Baumknoten navigieren.



- Klicken Sie mit der rechten Maustaste auf diesen Knoten und klicken Sie auf **Add Data Logger**, um Ihrer Konfiguration eine neue Instanz hinzuzufügen.



- Für die Konfiguration der Grundeinstellungen doppelklicken Sie auf das neue Tree Item.



Sie können Ihre spezifischen Analytics Logger-Einstellungen vornehmen.

-Datenformat: Binärdatei oder MQTT-Stream.

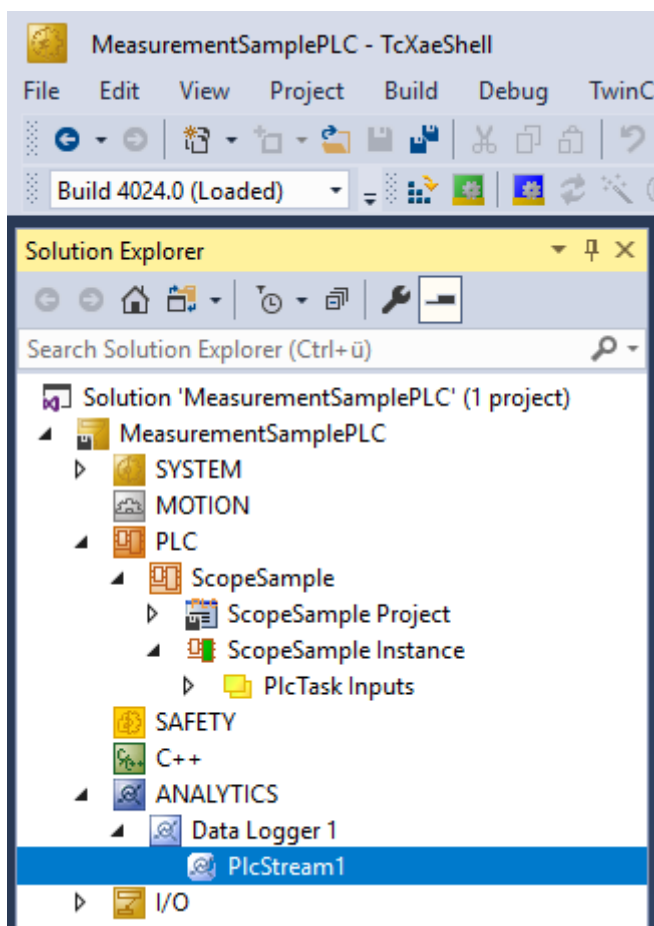
-FILE-Format: Der Analytics Logger speichert die Daten in lokalen Binärdateien und alle anderen Einstellungen sind nicht mehr notwendig. Die Dateien werden in *C:\TwinCAT\3.1\Boot\Analytics* gespeichert.

-BINARY-Format: Die Daten werden an den konfigurierten MQTT-Message-Broker gesendet. Sie können mehrere Logger in einem TwinCAT-Projekt vorsehen, um Daten an verschiedene MQTT-Message-Broker zu übermitteln.

- Datenkompression: ein (Standard) oder aus.
- Max. Kompression: Modus der Kompression.
- MQTT-Hostname
- MQTT-Tcp-Port
- MQTT-Maintopic für eigene Hierarchieebenen, damit die Identifikation einfach bleibt.
- MQTT-Client-ID, sollte im Netzwerk eindeutig sein.
- MQTT-Benutzername
- MQTT-Passwort für die Authentifizierung beim Message-Broker.

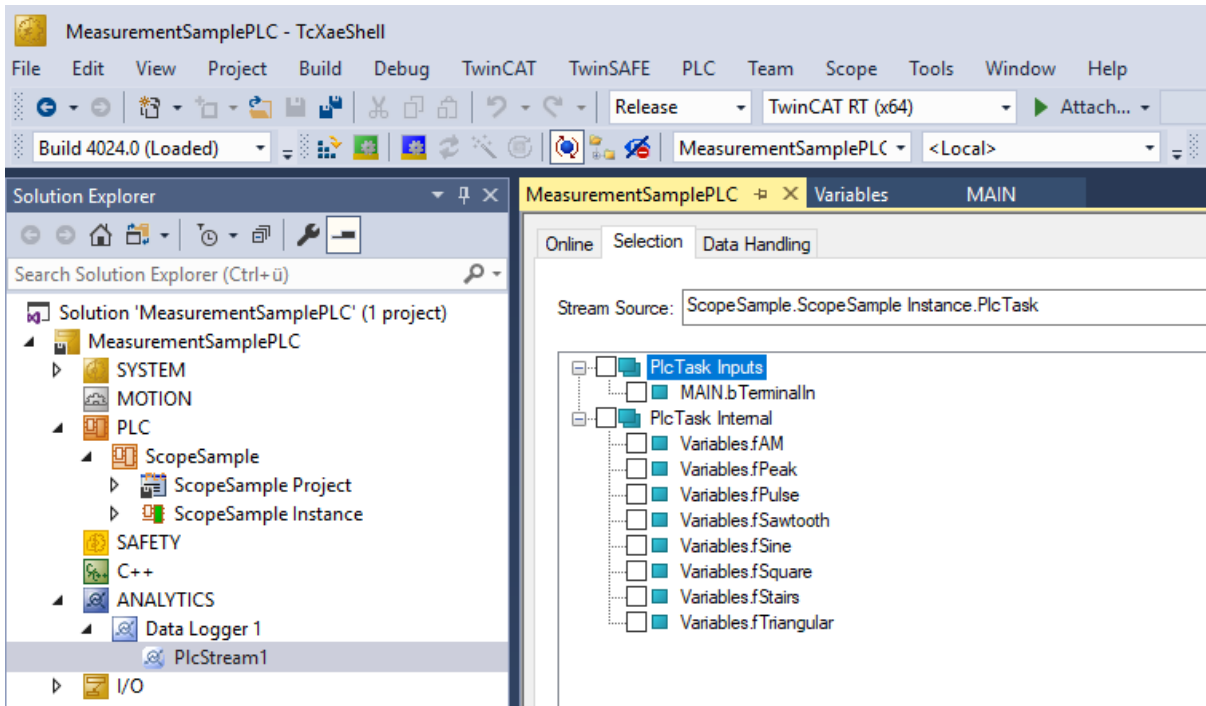
-Auf der Registerkarte **TLS** (Transport Layer Security) können die Sicherheitseinstellungen konfiguriert werden. TLS ist ein sicherer Kommunikationskanal zwischen Client und Server. Bei Verwendung von Zertifikaten ist der TCP-Port 8883 ausschließlich für MQTT über TLS reserviert! Der Analytics Logger unterstützt die Modi CA Certificates, CA Certificates & Client Certificate und Preshared Key (PSK).

- Wenn Variablen in Ihrer SPS-Anwendung in der Deklaration mit dem Attribut {attribute 'TcAnalytics'} gekennzeichnet sind, werden sie automatisch als Stream unter dem Daten-Logger-Baumknoten angezeigt.

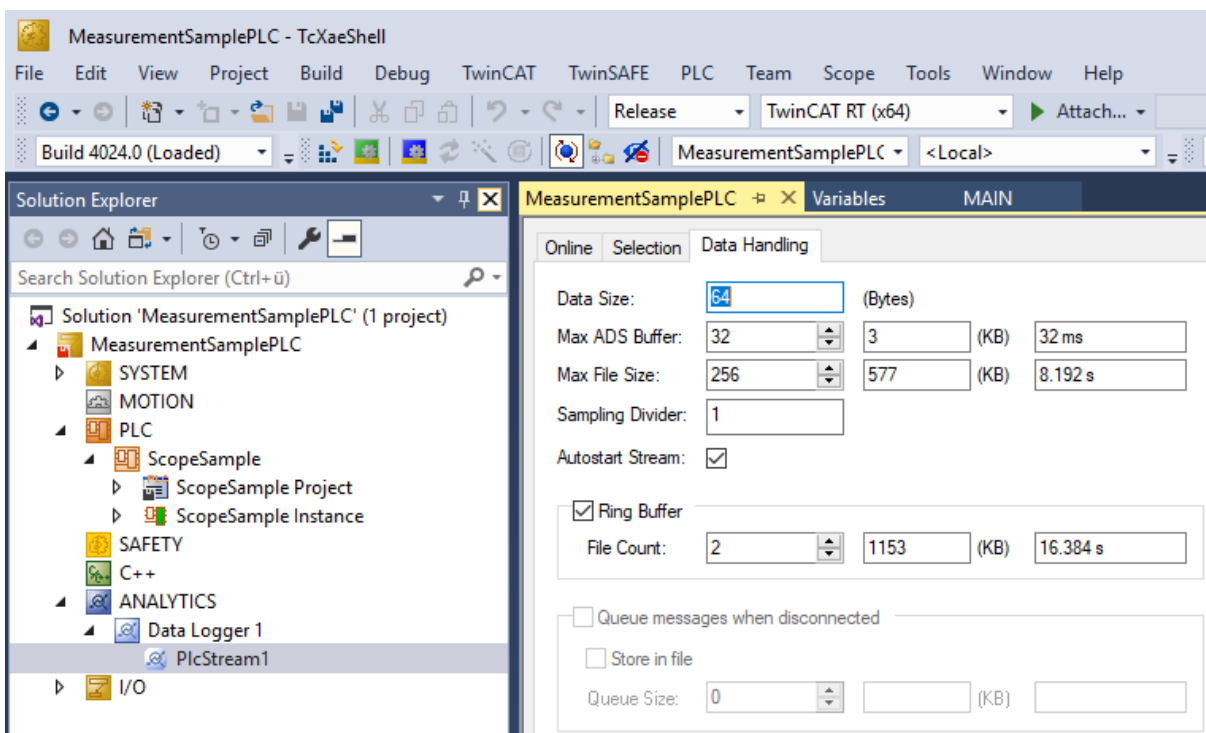


Ein zusätzlicher Geräte-Stream wird angezeigt, wenn Ihre Konfiguration ein EtherCAT-Prozessabbild liefert.

- In dem Stream gibt es einen Karteireiter **Selection**, wo die aufzuzeichnenden Variablen ausgewählt werden können.

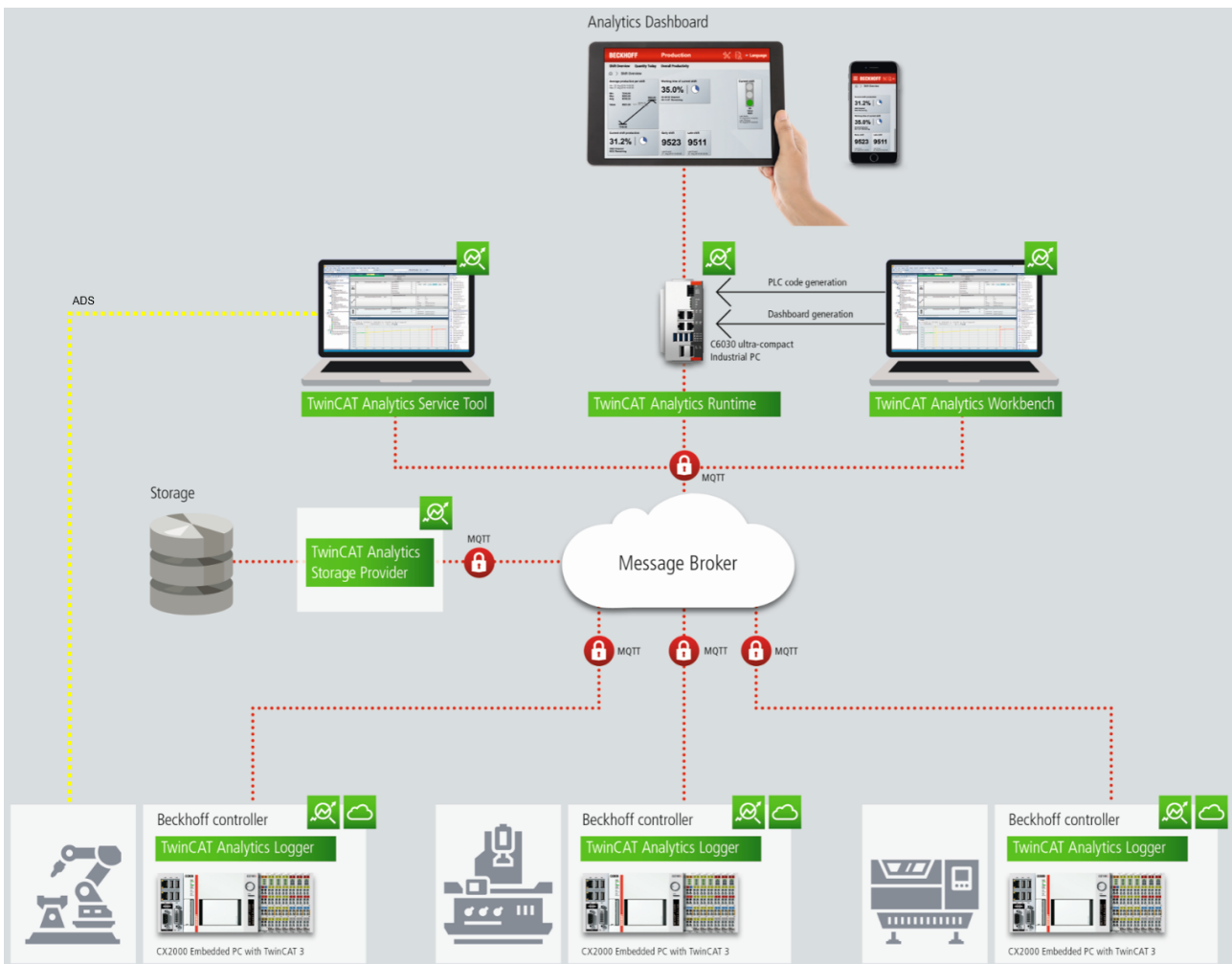


- Schließlich ist es im Karteireiter **Data Handling** möglich, die Paketgröße für die Frames zu ändern oder den Ringpuffer für Verbindungsunterbrechungen und Datei zu konfigurieren.



4.2 Kommunikation

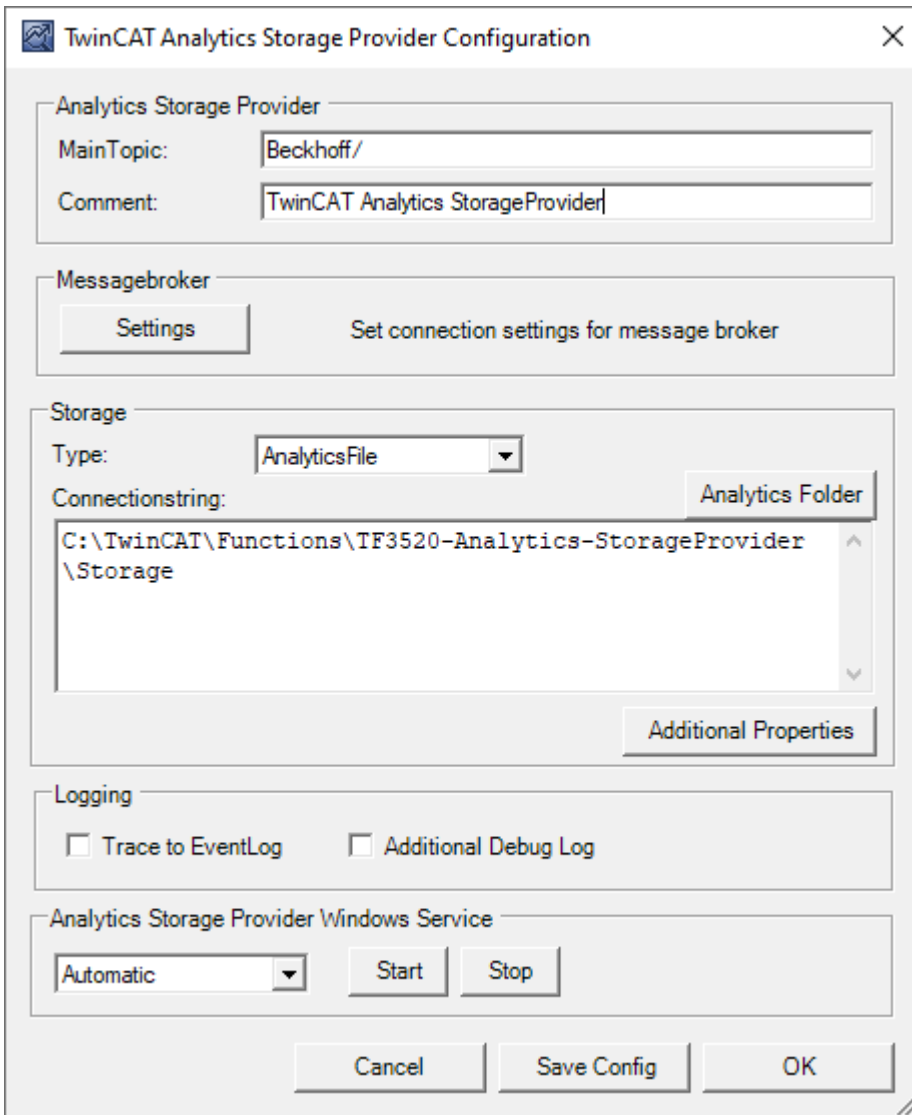
Derzeit kann der Analytics-Workflow vollständig über MQTT abgebildet werden. Die Engineering-Tools können auch über ADS auf die Daten der Maschinen zugreifen und Analysen durchführen.



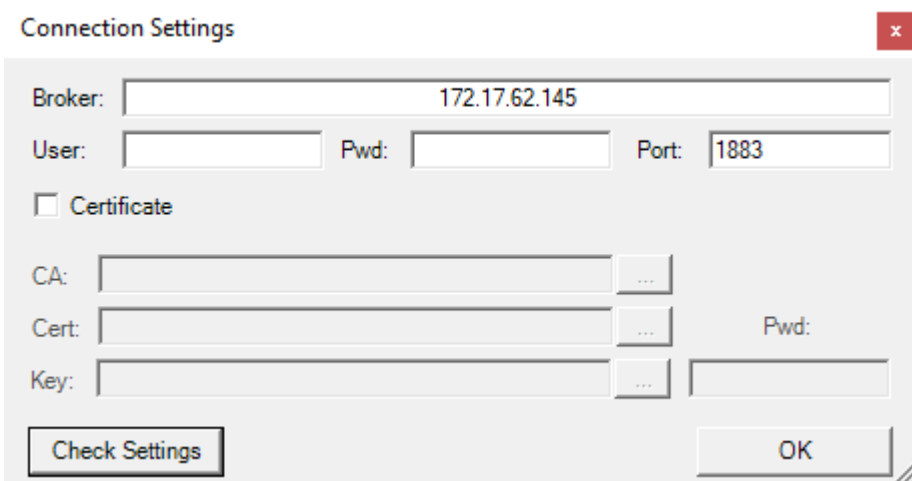
Wenn man für das IoT-Kommunikationsprotokoll MQTT wählt, muss irgendwo im Netzwerk ein nativer MQTT-Message-Broker eingerichtet werden (VM in einem Cloud-System ist ebenfalls möglich). Dieser Message-Broker sorgt für eine Entkopplung der verschiedenen Anwendungen im Analytics-Workflow.

4.3 Historisierung von Daten

Nach der Installation des TwinCAT Analytics Storage Provider kann der im Hintergrund laufende Dienst konfiguriert werden. Hierzu finden Sie die Anwendung TcAnalyticsStorageProvider_Config im Ordner C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\WinService.



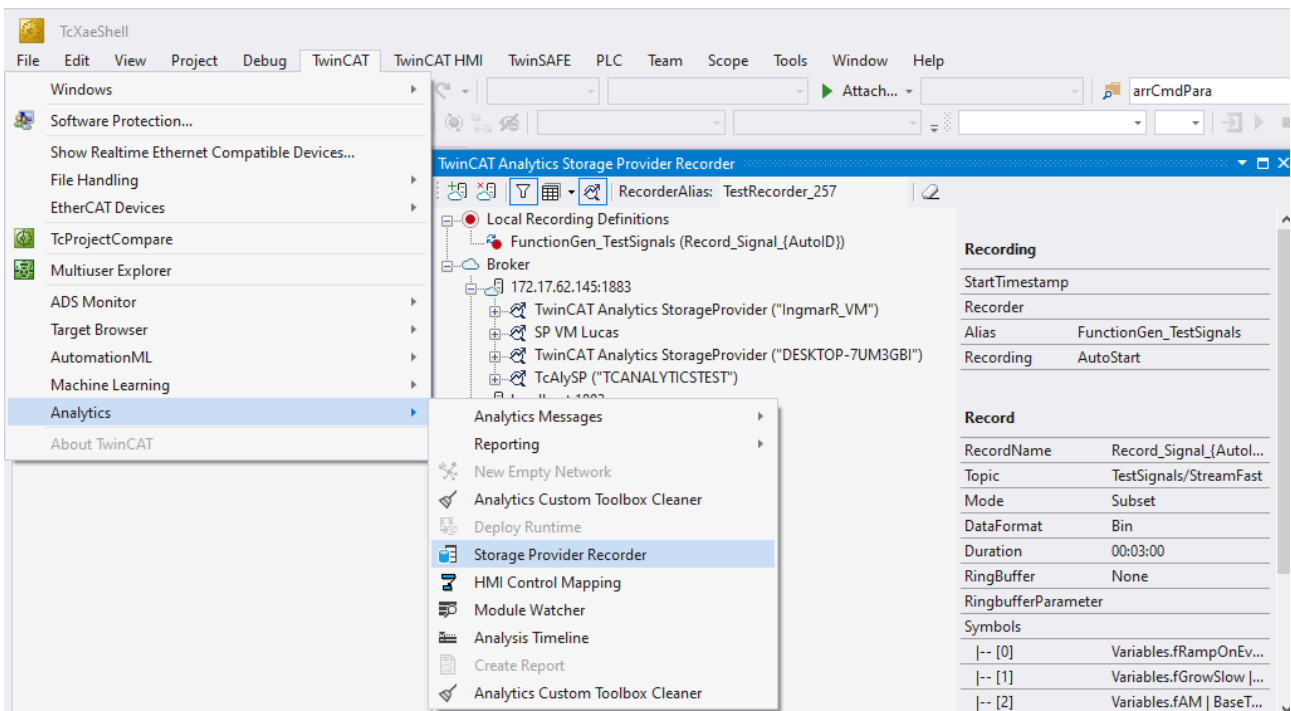
Der Hauptteil des Topics kann in der Konfiguration ebenso festgelegt werden wie der Kommentar, der zur Identifizierung dient, wenn mehr als ein Storage Provider beim Message-Broker registriert ist.



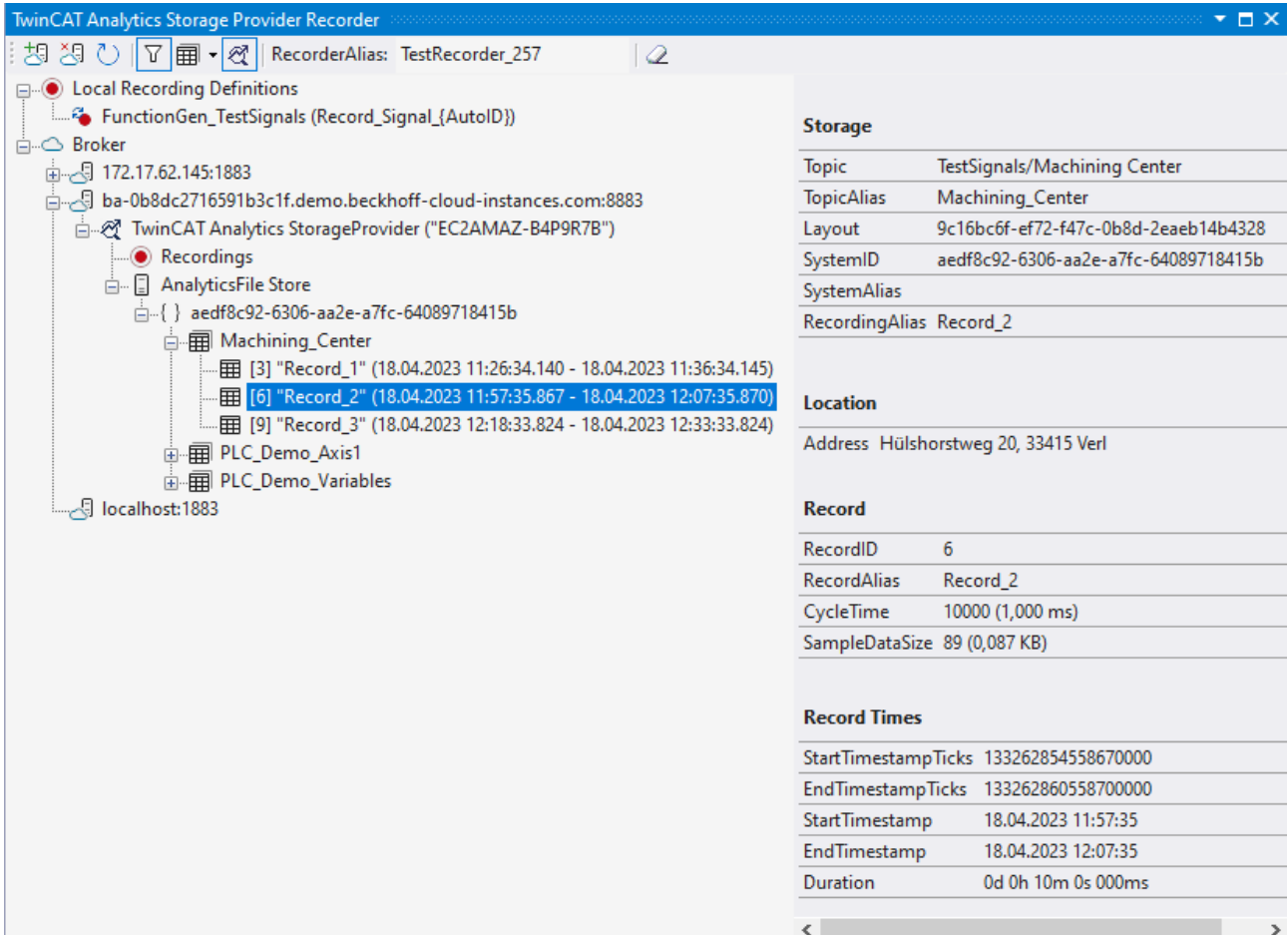
Sie können die Message-Broker-Einstellungen vornehmen und sich für einen Speichertyp entscheiden:

- Analytics File (Binärdatei)
- Microsoft SQL (Binär)
- Microsoft Azure Blob (Azure-Cloud erforderlich)

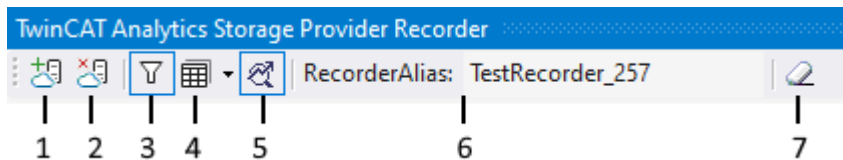
Zuletzt können Sie die Konfiguration speichern und den Dienst starten. Der nächste Schritt besteht in der Konfiguration der spezifischen Aufzeichnung. Hierzu sollten Sie in Ihrer Entwicklungsumgebung den **Storage Provider Recorder** auswählen.



Mit dem Storage Provider Recorder können Aufnahme Definitionen erstellt, gestartet und verwaltet werden. Zusätzlich besteht die Möglichkeit, die Datenspeicher einzelner Analytics Storage Provider zu verwalten. Alle wichtigen Eigenschaften der gefundenen Analytics Storage Providern und historisierten Daten werden übersichtlich dargestellt.



Symbolleiste

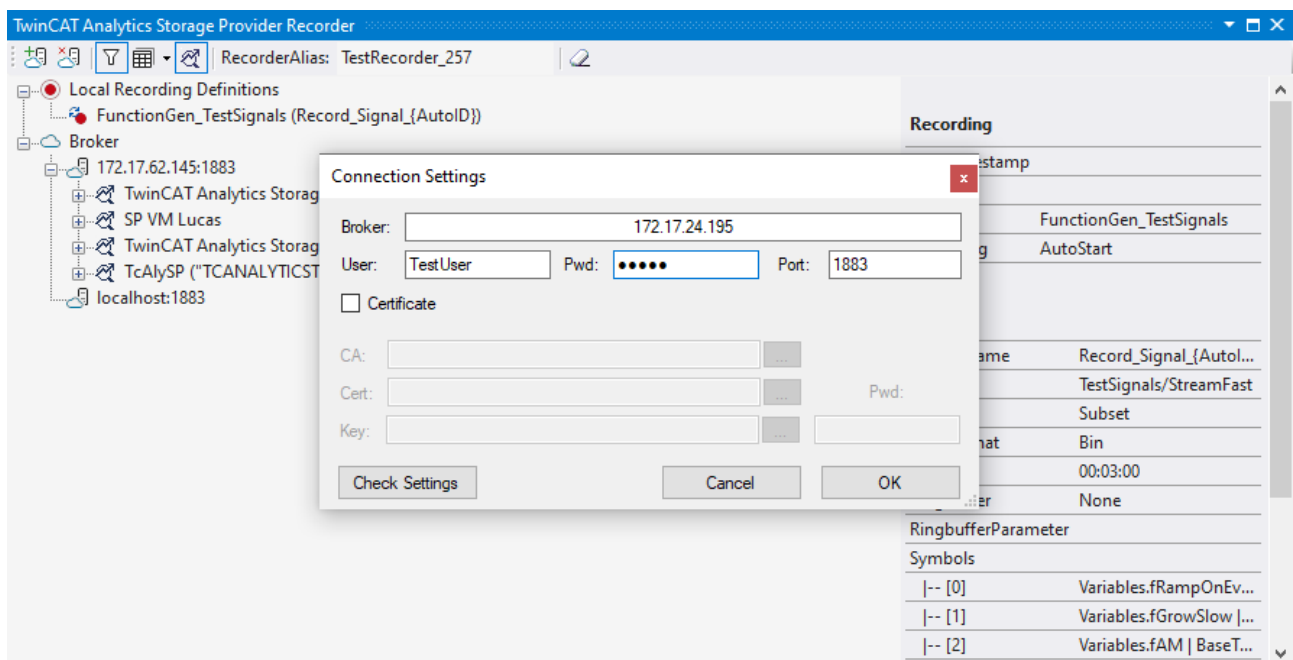


1	Neuen Broker hinzufügen
2	Ausgewählten Broker entfernen
3	Anzeige Filter (Alle / Meine aktiven Aufnahmen anzeigen)
4	Anzeige Art wählen (Alias / Topics)
5	Offline Analytics Storage Provider anzeigen / ausblenden
6	„RecorderAlias“ – Gruppierungsname von Aufnahmen
7	Meldungen aus Fehlerliste entfernen

Recorder Fenster einrichten

Vergeben Sie zunächst einen „RecorderAlias“. Dieser hilft die gestarteten Aufnahmen zu gruppieren und seine selbst gestarteten wieder zu finden. Mit dem Filter können auch fremd gestartete Aufnahmen angezeigt werden.

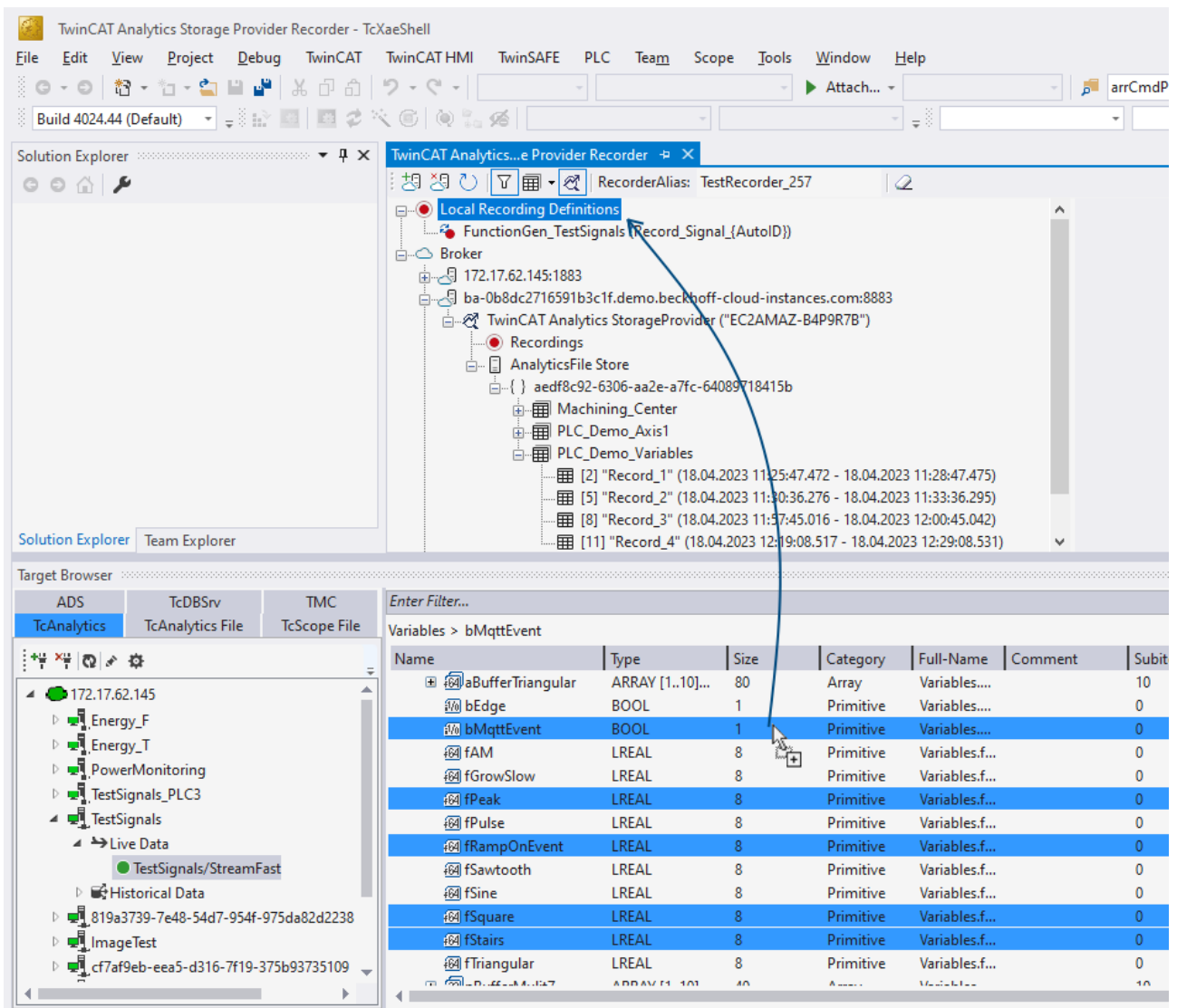
Danach können ein oder mehrere Broker eingerichtet werden. Dies geschieht über die schon bekannte Eingabemaske für MQTT-Verbindungseigenschaften.



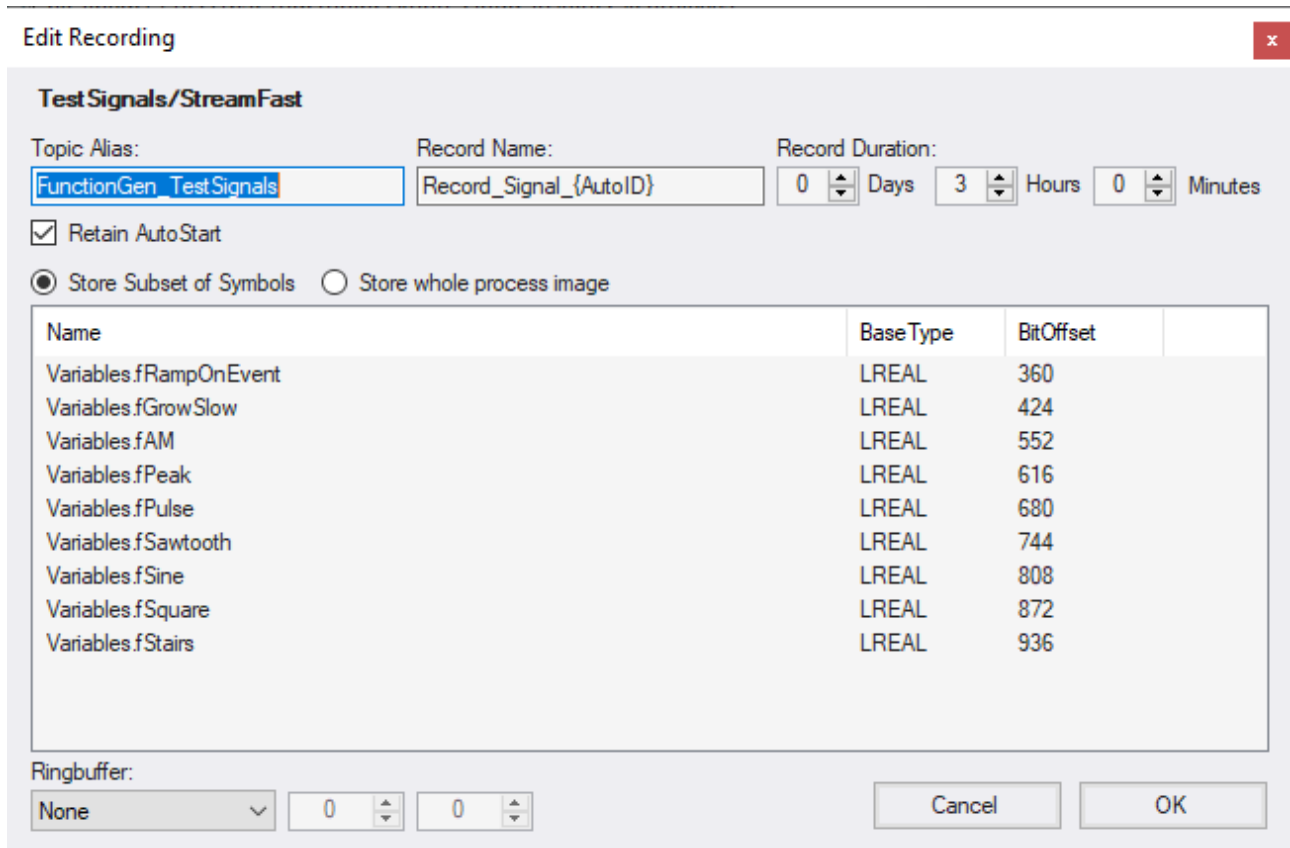
Sobald eine Verbindung zum Broker hergestellt werden konnte, werden alle darauf verbundenen Analytics Storage Provider aufgelistet.

Aufnahmedefinition erzeugen

Zur Konfiguration der Aufzeichnung wählen Sie Ihr Ziel im Target Browser aus. Klicken Sie auf **Live Data** und wählen Sie eine oder mehrere Variablen durch Mehrfachauswahl aus und ziehen Sie sie per Drag-and-Drop auf den **Local Recording Definitions** Knoten im Recorder Fenster.



Im Recorder können Sie die ausgewählten Variablen oder das vollständige Quell-Prozessabbild der Variablen hinzufügen.

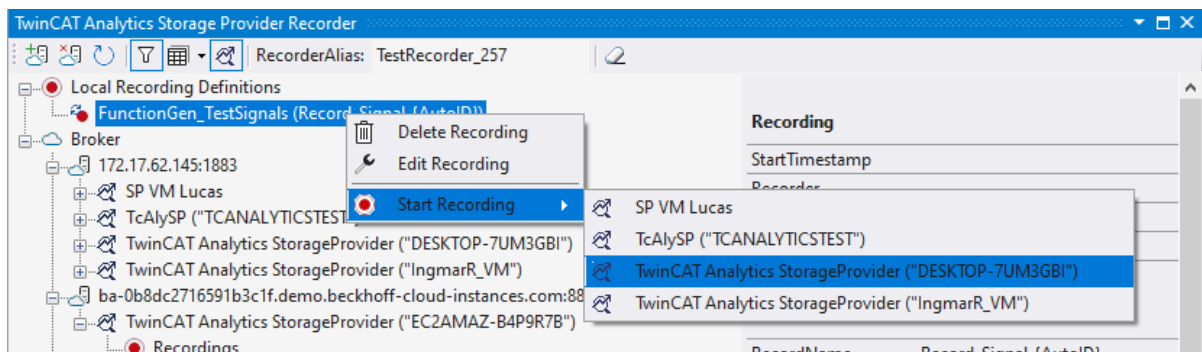


Sie können auch Aufzeichnungsnamen und eine Dauer konfigurieren (anderenfalls läuft die Aufzeichnung endlos, bis sie manuell gestoppt wird). Ein Ringpuffer kann nach Storage oder Zeit eingestellt werden.

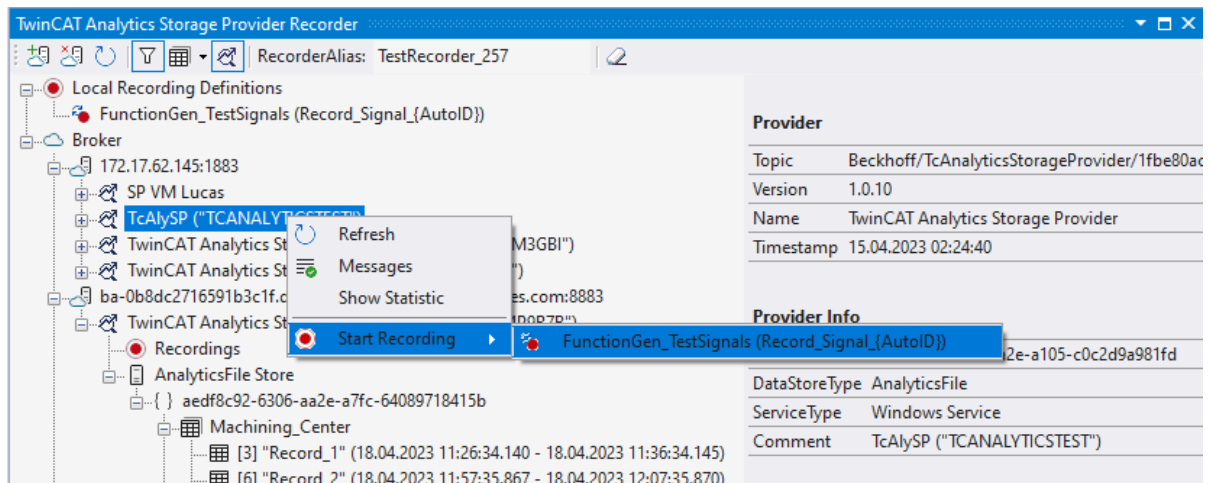
Aufnahme starten

Es gibt drei verschiedene Wege, um eine Aufnahme zu starten.

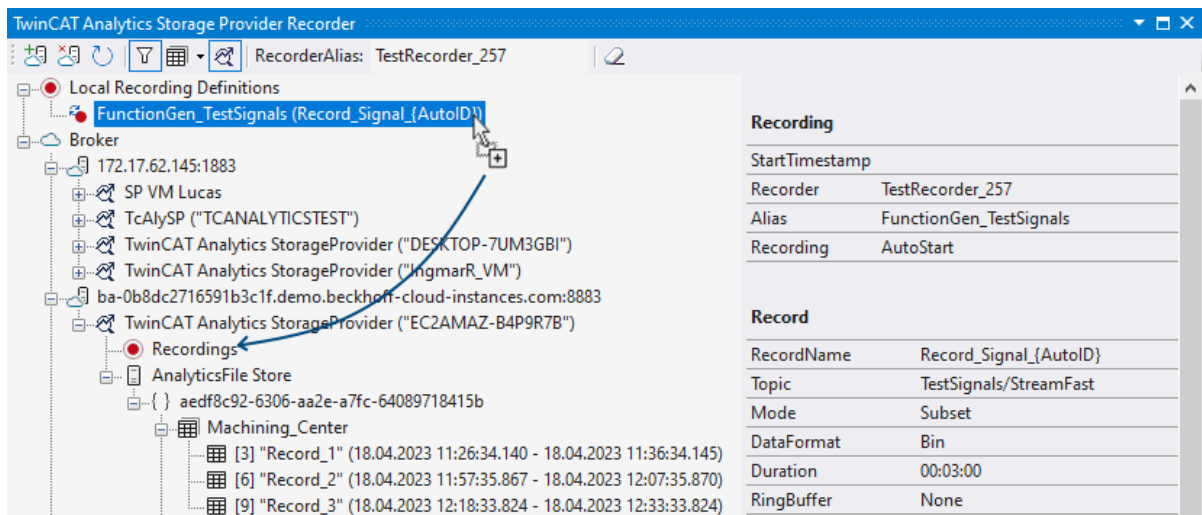
1. Über das Kontextmenü an einem lokalem Aufnahmedefinition Knoten. Hier kann der gewünschte Analytics Storage Provider ausgewählt werden, welcher die Daten aufzeichnen soll. Es werden nur die Storage Provider angezeigt, die auch Zugriff auf das ausgewählte Topic haben.



- Über das Kontextmenü am Analytics Storage Provider Knoten. Hier kann die gewünschte Aufnahme­definition ausgewählt werden. Es werden nur die Definitionen angezeigt, die auch vom Analytics Storage Provider verarbeitet werden können.

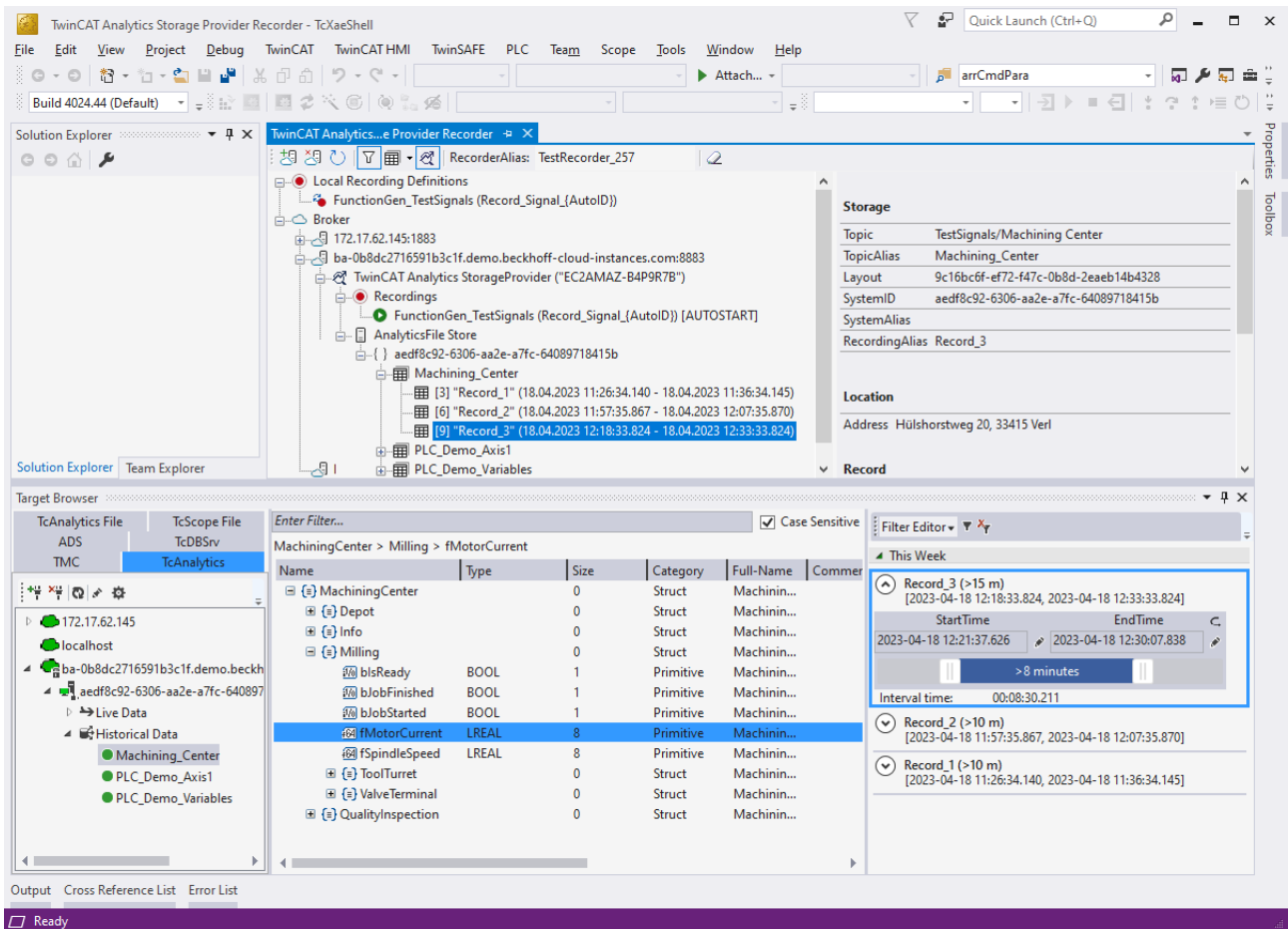


- Per Drag-and-Drop des lokalen Aufnahme­definition Knoten. Die gewünschte Aufnahme­definition kann auf den gewünschten Analytics Storage Provider per Drag-and-Drop gestartet werden. Sollte der Storage Provider die Aufnahme­definition nicht ausführen können, wird eine Fehlermeldung in der Fehlerliste ausgegeben.



Historisierte Daten verwenden

Nach und auch während der Aufzeichnung können Sie die historischen Daten als Eingang für Ihre Analyse im Target Browser auswählen. Im Target Browser finden Sie für die historischen Daten eine neue Steuerung auf der rechten Seite. Dort können Sie die Zeitspanne für Ihre Daten auswählen.

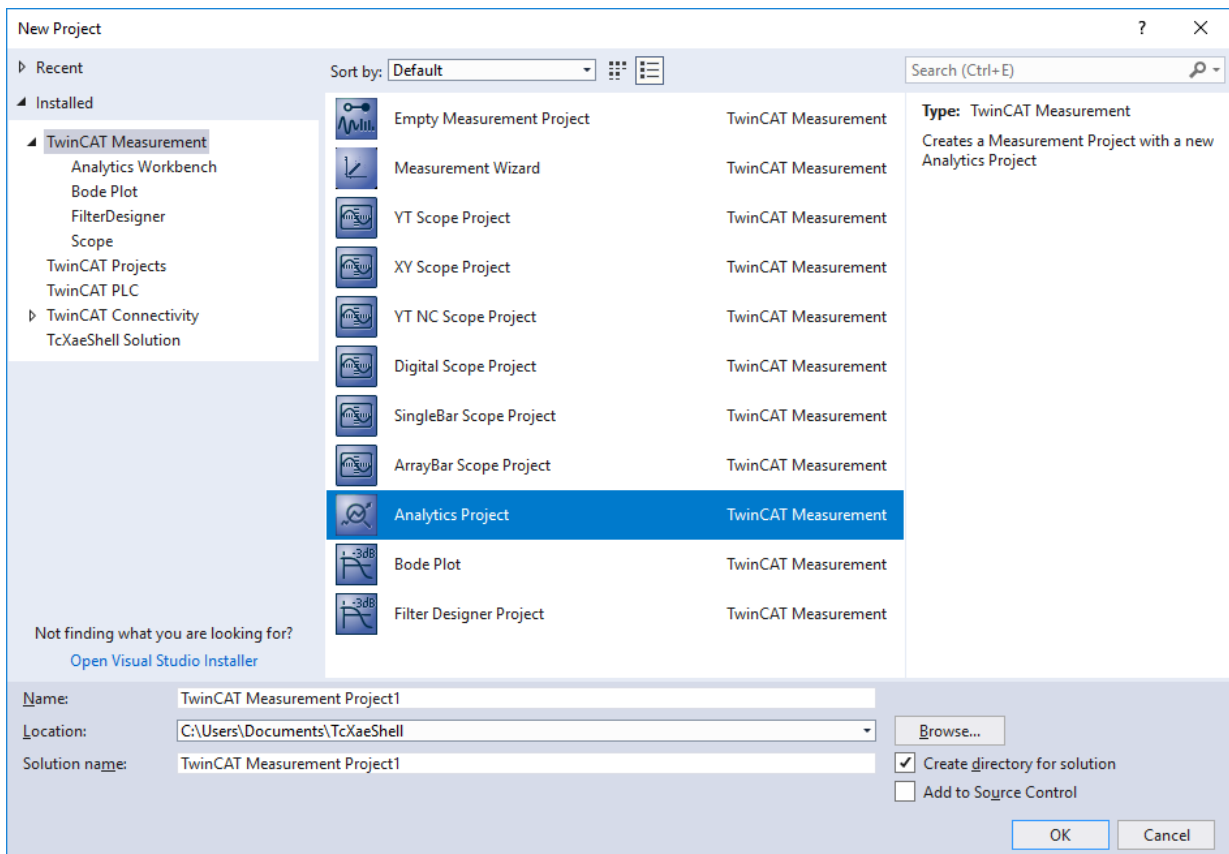


4.4 Analyse der Daten

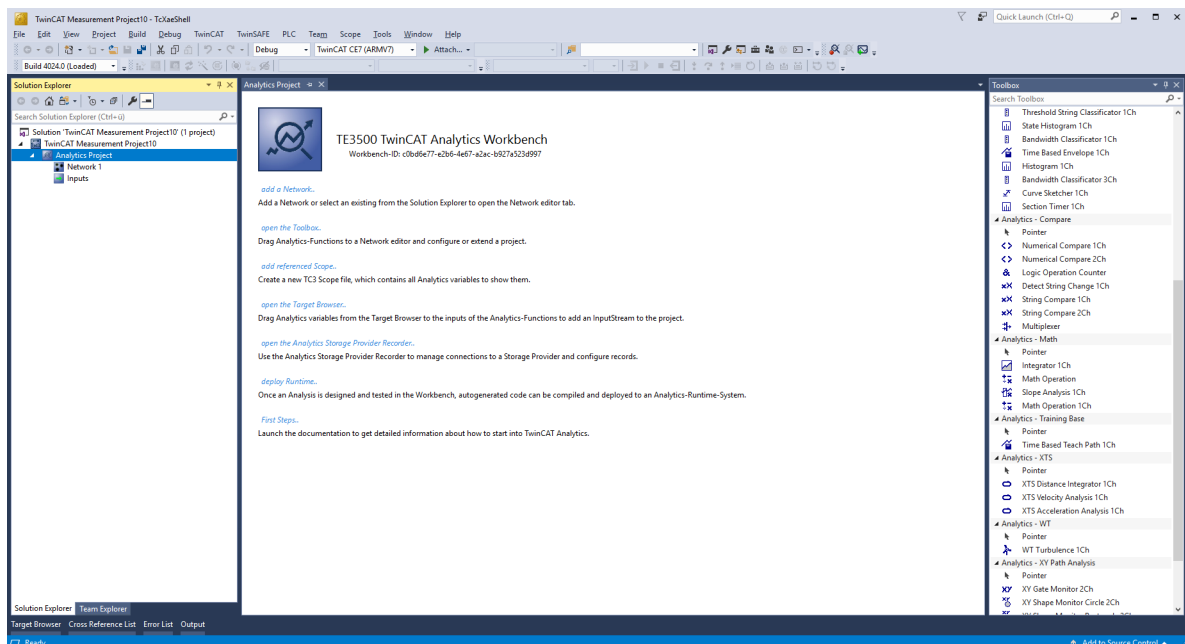
✓ Öffnen Sie Ihre TwinCAT Engineering-Umgebung, um die Analyse der Daten zu starten.

1. Öffnen Sie **Visual Studio® > File > New > Project...**

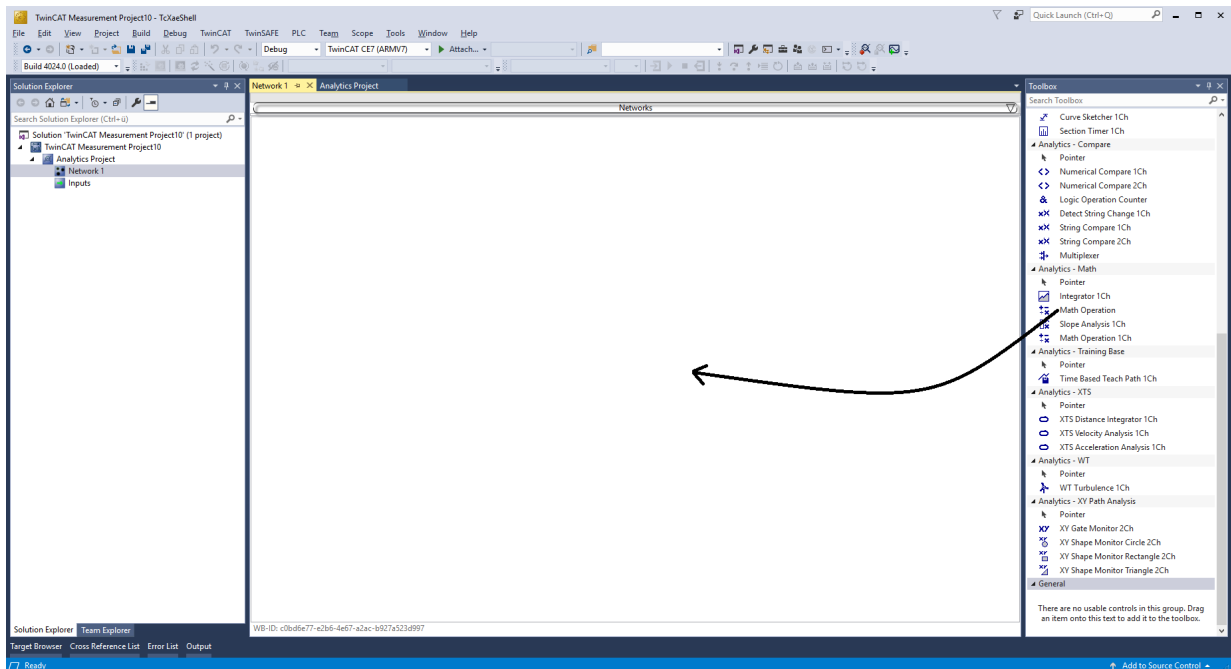
2. Wählen Sie aus **TwinCAT Measurement** die **Analytics-Projektvorlage** aus.



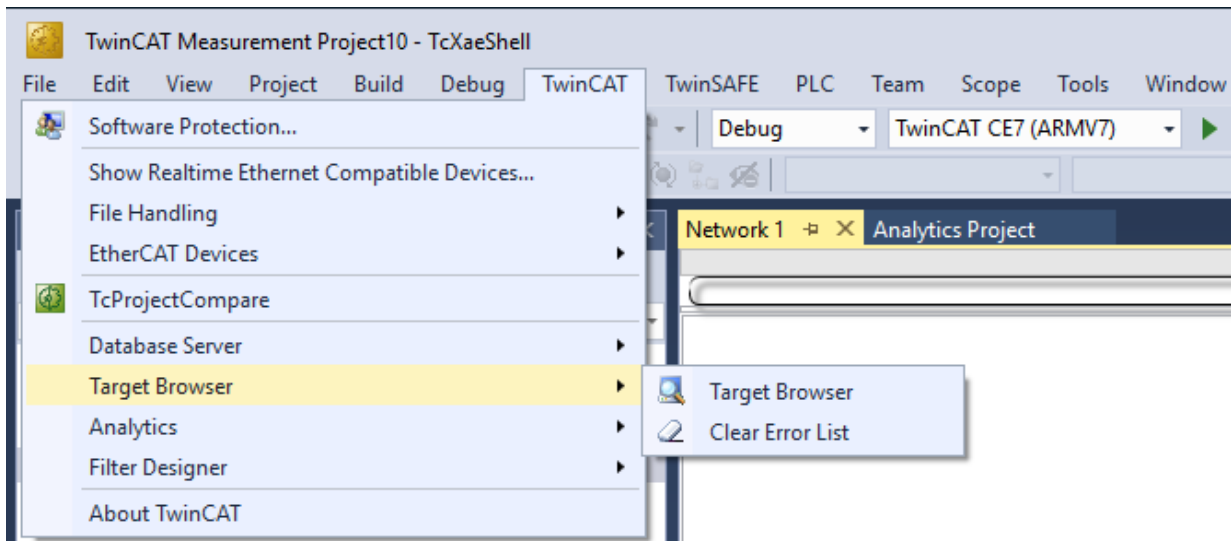
⇒ Das neue Projekt wird im Projektmappen-Explorer angezeigt. Nach einem Klick auf das Baumknotenenelement **Analytics Project** öffnet sich ein Startfenster, in dem Sie Ihre erste Aktion auswählen können. Von hier aus können Sie ein Netzwerk hinzufügen, die **Toolbox** öffnen, den **Target Browser** öffnen oder den **Analytics Storage Provider Recorder** öffnen. In den folgenden Schritten werden Sie all diese Aktionen durchführen.



- Es ist sinnvoll, als Erstes die **Toolbox** von Visual Studio® zu öffnen. Dort finden Sie alle unterstützten Algorithmen von TwinCAT Analytics. Die Algorithmen müssen in Netzwerke gruppiert und organisiert werden. Rechtsklicken Sie auf **Analytics Project**, um ein neues Netzwerk hinzuzufügen, oder fügen Sie mit Hilfe der Startseite ein Netzwerk hinzu. Das erste Netzwerk wird immer standardmäßig generiert.

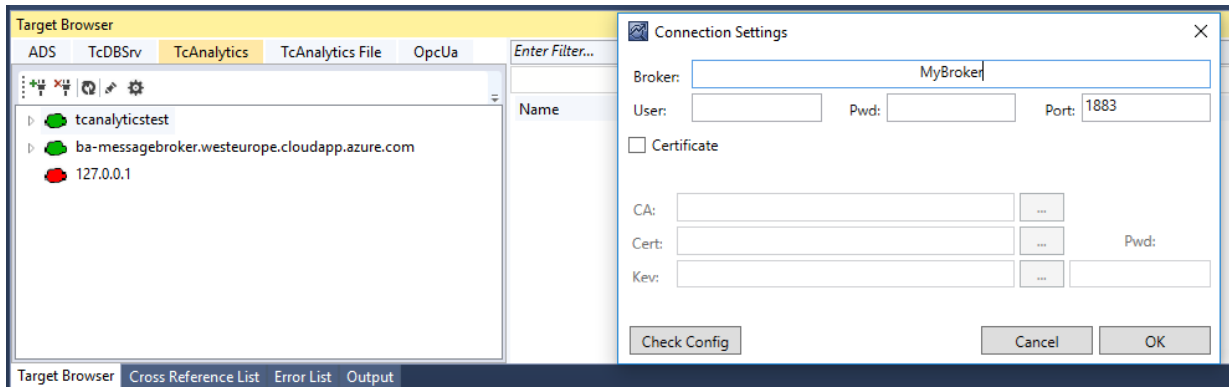


- Wenn Sie auf das Netzwerk klicken, wird ein Editor geöffnet. Nun können Sie den gewünschten Algorithmus per Drag-and-Drop in die Editor-Oberfläche ziehen.
- Nach der Auswahl des Algorithmus müssen Sie Eingangsvariablen mit den Modulen (Algorithmus) verbinden. Öffnen Sie dazu den **Target Browser**.
TwinCAT > Target Browser > Target Browser



- Wählen Sie nun den Karteireiter **TcAnalytics** oder **TcAnalyticsFile** im Target Browser aus. Mit dem Karteireiter **TcAnalytics** (MQTT) geht es weiter.

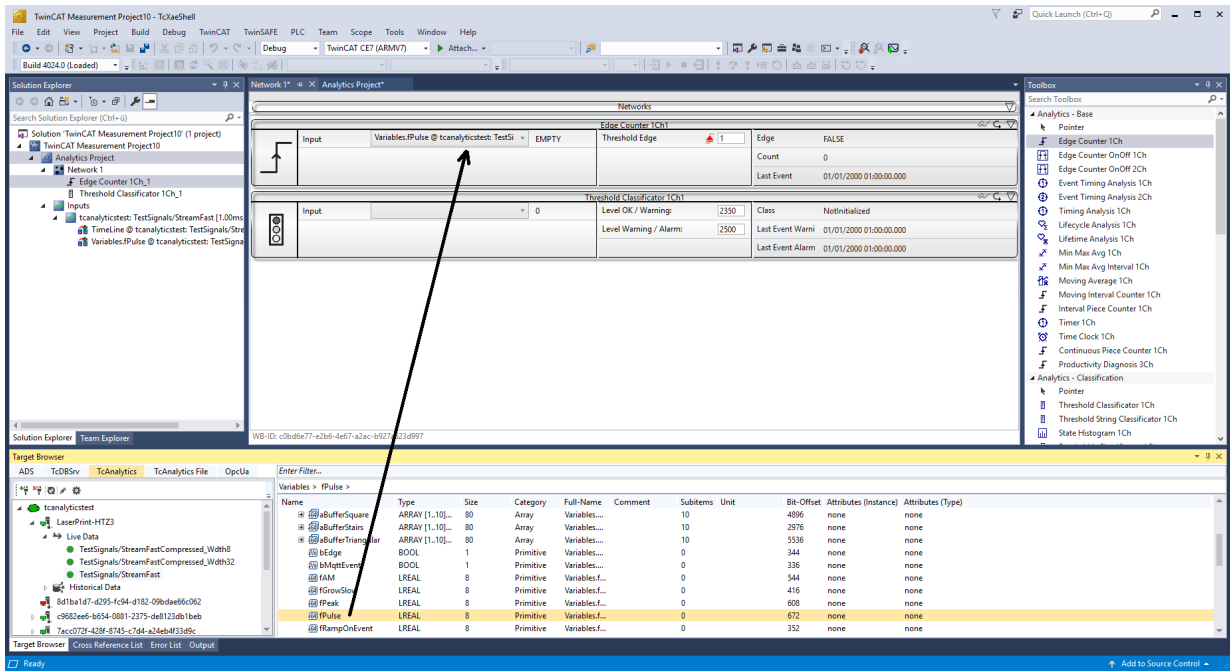
7. Klicken Sie auf das grün markierte Icon in der Symbolleiste dieser Analytics-Erweiterung. Es öffnet sich ein Fenster, in dem Sie die Konnektivitätsdaten Ihres Message-Brokers angeben können.



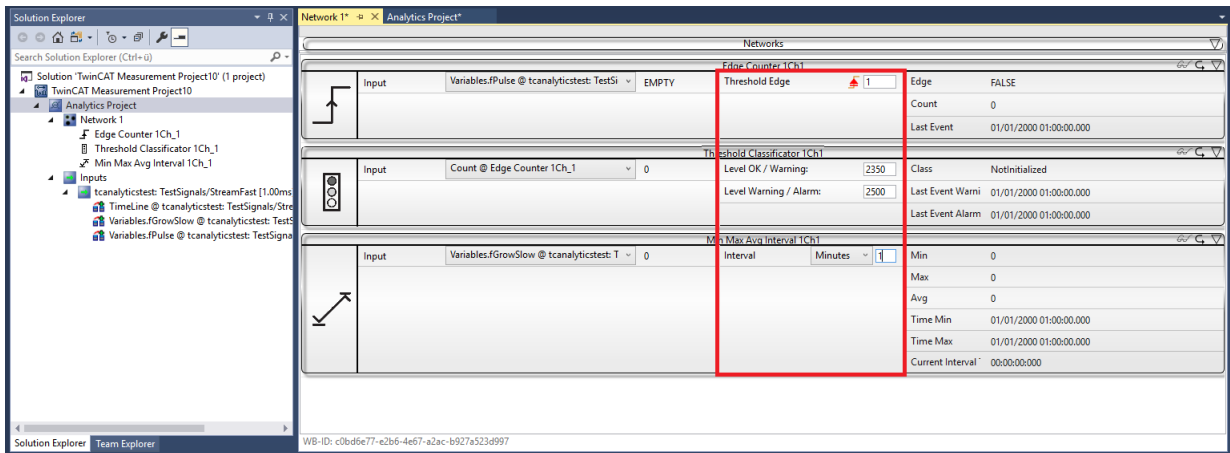
8. Wählen Sie Ihren MQTT-Analytics-Client aus (TwinCAT Analytics Logger, TwinCAT IoT Data Agent oder Beckhoff EK9160). Für jede Steuerung gibt es eine eindeutige ID. Diese ID wird im Target Browser angezeigt.
9. Über einen Klick auf das **Zahnrad-Icon**, gelangen Sie zur Seite Machine Administration. Hier können Sie einen System Alias-Namen vergeben, der im Target Browser anstelle der ID angezeigt wird.

Source	Customer	SystemID	System Alias	Online	Position	Topic Alias	Description
172.17.62.145							
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True	Laserstr. 13, 40597 Düsseldorf		MP4 Steel Laser Data
TwinCAT Analytics Logger	MA Laser	3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Logger	Fertig Motors	8d1ba1d7-d295-fc94-d182-09bdae66c062		False			
TwinCAT Analytics Logger	Fertig Motors	8d1ba1d7-d295-fc94-d182-09bdae66c062		False			
TwinCAT Analytics PLC DataLogger		7acc072f-428f-8745-c7d4-a24eb4f33d9c		False			
TwinCAT Analytics TEST Logger	Beckhoff	91c6eab3-1abb-5593-3651-1426874c91f		True	Hülshorstweg 20, 33415 Verl		
TwinCAT Analytics Storage Provider		c9682ee6-b654-0881-2375-de8123db1beb		True		RetroFit	
TwinCAT Analytics Storage Provider		7acc072f-428f-8745-c7d4-a24eb4f33d9c		True		AnalyticsSolution Results 1	
TwinCAT Analytics Storage Provider		7fb4f250-c130-7d7e-0a26-71ed8cee9340		True		CloudControl	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True			
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		TestSignals whole Image	
TwinCAT Analytics Storage Provider		c5eefcfd-4f14-5f45-dce4-7524715a9ae3		True		DataAgent Test	
TwinCAT Analytics Storage Provider		3f8a342a-6fac-3e76-6172-e7b5f62c0eb0		True		BigI40 FavValues	
TwinCAT Analytics Storage Provider		a313c550-7537-0617-827d-c6930e90d931		True		EK Test2	
TwinCAT Analytics Storage Provider		d180dde-afea-78d2-9ac1-65101d008687		True		NewMachineApp	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		LongTerm	
TwinCAT Analytics Storage Provider		3db95703-29fb-d99e-eb13-017b54677bb0	LaserPrintZG15	True		MyFavoriteData	
TwinCAT Analytics Storage Provider		56cfbec6-3ab5-c1cc-1a1d-e6f4da86adf0		True		EdgeComputingTc2	
TwinCAT Analytics Storage Provider		56cfbec6-3ab5-c1cc-1a1d-e6f4da86adf0		True		EdgeComputingTc3	

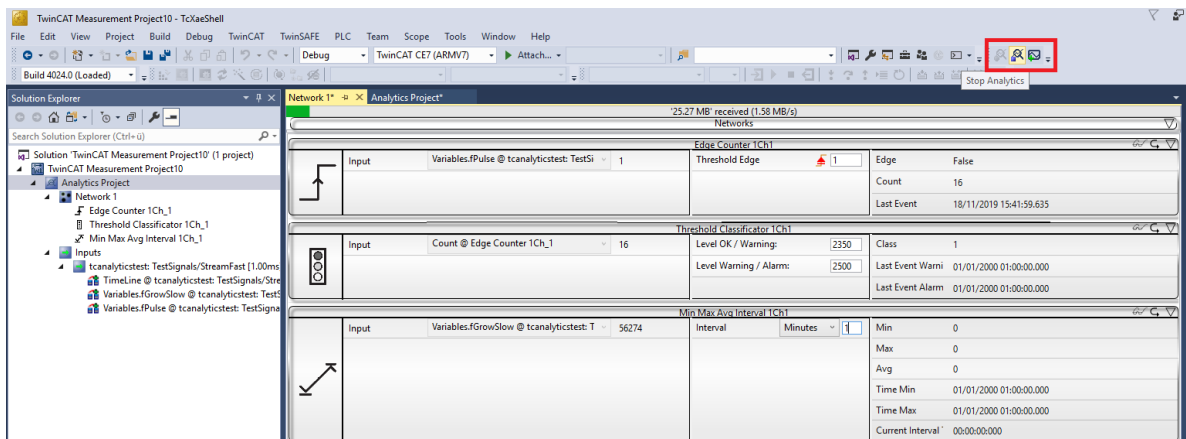
10. Im nächsten Schritt können Sie zwischen Live-Daten und historischen Daten für jeden MQTT Analytics-Client wählen. Die historischen Daten werden in diesem Fall von dem TwinCAT Analytics Storage Provider bereitgestellt.



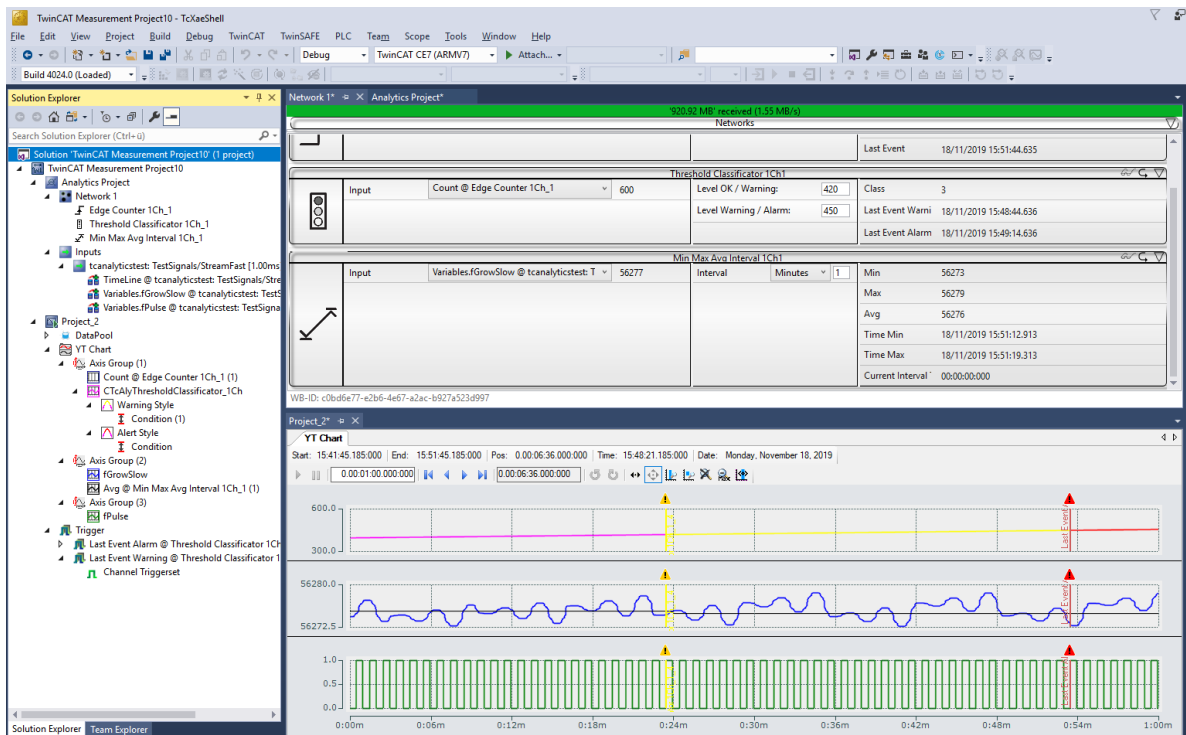
11. Sie können die Variablen per Drag-and-Drop in die Eingänge des spezifischen Algorithmus ziehen. Bei den meisten Algorithmen können Bedingungen wie Schwellen, Zeitintervalle, logische Operatoren usw. festgelegt werden. Diese Einstellungen werden in der Mitte jedes Moduls vorgenommen.



⇒ Schließlich ist Ihr erstes Analytics Project fertiggestellt. Um die Analyse zu starten, klicken Sie auf **Start Analytics**. Um die Analyse zu stoppen, klicken Sie auf **Stop Analytics**.



- ⇒ Vor dem Start der Analyse oder auch während der Laufzeit können Sie auf die Schaltfläche **Add Reference Scope** klicken. Dadurch wird automatisch eine zu Ihrem Analytics-Projekt passende Scope-Konfiguration erstellt.

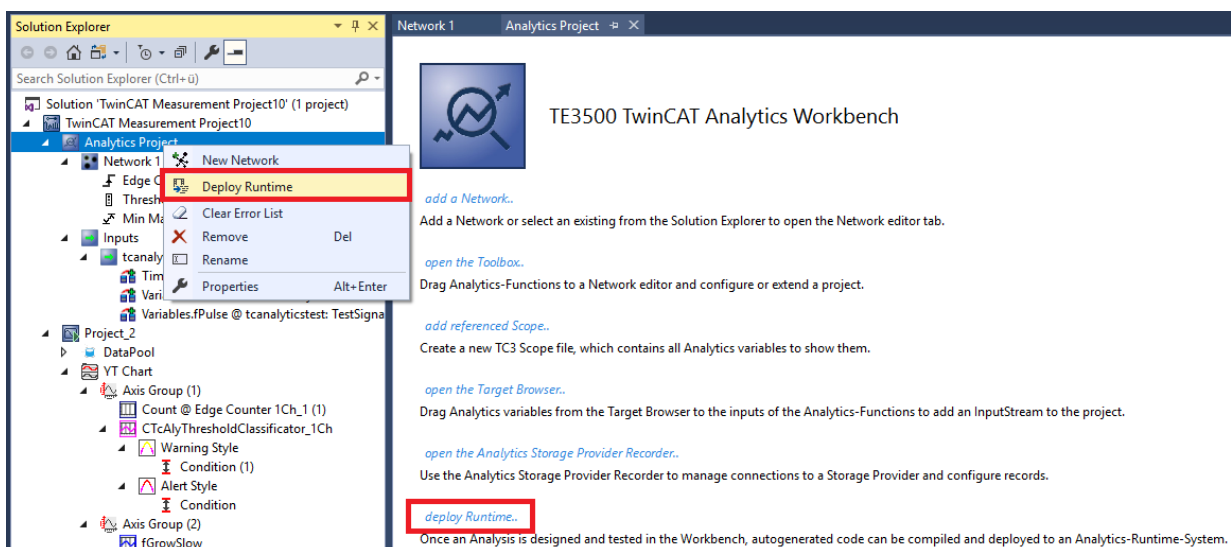


- ⇒ Die Ergebnisse der Analyse können in den Scope View-Grafiken per Drag-and-Drop angezeigt werden. Beispielsweise kann ein Mittelwert als neuer Kanal in der Ansicht angezeigt werden. Zeitstempel als Marker auf den X-Achsen zeigen signifikante Werte.

4.5 24-stündige Anwendung von Analytics

Der letzte große Schritt im Workflow von TwinCAT Analytics ist die kontinuierliche 24-stündige Maschinenanalyse. Sie läuft parallel zu den Maschinenanwendungen im Feld. Um dies ganz einfach zu gestalten, kann die TwinCAT Analytics Workbench automatisch einen SPS-Code und ein HTML5-basiertes Dashboard Ihrer Analytics-Konfiguration generieren. Beide können in eine TwinCAT Analytics Runtime (TC3 PLC und HMI Server) heruntergeladen werden und bieten dieselben Analyseergebnisse wie das Konfigurator-Tool in der Engineering-Umgebung.

- ✓ Speichern Sie zunächst Ihre Konfiguration und öffnen Sie den Analytics Deploy Runtime-Assistenten. Dies kann über das Kontextmenü im Analytics Project-Tree Item oder über die Startseite erfolgen.



1. Wenn der Assistent geöffnet ist, können Sie sich durch einige Karteireiter klicken. Der erste heißt Solution. Hier können Sie entscheiden, wie Ihr Analytics-Projekt im SPS-Code verwendet werden soll:
Als...
vollständig neue Lösung.
Teil einer vorhandenen Lösung.
Aktualisierung einer vorhandenen Analytics-Lösung.

Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

Solution | TwinCAT PLC Target | Results | HMI Dashboard | Visual Studio | Summary

Create new Solution

Solution Path: C:\temp\Analytics\Test

Solution Name: Production

Project Name: MachineAnalysis

Add to existing Solution

Solution Path:

Project Name: AnalyticsProject

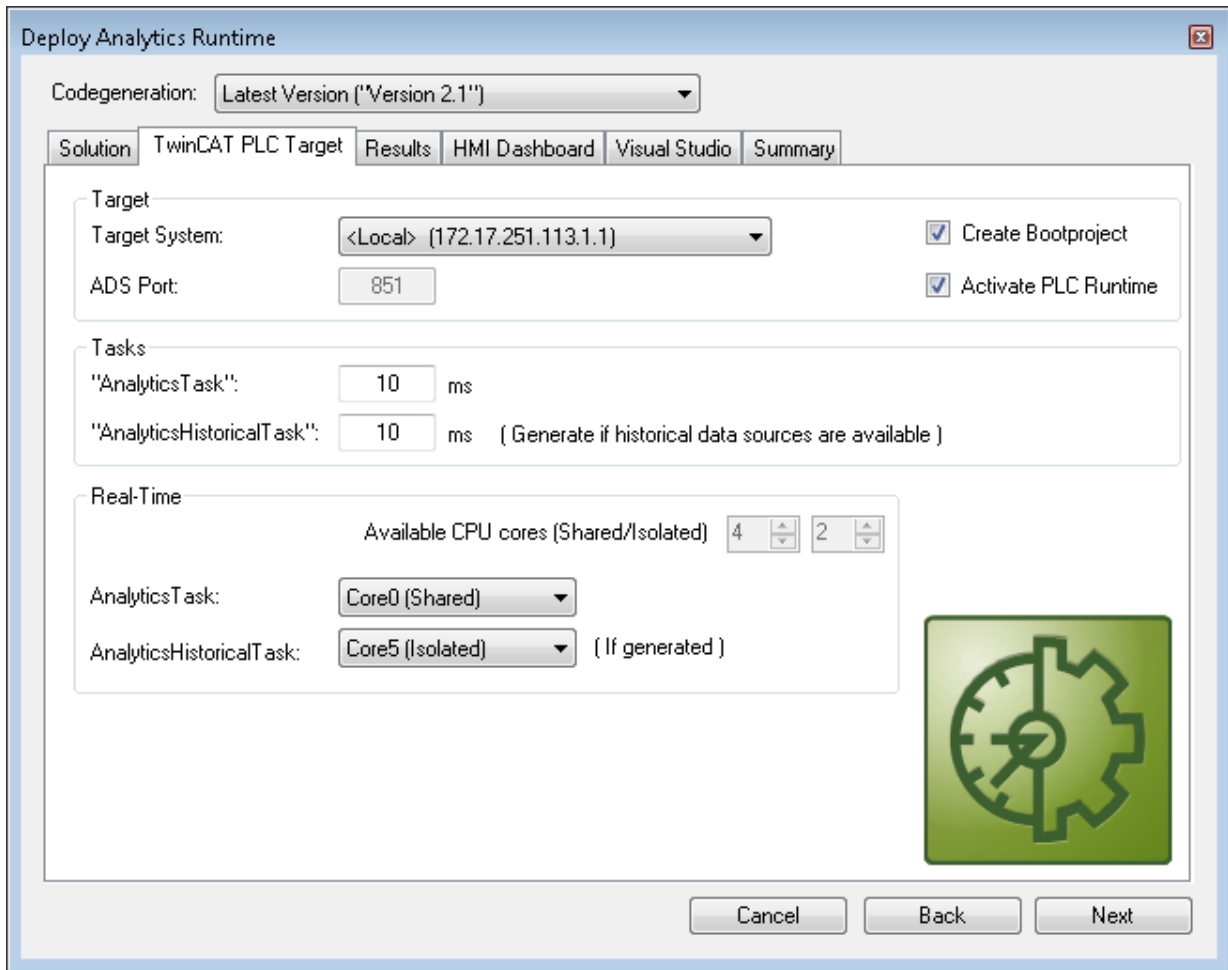
Merge to existing Project (TwinCAT Project Compare)

Solution Path:

Project Name:

Cancel Back Next

2. Im Karteireiter **TwinCAT PLC Target** können Sie das ADS-Zielsystem wählen, das die TwinCAT Analytics Runtime (TF3550) ausführt. Das erzeugte Projekt ist sofort ausführbar. Hierzu können Sie die Option **Activate PLC Runtime** setzen. Zudem kann ausgewählt werden, dass direkt ein Bootprojekt erstellt wird.



3. Insbesondere bei virtuellen Maschinen ist es wichtig, das Projekt auf isolierten Kernen auszuführen, was ebenfalls eine Option in diesem Karteireiter ist. Der nächste Karteireiter **Results** wird nur benötigt, wenn Sie in den Algorithmeigenschaften die Option **Stream Results** ausgewählt haben. Wenn Sie Ergebnisse senden möchten, können Sie hier entscheiden, in welcher Weise (lokal in einer Datei/durch MQTT) und welchem Format (binär/JSON) dies geschehen soll. Auch dies wird automatisch generiert und umgehend nach Aktivierung ausgeführt.

Deploy Analytics Runtime

Codegeneration: Latest Version ("Version 2.1")

Solution | TwinCAT PLC Target | **Results** | HMI Dashboard | Visual Studio | Summary

Create no Results

Stream Results to MQTT Broker

Topic: Analytics/Analysis/ResultStream

MQTT Connection Settings Json Format

Write Results to Analytics File

File Path: ...

Max File Size: 256 Sample buffer count

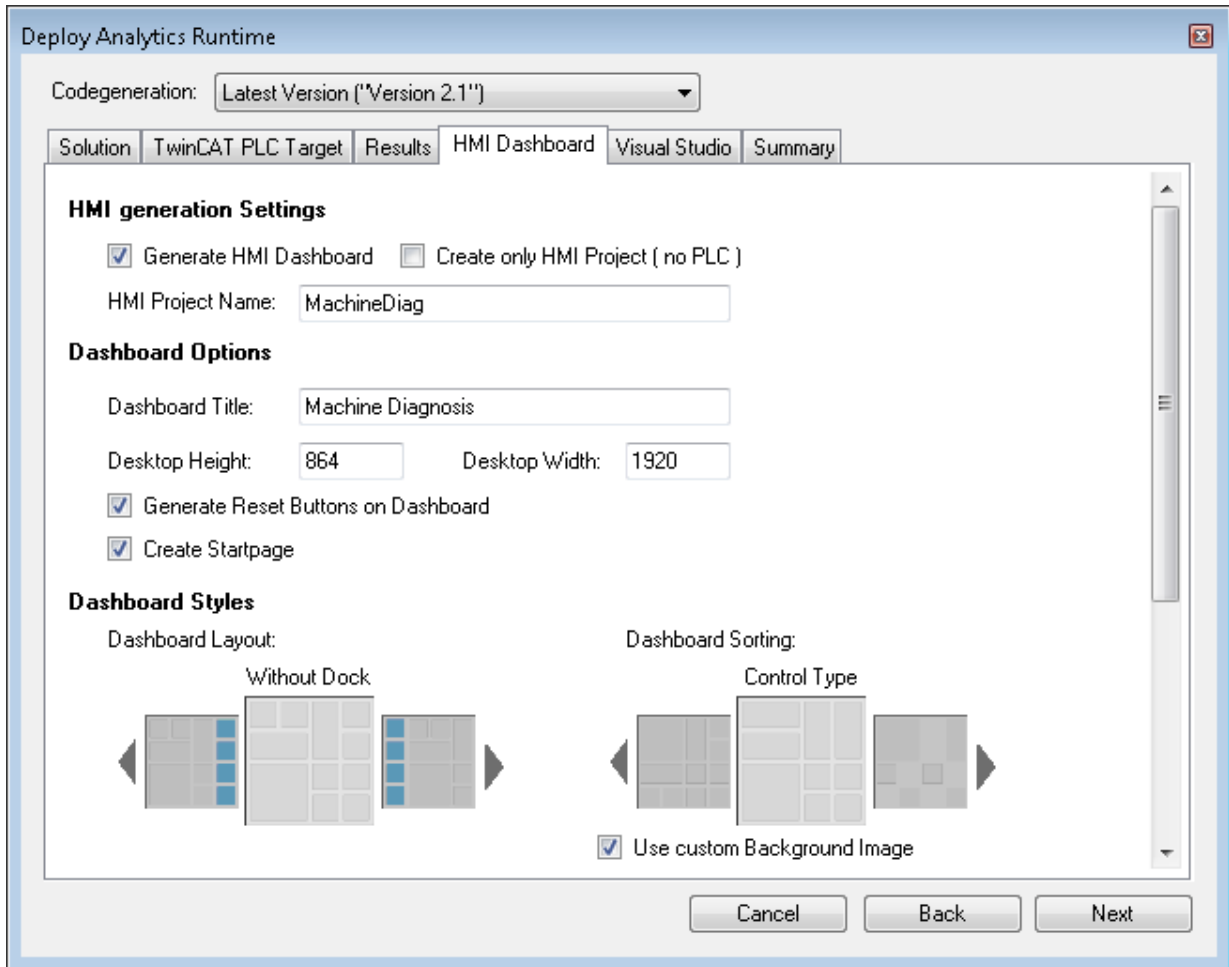
Select Result Items

CycleTime: User specified cycle time 5000 ms

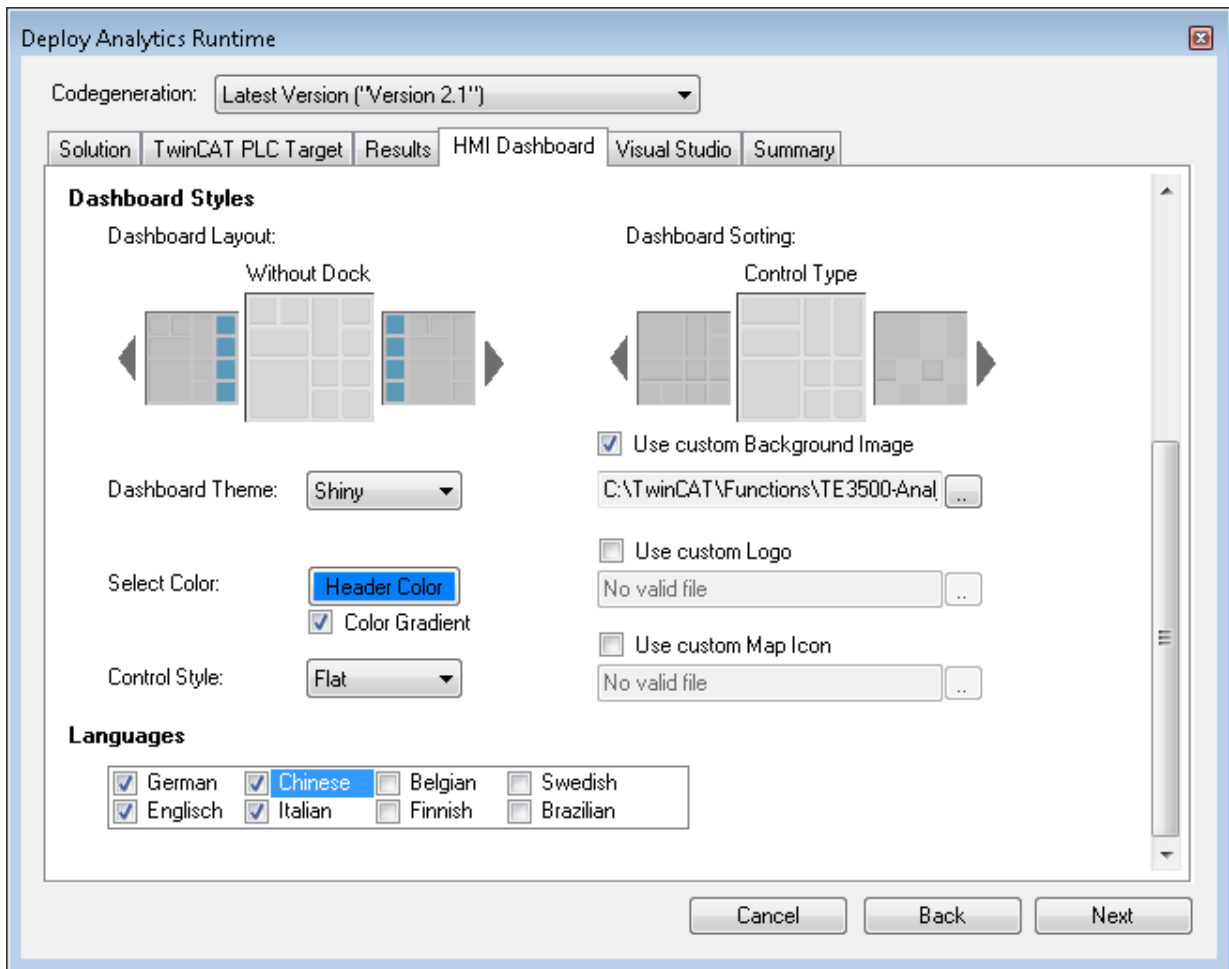
Cancel Back Next

Ein Downsampling der Ergebnisse ist durch die Festlegung einer Zykluszeit möglich. Der nächste Karteneiter ist für das **HMI-Dashboard** vorgesehen. Voraussetzung für die automatische Generierung

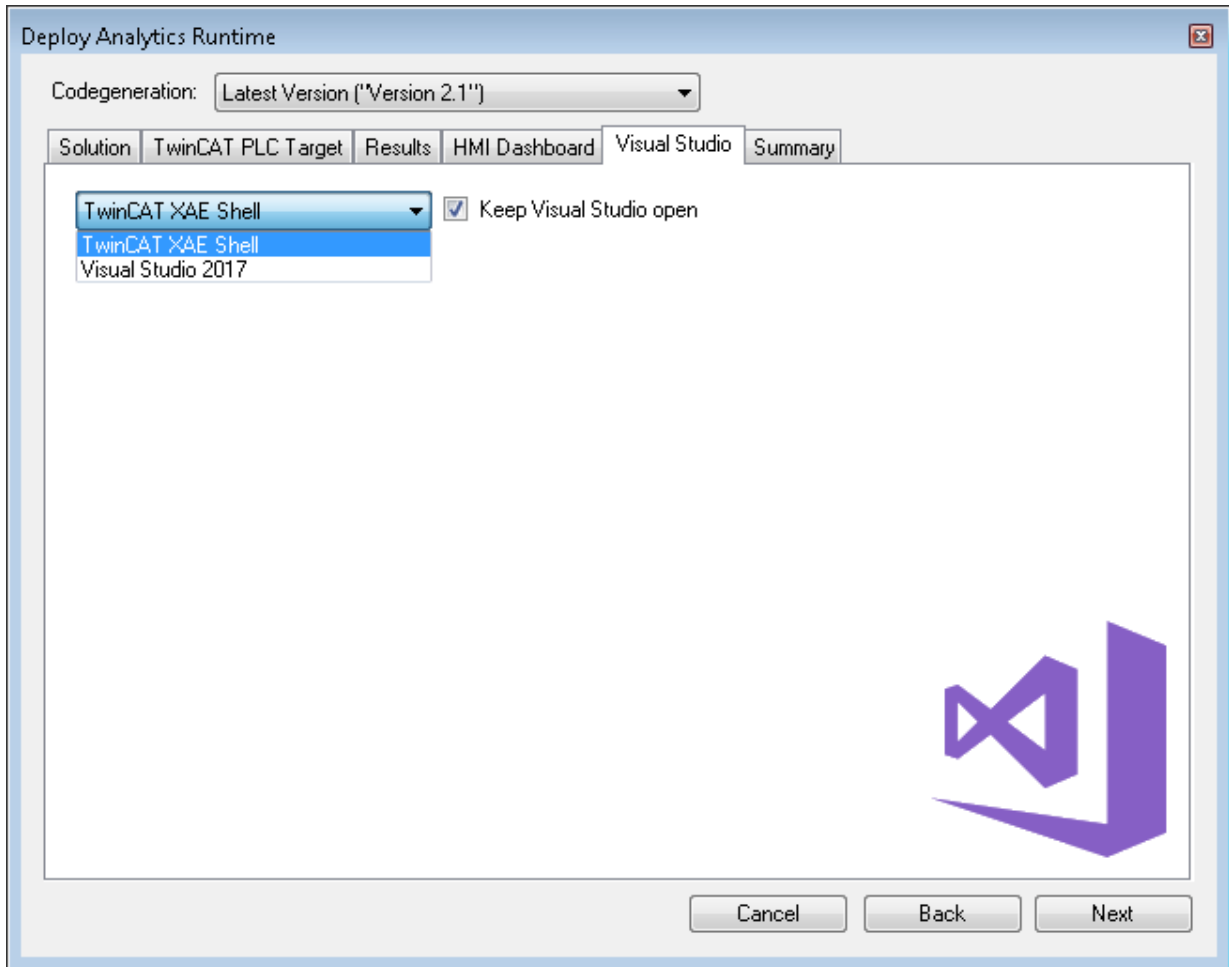
des Dashboards ist die Auswahl von HMI Controls für die entsprechenden Algorithmen, deren Ergebnisse im Dashboard angezeigt werden sollen.



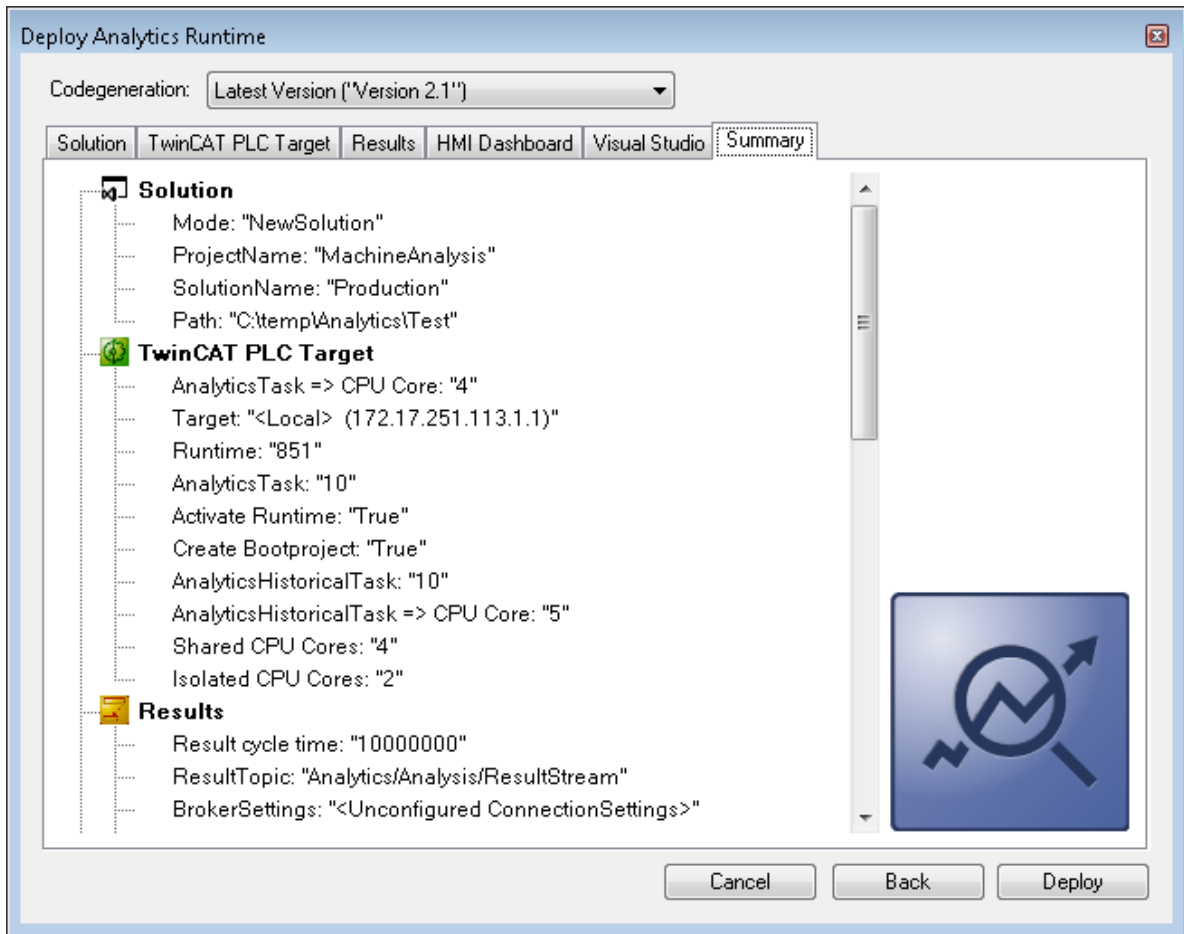
- Sie können verschiedene Optionen für Ihr Analytics Dashboard auswählen, wie eine Startseite mit einer Karte, Layouts, Sortieralgorithmen, eigene Farben und Logos. Wenn Sie mehrere Sprachen für die Analytics Controls auswählen, wird auch ein Menü zur Sprachumschaltung generiert.



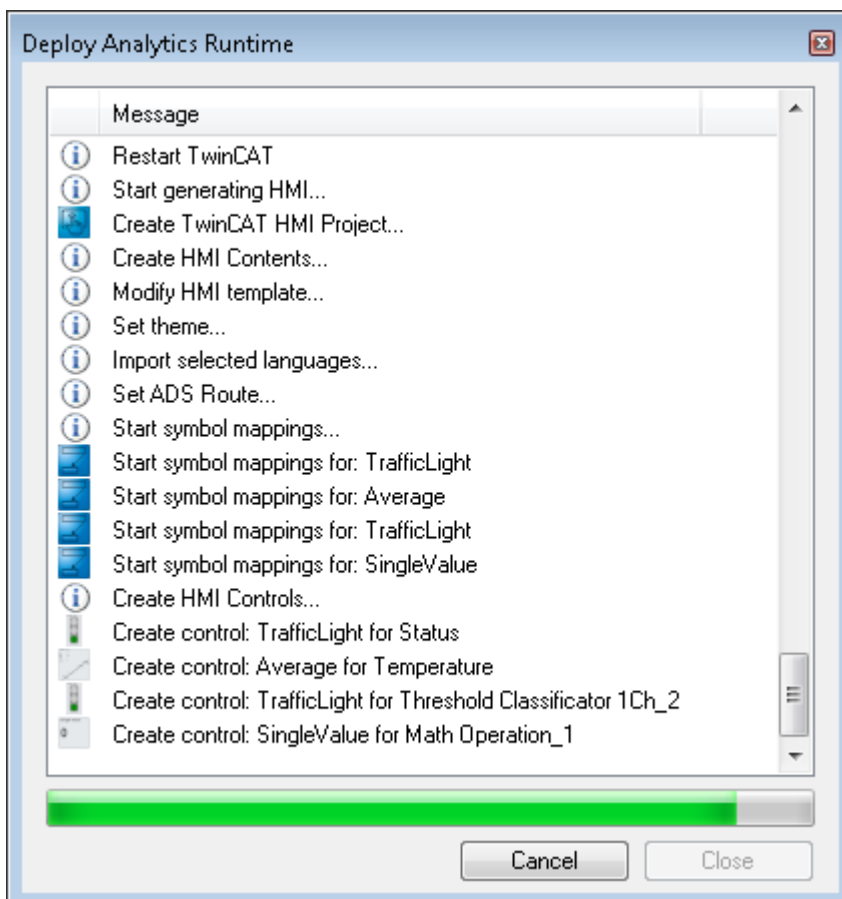
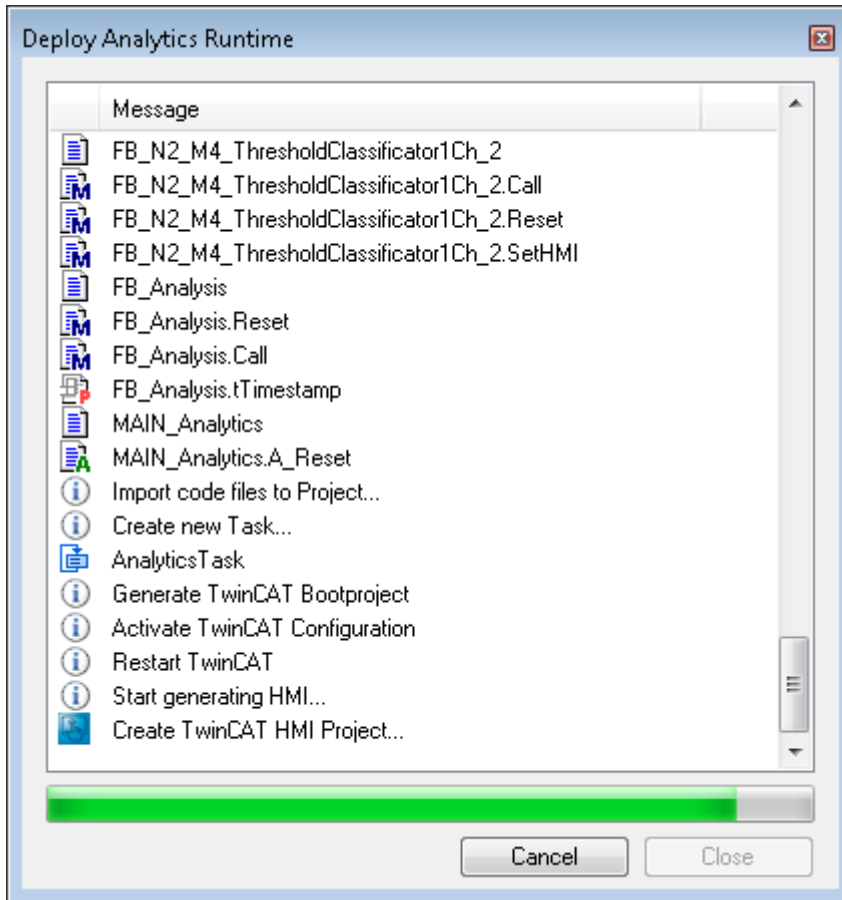
5. Wählen Sie eine der installierten Versionen von Visual Studio® aus, und, ob die Instanz sichtbar starten soll oder ob sie nur eingerichtet und im Hintergrund aktiviert werden soll.



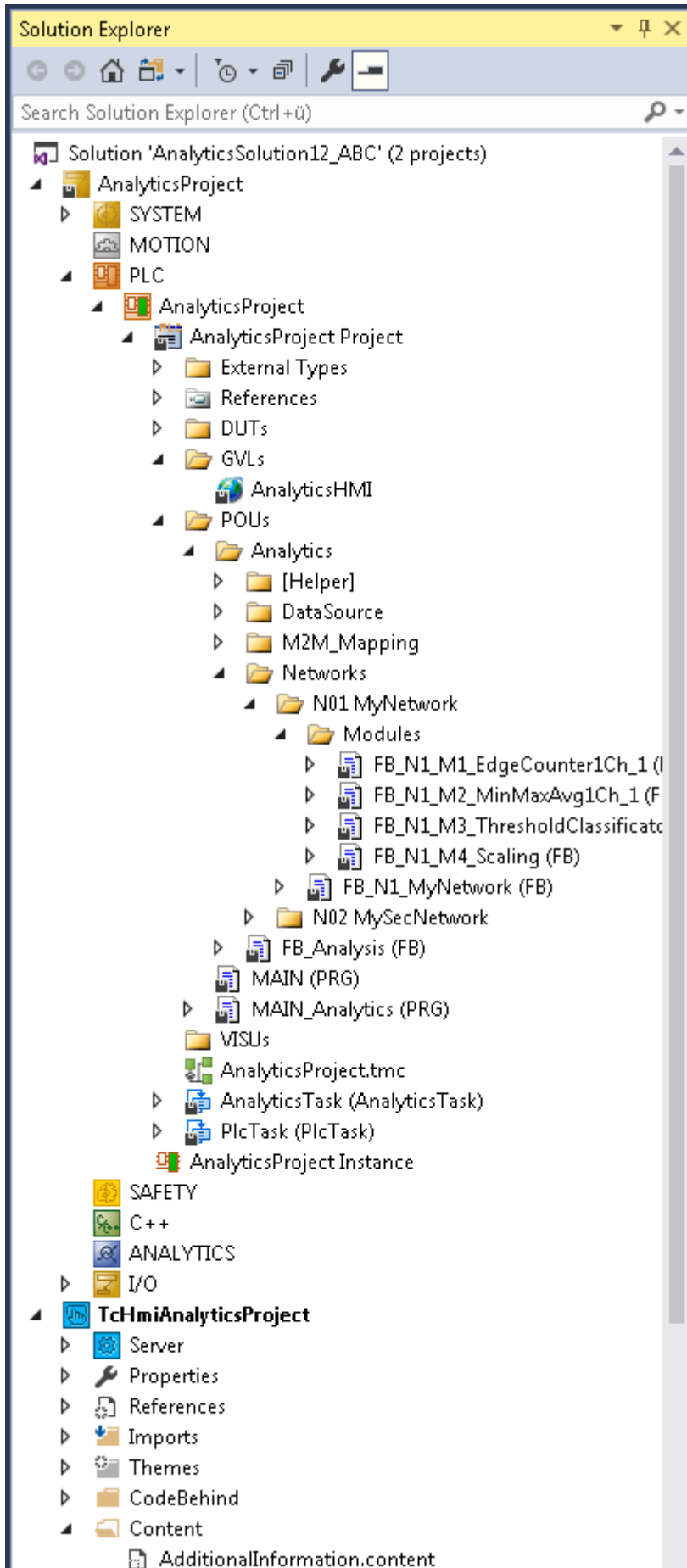
⇒ Zuletzt finden Sie eine Übersicht.



- Nun können Sie auf die Schaltfläche **Deploy** klicken, um den Generierungsprozess zu starten. Das SPS-Projekt und das HMI-Dashboard werden nun generiert.



⇒ Nach der Meldung „Deploy Runtime succeeded“ finden Sie eine neue Visual Studio®/XAE Shell-Instanz auf Ihrem Desktop. Die neue Projektmappe und beide Projekte werden erstellt.



5 Technische Einführung

5.1 Grundkonzepte

Variablen und Datentypen

Es können mehrere Arten von Variablen geloggt werden. Die Variablen, sind Teil von:

- SPS- oder NC-Prozessabbildern
- SPS-Programmen
- Prozessabbildern von Geräten, z. B. einem EtherCAT-Master und
- Datenbereiche von generischen TcCom-Objekten

Darüber hinaus können sie von jedem Datentyp sein, der in IEC 61131 oder dem C++-Standard im Fall von generischen TcCom-Objekten definiert ist.

Strukturierte Datentypen können rekursiv andere strukturierte Datentypen enthalten und können ganz oder teilweise geloggt werden. Weitere Informationen zu diesem Thema finden Sie im Abschnitt „Konfiguration“.

Betriebsarten

In dieser Dokumentation werden die Terminologie und Konzepte verwendet, die im Abschnitt MQTT beschrieben sind. Für allgemeine Informationen über das Protokoll verweisen wir auf diesen Abschnitt.

Ein wichtiges Konzept von TwinCAT Analytics sind Streams, die die Basiseinheit der Transaktion zwischen einer Datenquelle und einem Ziel bilden. Ein Daten-Logger kann mehrere Streams steuern.

Ein Stream kann vier Komponenten umfassen:

- Streambeschreibung
- Stream-Tx-Beschreibung
- Symbolinfo
- und Streamdaten.

Im **MQTT-Modus** koordinieren die Analytics-Teilnehmer mittels der ersten drei Komponenten unter Verwendung eines MQTT-Brokers und eines spezifischen MQTT-Topic für jede Komponente. Die **Topics** sind im Allgemeinen wie in der folgenden Tabelle veranschaulicht strukturiert, wobei <>-Klammern auf Variablen hinweisen, gegenüber den anderen Teilen, die fest sind.

Komponente	Topic-Struktur	Format	Zweck
Beschreibung	<MainTopic>/<StreamTopic>/Beschr	JSON	Informiert darüber, ob eine Streamquelle online oder offline ist. Umfasst einen Zeitstempel der Information.
Tx-Beschreibung	<MainTopic>/<StreamTopic>/Bin/Tx/Beschr	JSON	Informiert über die Übertragungsparameter, wenn eine Streamquelle aktiv Daten überträgt.
Symbolinfo	<MainTopic>/<StreamTopic>/Bin/Tx/Symbole	Binär	Enthält Metainformationen über die Variablen, d. h. ausschließlich des Istwerts.
Daten	<MainTopic>/<StreamTopic>/Bin/Tx/Daten	Binär	Enthält die reinen Variablenwerte.

Das Main-Topic ist das einzige Sub-Topic, das frei vom Benutzer gesetzt werden kann.

Ein Stream kann **gestartet** und **gestoppt** werden. Über die Systemmanager-Konfiguration kann ein Stream standardmäßig gestartet werden, sobald TwinCAT im Run-Modus startet. Außerdem können Streams aus dem SPS-Code gestartet und gestoppt werden.

Der Logger sendet, wenn er mit einem Broker verbunden ist, zuerst die Streambeschreibung, gefolgt von der Tx-Beschreibung und Symbolinfo, sobald der Stream startet. Dies ermöglicht es den Empfängern, alle notwendigen Maßnahmen zu ergreifen, bevor Daten eintreffen. Schließlich werden die Daten zyklisch gesendet.

Beim **Dateimodus** dient das TwinCAT-Bootverzeichnis auf dem Zielgerät als Basis für ein spezifisches Analytics-Verzeichnis, welches wiederum ein Unterverzeichnis pro Stream enthält, sobald das jeweilige TwinCAT-Projekt aktiviert worden ist. Eine spezifische .tas-Datei im Stream-Verzeichnis enthält die Symbolinfo, während .tay-Dateien, die zyklisch erstellt werden, die Streamdaten enthalten.

Beziehung von Logger und Streams

Ein Daten-Logger kann mehrere Streams steuern. Wie im Abschnitt Konfiguration ausführlicher beschrieben, kann ein Benutzer der Analytics-Konfiguration einen oder mehrere Daten-Logger hinzufügen. Streams werden dann automatisch hinzugefügt, abhängig davon, welche Variablen zum Loggen verfügbar sind. Um zu verstehen, wie Streams einem Logger zugeordnet werden, ist es hilfreich zu wissen, dass jeder Stream spezifische Eigenschaften hat. Eine Eigenschaft, die **Zykluszeit**, ergibt sich daraus, dass jede Variable, auf die zyklisch reagiert wird, einem zyklischen Task unterliegt; daher können die Zykluszeiten, denen verschiedene Variablen zugrunde liegen, variieren. Da ein Stream definitionsgemäß eine feste Zykluszeit hat und um Tasks mit derselben Zykluszeit zu entkoppeln, wird für jeden Task, der entsprechende Variablen steuert, ein Stream erzeugt. Zusätzlich ist es sinnvoll, Streams abhängig von der **Streamquelle**, d. h. der ursprünglichen SPS-Instanz oder dem TcCom-Objekt, weiter zu unterteilen. Dadurch können die Benutzer Variablen von verschiedenen Quellen an verschiedene MQTT-Topics senden und die Übertragung unabhängig starten/stoppen. Letztendlich läuft das Ganze auf folgendes Schema hinaus:



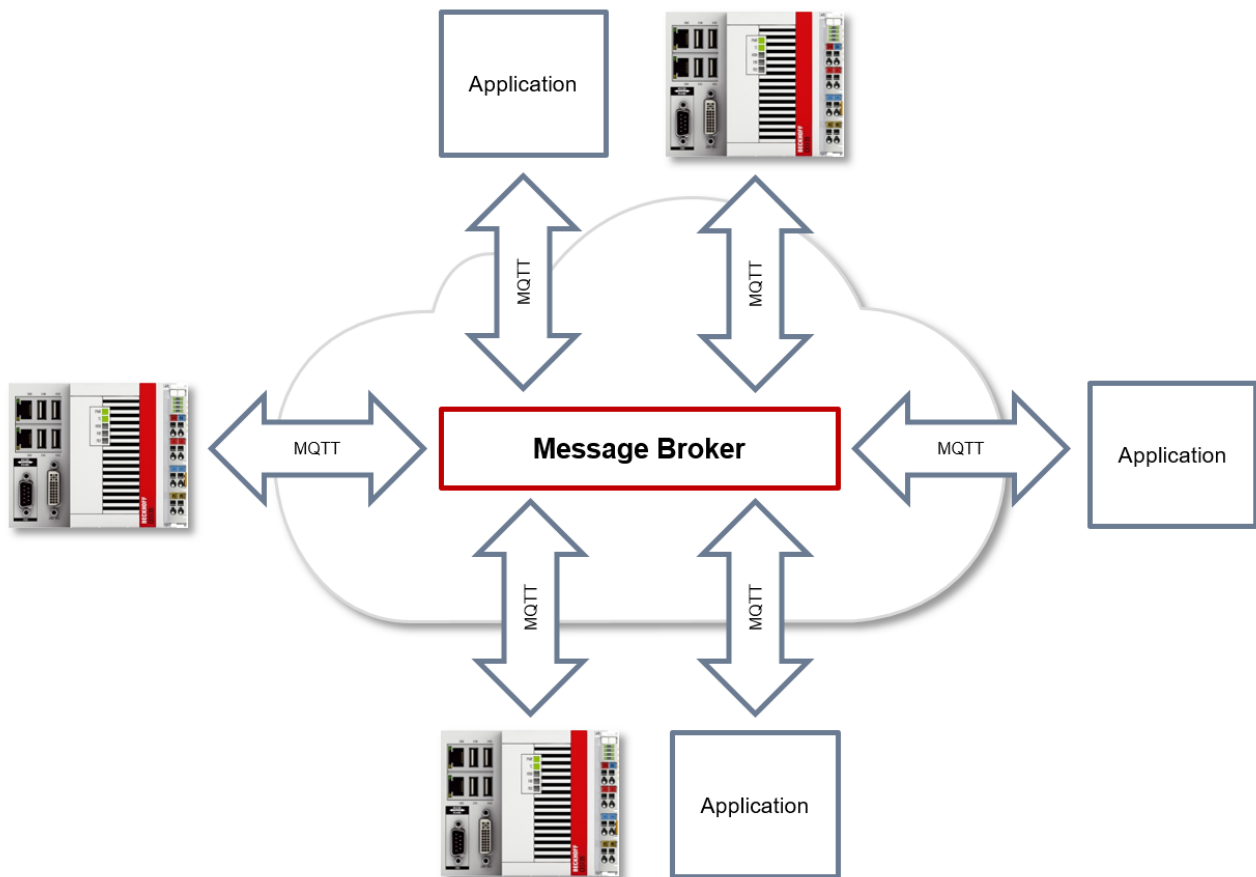
Für jede Streamquelle und jeden Task, der Variablen dieser Quelle steuert, wird ein Stream erzeugt.

Beim Konfigurieren des Daten-Loggers gibt es Konfigurationsparameter, die alle Streams gemeinsam haben, und solche, die streamspezifisch sind. Der Logger legt das Ziel, die Kompressionsmethode, MQTT-Zugangsdaten usw. fest, während z. B. die Datengröße und Start/Stop-Funktionalität streamspezifisch sind.

In den meisten Fällen ist es nicht unbedingt notwendig, jeden Taskzyklus Daten zu senden oder zu schreiben, sodass die in einem Zyklus aufgezeichneten Daten eines Streams, ein sogenanntes **Sample**, gepuffert werden können, bevor sie an den Broker gesendet oder in eine Datei geschrieben werden. Die Anzahl der Samples in einem Puffer und damit die Zykluszeit eines gesendeten/geschriebenen **Puffers** kann konfiguriert werden. Darüber hinaus kann die Anzahl der Puffer in einer Datei und damit die **Dateigröße** konfiguriert werden.

5.2 MQTT Grundlagen

MQTT (Message Queueing Telemetry Transport) ist ein Publisher/Subscriber-basiertes Kommunikationsprotokoll, welches eine Nachrichten-basierte Übertragung zwischen Applikationen ermöglicht. Eine zentrale Komponente bei dieser Art der Übertragung ist der sogenannte Message Broker. Dieser hat die Aufgabe, Nachrichten zwischen den einzelnen Applikationen, bzw. dem Sender und Empfänger einer Nachricht, zu verteilen. Der Message Broker entkoppelt dabei Sender und Empfänger voneinander, sodass diese keine gegenseitigen Addressinformationen kennen und austauschen müssen. Alle Kommunikationsteilnehmer wenden sich beim Senden und Empfangen an den Message Broker und dieser übernimmt die Verteilung der Nachrichten.



ClientID

Beim Herstellen einer Verbindung mit dem Message Broker übermittelt der Client eine sogenannte ClientID, welche zur eindeutigen Identifizierung des Clients auf dem Message Broker dient. Der MQTT-Kommunikationstreiber aus TwinCAT3 erzeugt automatisch eine eigene ClientID, welche sich an dem folgenden Namensschema orientiert:

PlcProjectName-TcMqttClient%n

%n ist hierbei ein inkrementeller Zähler für die Nummer der jeweiligen MQTT Client Instanz. Jede Instanz des Funktionsbausteins FB_lotMqttClient erhöht hierbei diesen Zähler. In den meisten Fällen ist das Verwenden dieses ClientID Formats ausreichend. In speziellen Fällen, z.B. abhängig vom Message Broker oder auch durch die eigene MQTT Applikation bedingt, muss eine anwendungsspezifische ClientID vergeben werden. Dies kann über einen entsprechenden Eingang an den Funktionsbausteinen FB_lotMqttClient und FB_lotMqtt5Client erfolgen.

Soll eine eindeutige ClientID automatisch beim Start des SPS Projekts generiert werden, so bietet sich die Verwendung einer GUID an, welche über den Funktionsbaustein FB_CreateGuid aus der Bibliothek Tc2_System erzeugt werden kann. Der folgende Beispielcode verdeutlicht die Verwendung dieses Funktionsbausteins.

```
PROGRAM MAIN
VAR
  fbGuid : FB_CreateGUID;
  objGuid : GUID;
  sGuid : STRING;
  nState : UINT;
  bStart : BOOL; // set to TRUE to start this sample
END_VAR

CASE nState OF
  0 :
    IF bStart THEN
      bStart := FALSE;
      nState := nState + 1;
    END_IF
END_CASE
```



```
1 : // create GUID using FB_CreateGuid from Tc2_System library
fbGuid(bExecute := TRUE, pGuidBuffer := ADR(objGuid), nGuidBufferSize := SIZEOF(objGuid));
IF NOT fbGuid.bBusy THEN
  fbGuid(bExecute := FALSE);
  IF NOT fbGuid.bError THEN
    nState := nState + 1;
  ELSE
    nState := 255; // go to error state
  END_IF
END_IF

2: // GUID has been created, now convert to STRING
sGuid := GUID_TO_STRING(objGuid);
nState := nState + 1;

3: // done

255: // error state

END_CASE
```

Nach Ausführung dieser State Machine enthält die Variable sGuid die generierte GUID als STRING. Diese kann dann an den Funktionsbausteinen FB_IotMqttClient und FB_IotMqtt5Client als ClientID verwendet werden.

Payload

Der Nachrichtinhalt einer MQTT-Nachricht wird als sogenannter Payload bezeichnet. Es können beliebige Daten übertragen werden, beispielsweise ein Text, ein einzelner Zahlenwert oder eine gesamte Informationsstruktur.

● Message-Payload-Formatierung

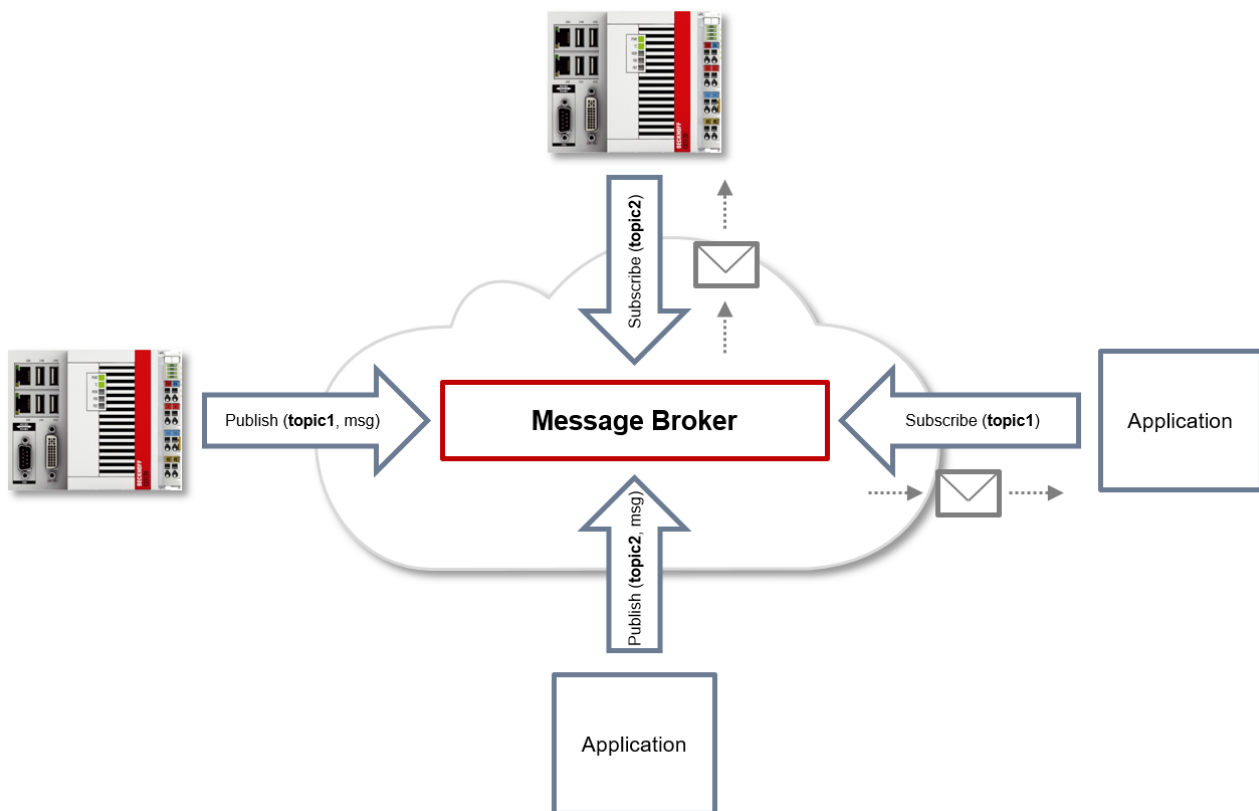
I Beachten Sie, dass der Datentyp und die Formatierung des Inhalts der Sender- und Empfängerseite bekannt sein müssen, insbesondere beim Versand von Binärinformationen (Alignment) oder Strings (mit/ohne Nullterminierung).

Topics

Bei Verwendung eines Message Brokers, welcher auf dem Protokoll MQTT basiert, wird das Senden (Publish) und Abonnieren (Subscribe) von Nachrichten mithilfe sogenannter Topics organisiert. Der Message Broker filtert eingehende Nachrichten anhand dieser Topics für jeden verbundenen Client. Ein Topic kann hierbei auch aus mehreren Ebenen bestehen, wobei die Ebenen durch ein „/“ voneinander getrennt sind.

Beispiel: Campus / Building1 / Floor2 / Room3 / Temperature

Der Publisher einer Nachricht gibt beim Versand immer an, für welches Topic eine Nachricht gedacht ist. Ein Subscriber hingegen gibt an, für welches Topic er sich interessiert. Der Message Broker leitet dann die Nachricht entsprechend weiter.



Beispielkommunikation 1 in der Grafik oben:

- Eine Applikation subscribed sich auf das Topic mit dem Namen „topic1“.
- Ein Controller published eine Nachricht an das Topic mit dem Namen „topic1“.
- Der Message Broker leitet die Nachricht entsprechend an die Applikation weiter.

Beispielkommunikation 2 in der Grafik oben:

- Ein Controller subscribed sich auf das Topic mit dem Namen „topic2“.
- Eine Applikation published eine Nachricht an das Topic mit dem Namen „topic2“.
- Der Message Broker leitet die Nachricht entsprechend an den Controller weiter.

Wildcards

Bei der Verwendung von Topics können auch sogenannte „Wildcards“ benutzt werden. Eine Wildcard ersetzt einen Teil des Topics. Ein Subscriber erhält dann ggf. Nachrichten aus mehreren Topics. Es werden zwei Arten von Wildcards unterschieden:

- Single Level Wildcards
- Multi Level Wildcards

Beispiel „Single Level Wildcard“:

Das +-Symbol beschreibt eine Single Level Wildcard. Wird es z.B. vom Subscriber wie folgt verwendet, so werden entsprechende Nachrichten an die Topics entweder vom Subscriber empfangen oder nicht empfangen.

- Der Empfänger subscribed sich auf Campus/Building1/Floor2+/Temperature
- Der Publisher sendet an Campus/Building1/Floor2/Room1/Temperature - OK
- Der Publisher sendet an Campus/Building1/Floor2/Room2/Temperature - OK
- Der Publisher sendet an Campus/Building42/Floor1/Room1/Temperature - NOK
- Der Publisher sendet an Campus/Building1/Floor2/Room1/Fridge/Temperature - NOK

Beispiel „Multi Level Wildcard“:

Das #-Symbol beschreibt eine Multi Level Wildcard. Wird es z.B. vom Subscriber wie folgt verwendet, so werden entsprechende Nachrichten an die Topics entweder vom Subscriber empfangen oder nicht empfangen. Das #-Symbol muss hierbei immer als letztes Symbol im Topic-String verwendet werden.

- Der Empfänger subscribed sich auf Campus/Building1/Floor2/#
- Der Publisher sendet an Campus/Building1/Floor2/Room1/Temperature - OK
- Der Publisher sendet an Campus/Building1/Floor2/Room2/Temperature - OK
- Der Publisher sendet an Campus/Building42/Floor1/Room1/Temperature - NOK
- Der Publisher sendet an Campus/Building1/Floor2/Room1/Fridge/Temperature – OK
- Der Publisher sendet an Campus/Building1/Floor2/Room1/Humidity - OK

QoS (Quality of Service)

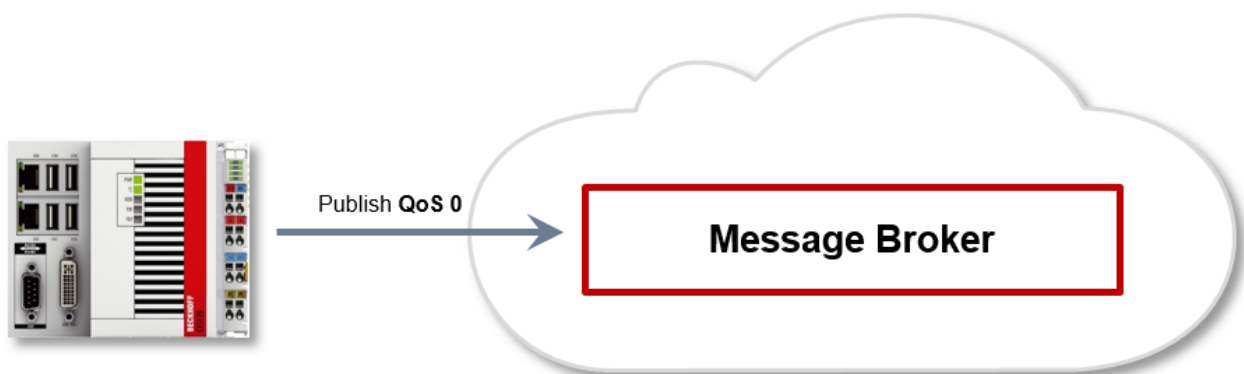
QoS ist eine Vereinbarung zwischen dem Sender und Empfänger einer Nachricht in Bezug auf das Garantieren der Nachrichtenübermittlung. Es existieren drei verschiedene Level in MQTT:

- 0 – höchstens einmal
- 1 – mindestens einmal
- 2 – genau einmal

Beide Kommunikationsarten (Publish/Subscribe) zum Message Broker müssen berücksichtigt und getrennt voneinander betrachtet werden. Das QoS-Level, welches ein Client beim Publishen einer Nachricht verwendet, wird vom jeweiligen Client gesetzt. Wenn der Broker nun die Nachricht an einen Client weiterleitet, der sich entsprechend auf das Topic subscribed hat, wird das QoS-Level vom Subscriber verwendet, welches beim Herstellen der Subscription angegeben wurde. Dies bedeutet, dass ein QoS-Level, welches vom Publisher vielleicht mit 2 angegeben wurde, vom Subscriber mit 0 „überschrieben“ werden kann.

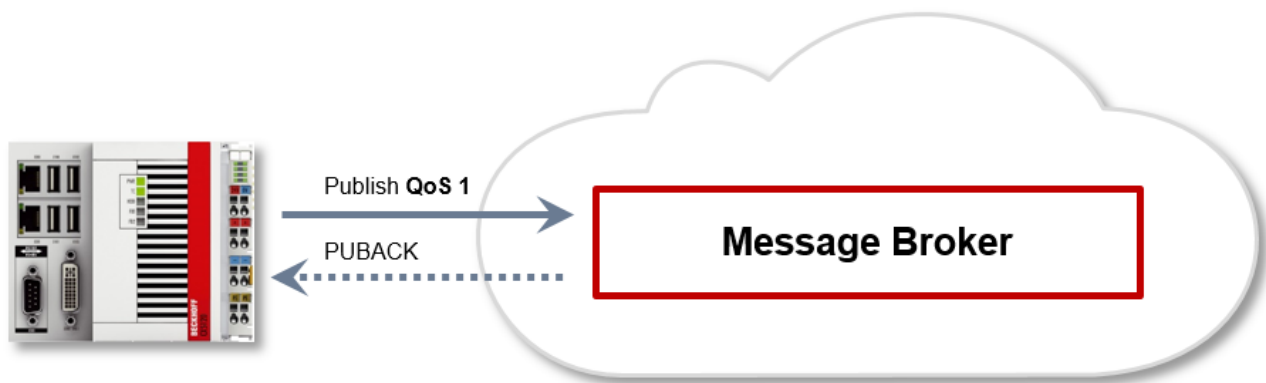
QoS-Level 0

Bei diesem QoS-Level erfolgt keine Bestätigung des Empfängers, ob die Nachrichten empfangen wurden oder nicht. In der Folge wird die Nachricht auch kein zweites Mal gesendet.



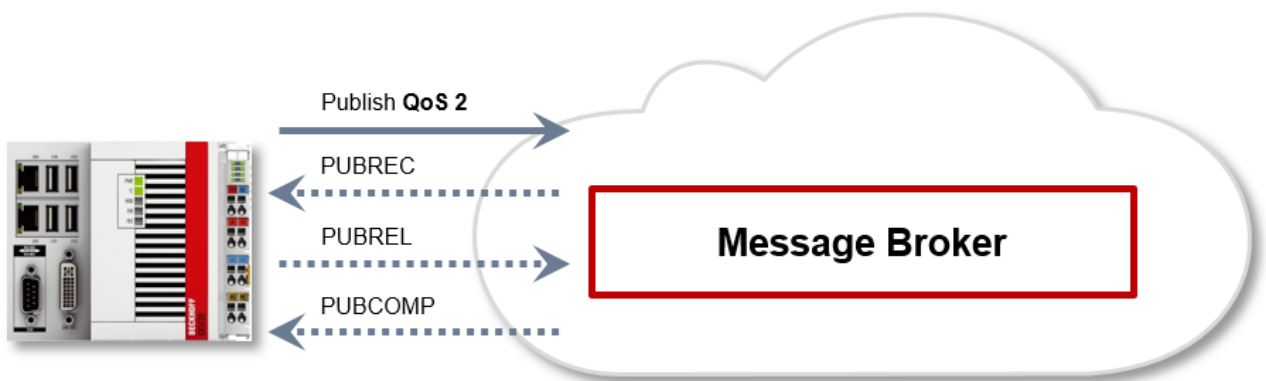
QoS-Level 1

Bei diesem QoS-Level wird garantiert, dass die Nachricht zumindest einmal beim Empfänger ankommt. Aber die Nachricht kann unter Umständen auch mehrfach beim Empfänger eintreffen. Der Sender speichert die Nachricht intern bis er eine Bestätigung in Form einer PUBACK-Nachricht vom Empfänger erhält. Wenn die PUBACK-Nachricht für eine bestimmte Zeit ausbleibt, wird die Nachricht erneut gesendet.



QoS-Level 2

Bei diesem QoS-Level wird garantiert, dass die Nachricht maximal einmal beim Empfänger ankommt. Dies wird MQTT-seitig durch einen Handshake-Mechanismus realisiert. QoS-Level 2 ist der sicherste (aus Sicht der Nachrichtenübermittlung), aber auch langsamste QoS-Level. Wenn ein Empfänger eine Nachricht mit QoS-Level 2 erhält, bestätigt er die Nachricht mit einem PUBREC. Der Absender der Nachricht merkt sich diese intern bis er ein PUBCOMP empfangen hat. Dieser zusätzliche Handshake (verglichen mit QoS 1) ist wichtig, damit die Nachricht nicht doppelt übertragen wird. Wenn der Absender der Nachricht ein PUBREC erhält, kann er den initialen Publish verwerfen, da er weiß, dass die Nachricht einmal vom Empfänger empfangen wurde. Er merkt sich somit intern den PUBREC und sendet seinerseits ein PUBREL. Nachdem der Empfänger ein PUBREL empfangen hat, kann er die sich zuvor gemerkten Zustände verwerfen und mit einem PUBCOMP antworten. Umgekehrt genauso. Immer dann, wenn ein Paket verloren geht, ist der jeweilige Kommunikationsteilnehmer dafür verantwortlich, die zuletzt gesendete Nachricht nach einer bestimmten Zeit noch einmal zu senden.



Der LastWill ist eine Nachricht, die im Falle eines irregulären Verbindungsabbruches vom Broker an alle Clients gesendet wird, die das passende Topic abonniert haben. Verliert der MQTT-Client in der SPS die Verbindung zum Broker und es wurde beim Verbindungsaufbau ein LastWill hinterlegt, so wird dieser LastWill vom Broker kommuniziert, ohne dass der Client sich darum kümmern muss.

Bei einem geplanten Disconnect wird der LastWill laut Spezifikation nicht zwingend übertragen. Aus Sicht des SPS-Programmierers kann dieser entscheiden, ob er vor Aufruf des Disconnects den LastWill publishen will. Dazu wird auf dem LastWill-Topic die LastWill-Nachricht noch einmal gepublished. Das ist notwendig, da der Broker aufgrund der regulären Verbindungsabbruches die Last Will-Nachricht nicht veröffentlichen würde.

Bei einem TwinCAT-Kontextwechsel und einem daraus folgenden Neustart der MQTT-Kommunikation sendet der IoT-Treiber den vorher spezifizierten LastWill an den Broker, weil in diesem Moment aus der SPS keine Möglichkeit mehr dazu besteht. Wenn bei Verbindungsherstellung kein LastWill definiert wurde, wird auch keine Nachricht vor dem Disconnect übertragen.

Sicherheit

Bei der Herstellung einer Verbindung zum Message Broker können hierbei auch Sicherheitsmechanismen, wie TLS eingesetzt werden, um die Kommunikationsverbindung zu verschlüsseln oder eine Authentifizierung zwischen Client und Message Broker zu realisieren.

Quellen

Für weitere und detailliertere Informationen zu MQTT empfehlen wir die folgenden Webseiten:

HiveMq Blog: <http://www.hivemq.com/blog/mqtt-essentials/> (Hauptgrundlage dieses Artikels)

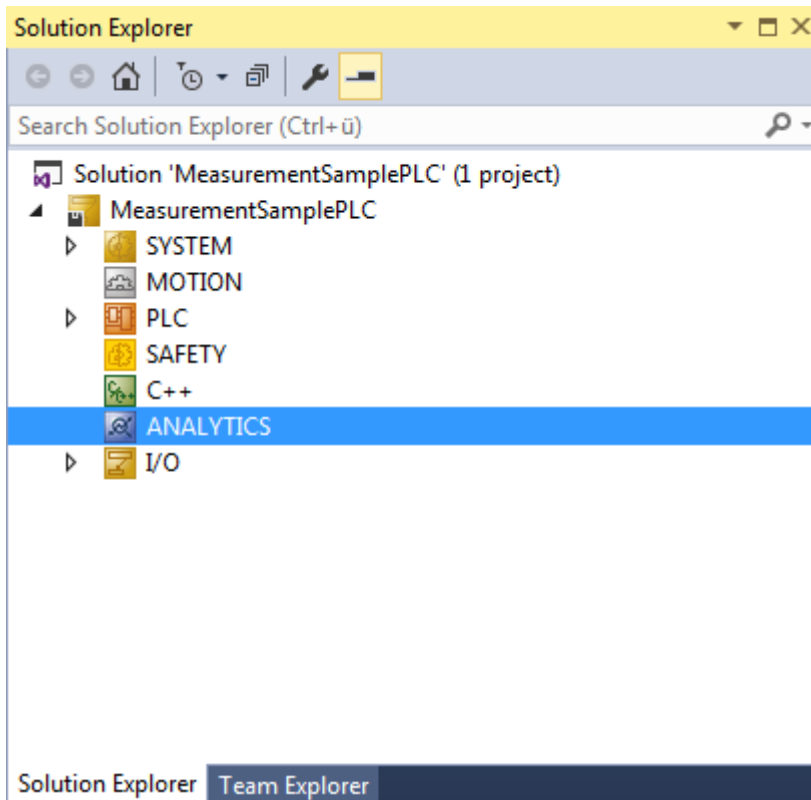
5.3 Datenkompression

Um die Menge der zu sendenden Daten zu verringern, ohne die Menge an Informationen zu verringern, und so die Leistung zu verbessern, kann ein vom Run Length Encoding-Verfahren abgeleitetes Kompressionsverfahren verwendet werden. Dieses macht sich zunutze, dass Daten von zwei aufeinanderfolgenden Samples in einem Puffer nicht in Teilen variieren dürfen. Durch das Wissen, welche Teile eines zuvor gesendeten Puffers gleich sind, in Verbindung mit den variierenden Daten, kann ein Empfänger den nächsten Puffer rekonstruieren, ohne dass er den gesamten Puffer erhalten muss. Nachdem der erste Puffer unkomprimiert gesendet wurde, konstruiert der Logger den komprimierten Puffer, indem er vom Benutzer festgelegte Dateneinheiten (z. B. jedes Byte, alle 8 Byte usw.) nacheinander vergleicht. Der Logger zählt die Menge der Dateneinheiten (bezeichnet als Compression Compare Width), die gleich sind, und platziert diese Information anstelle der Daten in den Puffer. Abhängig von der Art der Daten kann dies zu einer Dateneinsparung oder einem Overhead führen. Um zu entscheiden, ob eine Kompression sinnvoll ist oder nicht, wird dem Benutzer ein Einsparungswert der Datenkompression bereitgestellt, der sich auf dem Karteireiter Online jedes Stream-Dialogfensters findet. Ein positiver Wert bedeutet eine Einsparung und ein negativer Wert einen Overhead.

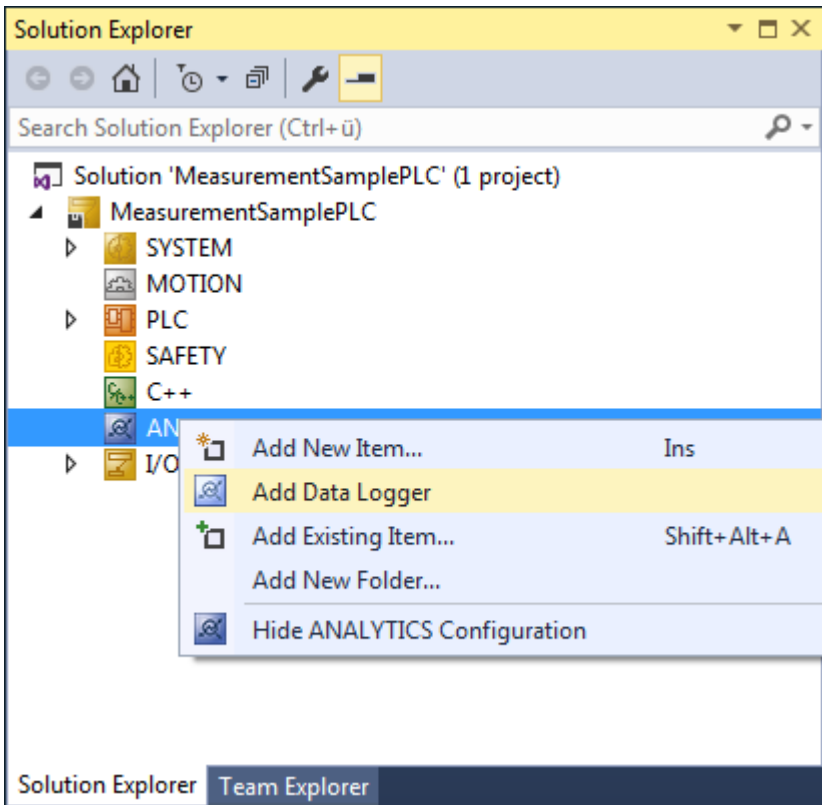
6 Konfiguration

6.1 Basiseinstellungen

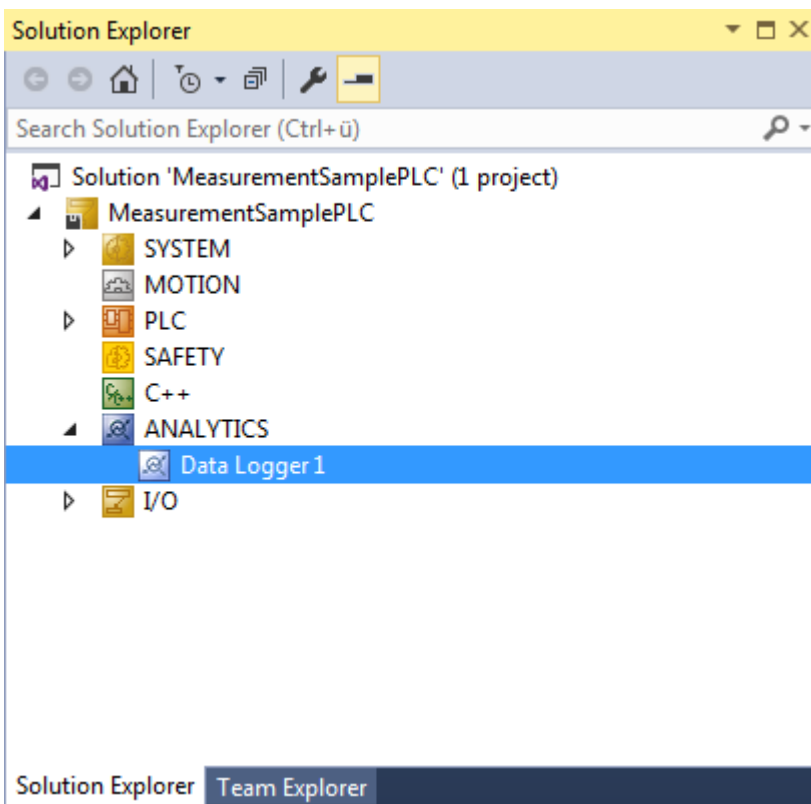
Um den TwinCAT Analytics Logger zu konfigurieren, wird dem Benutzer eine spezielle Analytics-Konfiguration in einem XAE-Projekt bereitgestellt.



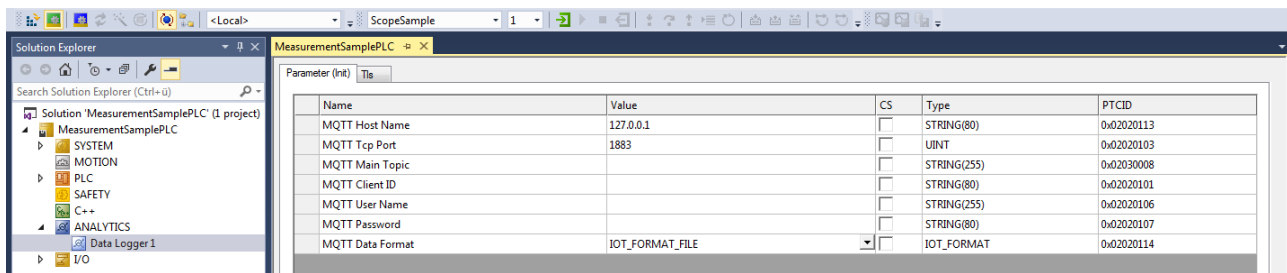
Um einen Daten-Logger hinzuzufügen, wählen Sie den entsprechenden Punkt im Kontextmenü dieses Konfigurationsknotens aus.



Dies kann entweder den zusätzlichen Daten-Logger-Knoten allein oder untergeordnete Streamknoten zum Ergebnis haben, falls bereits Variablen vorhanden sind, die geloggt werden können.



Durch Doppelklick auf den neuen Daten-Logger-Knoten wird das Editorfenster geöffnet. Im Karteireiter Parameter können Sie Ihre spezifischen Analytics Logger-Einstellungen vornehmen.



- **Data Format:** Hier kann der Benutzer zwischen *IOT_FORMAT_FILE* und *IOT_FORMAT_MQTT_BINARY* wählen. Bei Verwendung des *FILE*-Formats speichert der Analytics Logger die Daten in lokalen Binärdateien. Die Dateien werden in *C:\TwinCAT\3.1\Boot\Analytics* gespeichert. Bei Verwendung von *MQTT_BINARY* werden die Daten an den konfigurierten MQTT-Message-Broker gesendet.
- **Data Compression:** Hier kann die Datenkomprimierung ein- und ausgeschaltet werden.
- **Max. Compression Compare Width:** Stellt den Komprimierungsmodus ein.
- **MQTT Host Name:** Geben Sie hier den Hostnamen oder die IP-Adresse Ihres nativen MQTT-Message-Brokers ein.
- **MQTT Tcp Port:** Legen Sie hier den Tcp-Port für die Kommunikation fest. MQTT-Standardport: 1883.
- **MQTT Main Topic:** Es ist möglich, ein eigenes und individuelles Main-Topic anzugeben. Beispiel: *Beckhoff/Verl/Production/Drives/Machine5* – der Analytics Logger fügt seine eigenen spezifischen Sub-Topics automatisch hinzu: *Beckhoff/Verl/Production/Drives/Machine5/Bin/Tx/Data*.
- **MQTT Client ID:** Die Client-ID ist eine Identifizierung jedes MQTT-Clients, der sich mit einem nativen MQTT-Message-Broker verbindet. Sie sollte für jeden Broker eindeutig sein.
- **MQTT User Name:** MQTT ermöglicht das Senden eines Benutzernamens zur Authentifizierung des Clients.
- **MQTT Password:** MQTT ermöglicht auch das Senden eines Passworts zur Authentifizierung des Clients und Autorisierung.

Es ist möglich, mehrere Logger in einem TwinCAT-Projekt vorzusehen, um Daten an verschiedene MQTT-Message-Broker zu übermitteln oder zum Teil einen Speicher in einer lokalen Binärdatei zu haben.

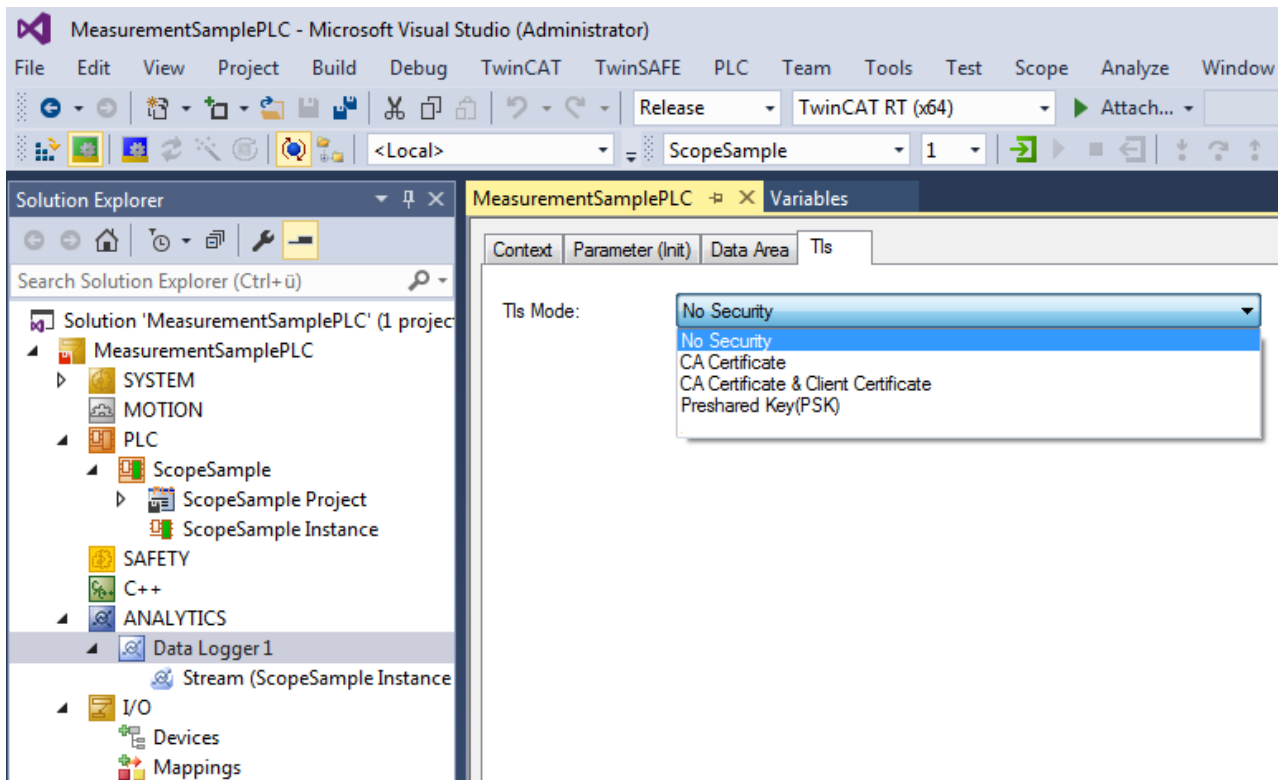
6.1.1 TLS

TLS (Transport Layer Security) sorgt für einen sicheren Kommunikationskanal zwischen einem Client und einem Server. Im Kern handelt es sich bei TLS um kryptographische Protokolle, die mit Hilfe eines Handshake-Mechanismus verschiedene Parameter vereinbaren, um eine sichere Verbindung zwischen dem Client und dem Server herzustellen. Der TwinCAT Analytics Logger unterstützt TLS Version 1.2.

MQTT-Kommunikation mit TLS

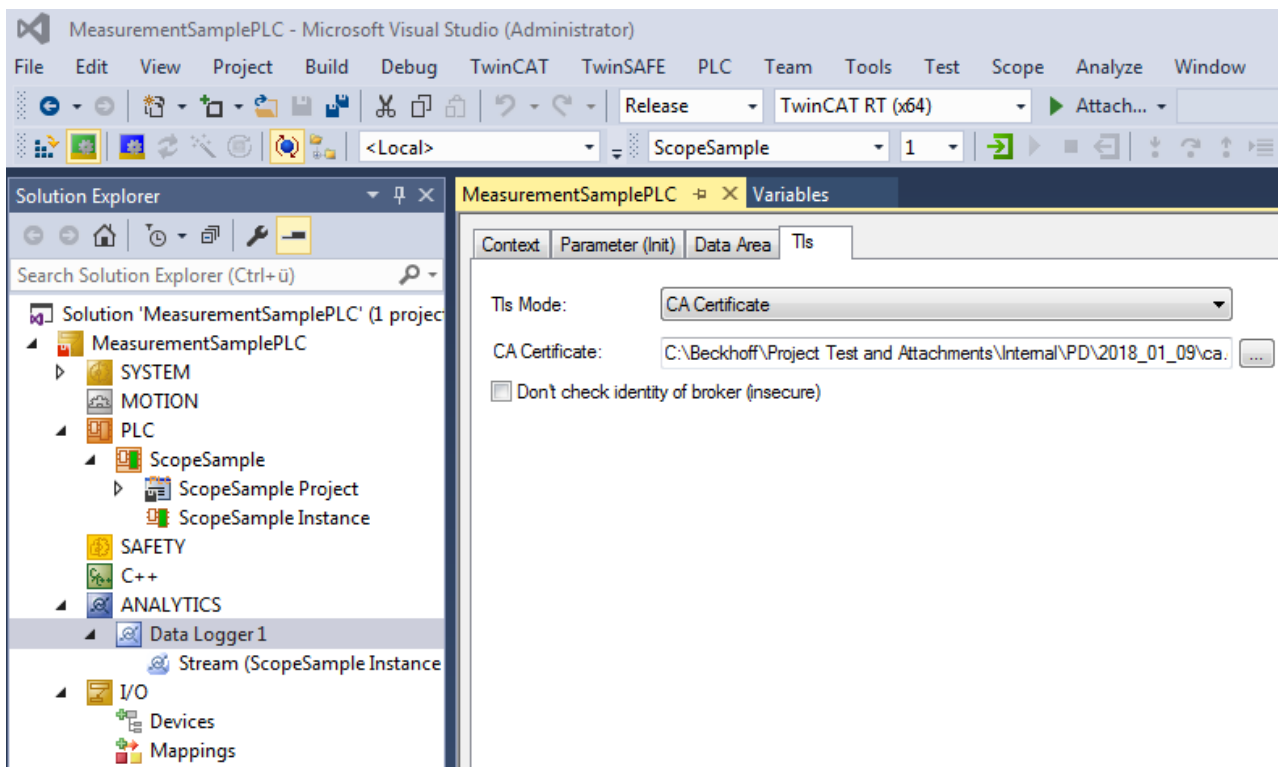
i Bei Verwendung von Zertifikaten ist der TCP-Port 8883 ausschließlich für MQTT über TLS reserviert!

Auf dem Karteireiter TLS des Daten-Loggers ist als Erstes der TLS-Modus in einem Dropdown-Feld auszuwählen. Abhängig vom Message-Broker können verschiedene TLS-Mechanismen/Modi verwendet werden. Der Analytics Logger unterstützt die Modi CA Certificates, CA Certificates & Client Certificate und Preshared Key (PSK).



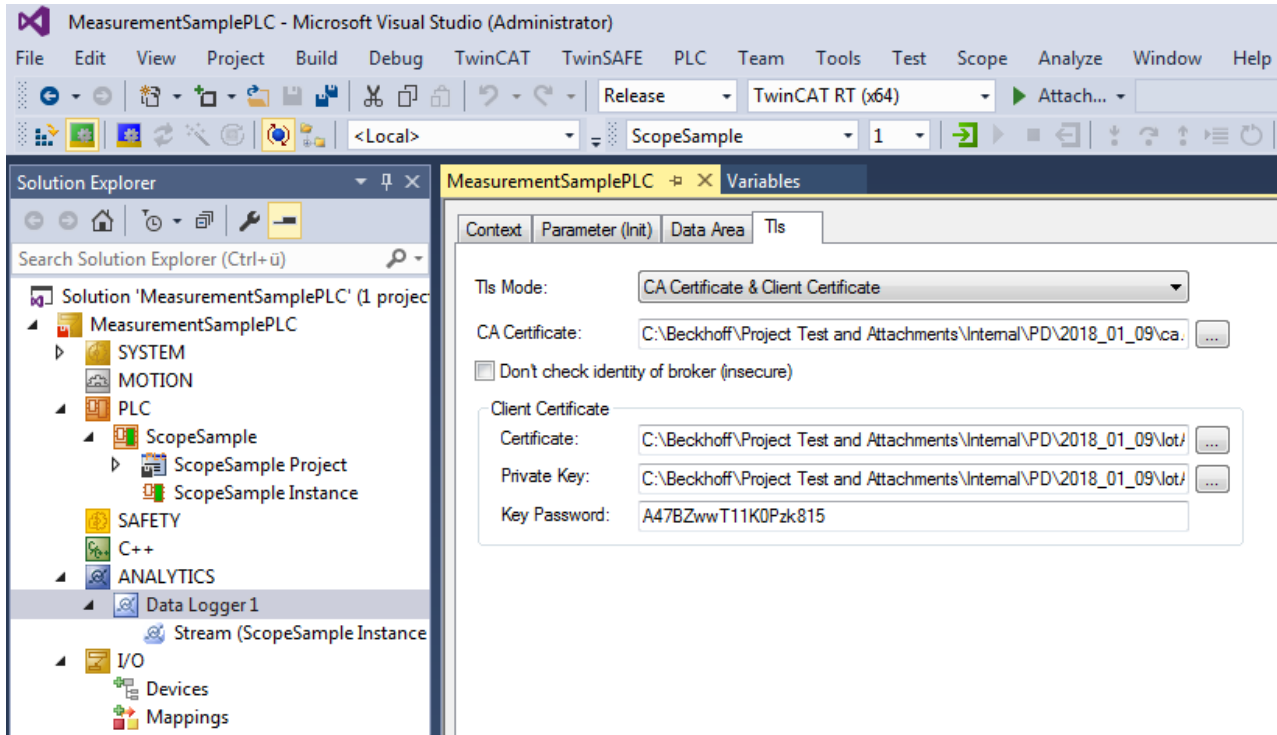
CA Certificate

Die Verschlüsselung und Authentifizierung über TLS können auch durch eine Zertifizierungsstelle (Certificate Authority; CA) erfolgen. Die CA stellt eine Signatur über den öffentlichen Schlüssel für alle Kommunikations-Clients bereit. In diesem Fall verbindet sich ein MQTT-Client ohne ein eigenes Client-Zertifikat mit einem Message-Broker.



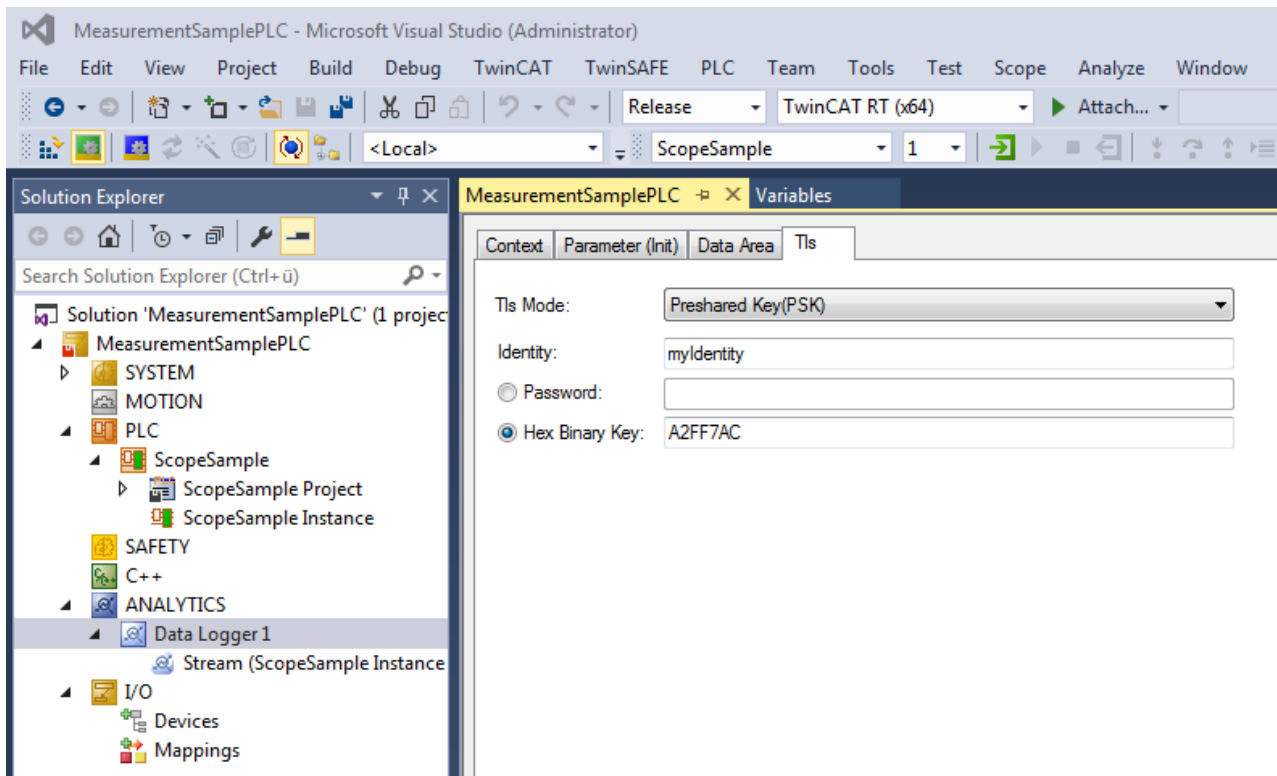
CA Certificate & Client Certificate

Die Verschlüsselung und Authentifizierung über TLS können auch durch eine Zertifizierungsstelle (Certificate Authority; CA) erfolgen. Die CA stellt eine Signatur über den öffentlichen Schlüssel für den Message-Broker (den sogenannten Serverschlüssel) und in der Regel auch für alle sich verbindenden Clients bereit. Alle Kommunikationsteilnehmer können einander vertrauen, da die ausstellende Zertifizierungsstelle vertrauenswürdig ist.



Preshared Key (PSK)

Die TLS-PreSharedKey (PSK)-Methode bietet eine einfache Option für die Durchführung der Verschlüsselung zwischen Client und Message-Broker. Client und Broker erkennen ein gemeinsames Passwort an, das zum Verschlüsseln und Entschlüsseln der Pakete verwendet wird.

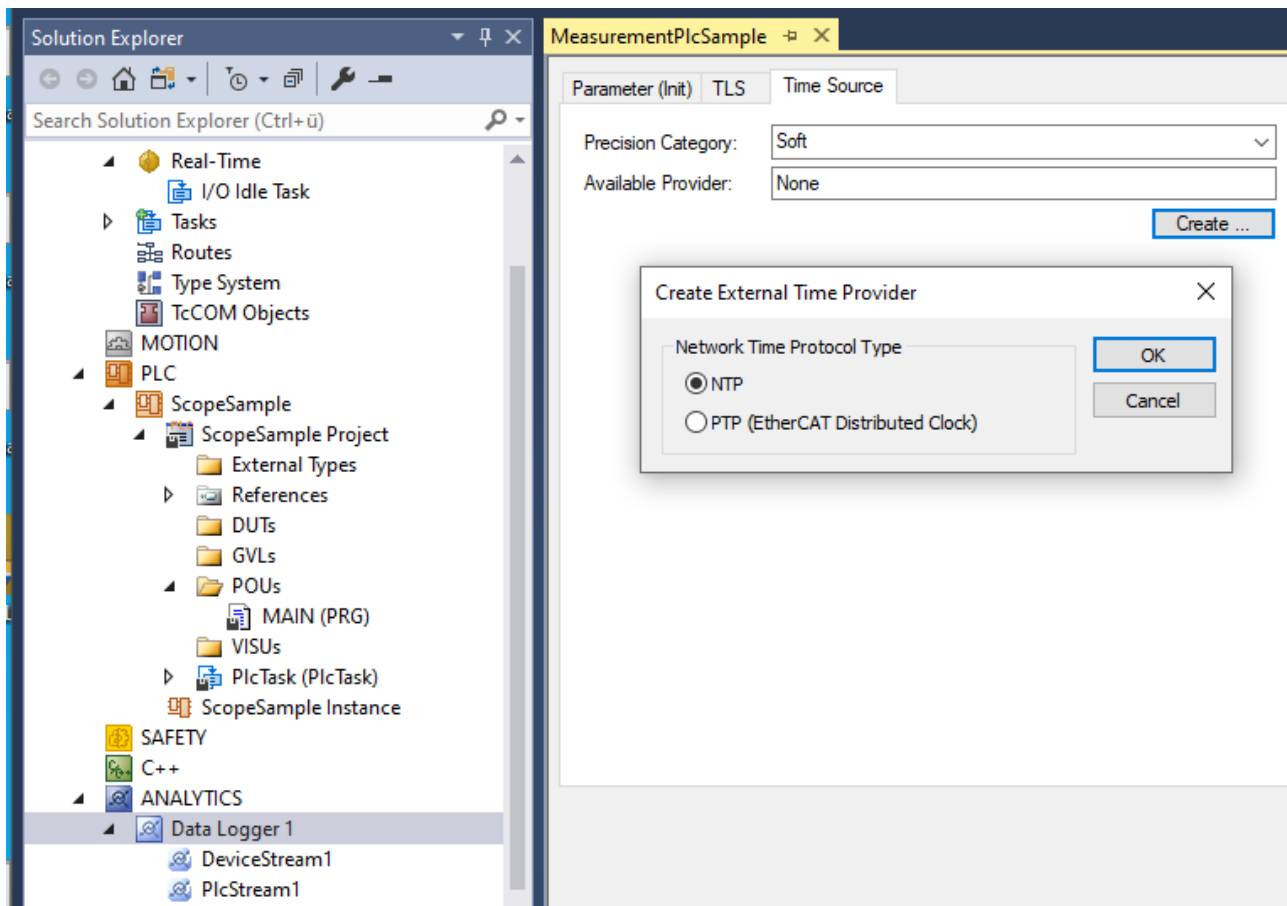


6.1.2 Zeitstempelkorrektur

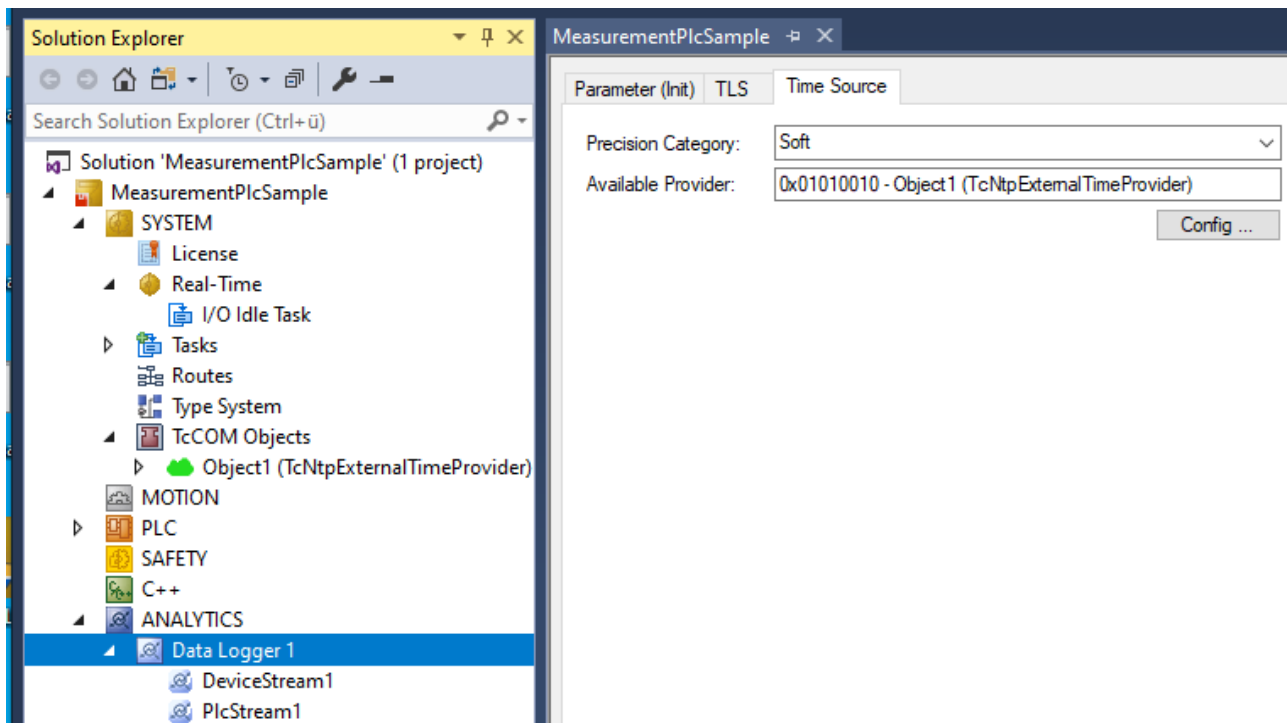
Die TwinCAT-Zeit, welche standardmäßig für Zeitstempel im Data Logger verwendet wird, weicht mit steigender Dauer, in der eine Steuerung im Run-Modus ist, immer mehr von der tatsächlichen Systemzeit ab. Dies ist darauf zurückzuführen, dass jeweils verschiedene Hardware-Counter als Taktgeber genutzt werden. Mit dem Konzept der **Zeitstempelkorrektur**, das TwinCAT bietet, besteht die Möglichkeit den aufgenommenen Samples korrigierte Zeitstempel anzuhängen. Die Korrektur mittels der External Time Provider, die TwinCAT mitbringt, kann dabei sowohl in Bezug auf eine externe Zeitquelle per NTP (Network Time Protocol), als auch in Bezug auf die EtherCAT Distributed Clock per PTP (Precision Time Protocol) erfolgen.

Die Einstellungen zur Zeitstempelkorrektur befinden sich unter dem **Data Logger** Projektknoten im Reiter **Time Source**.

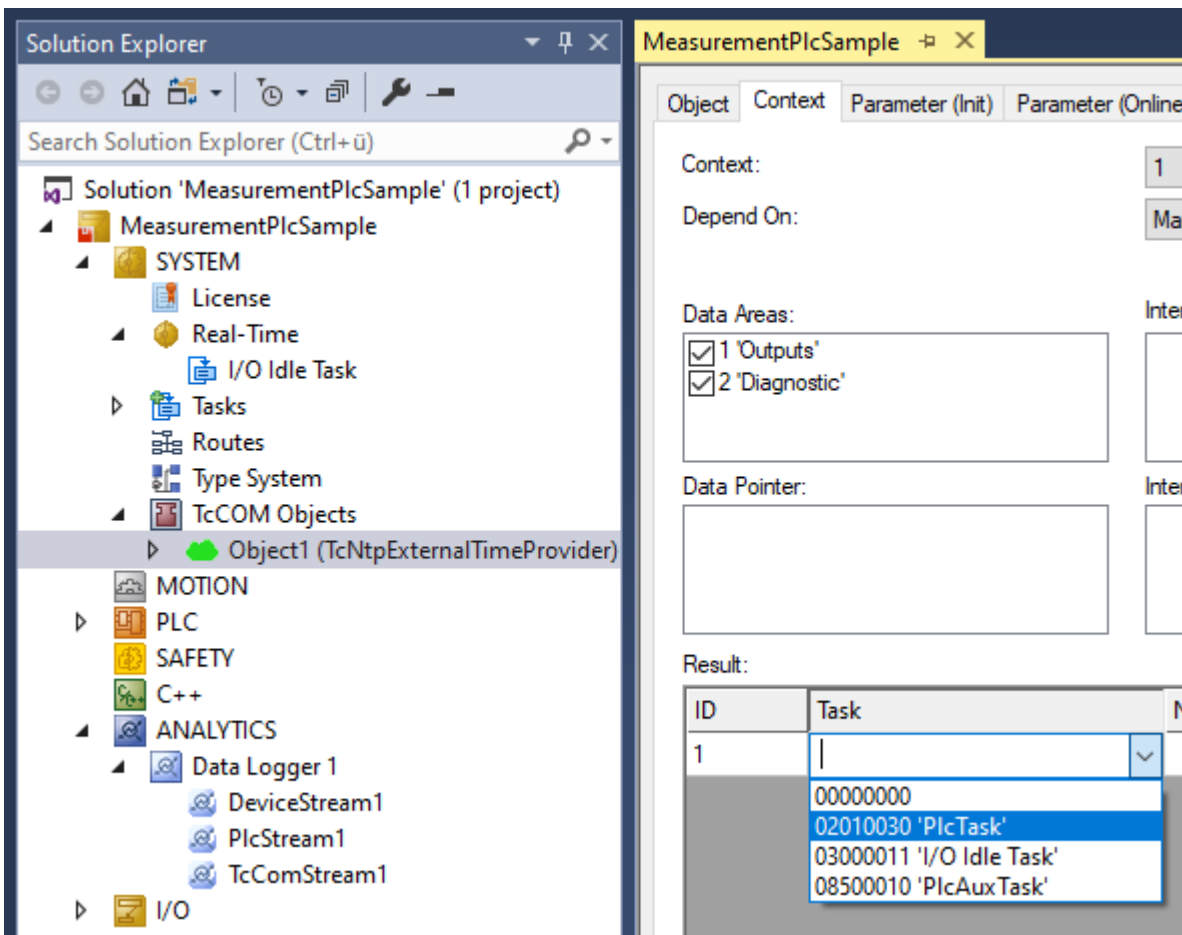
Bei Auswahl der gewünschten Genauigkeitskategorie, wird ein gegebenenfalls vorhandener External Time Provider angezeigt. Sollte noch keiner existieren, kann über den Button **Create** ein Provider erzeugt werden.



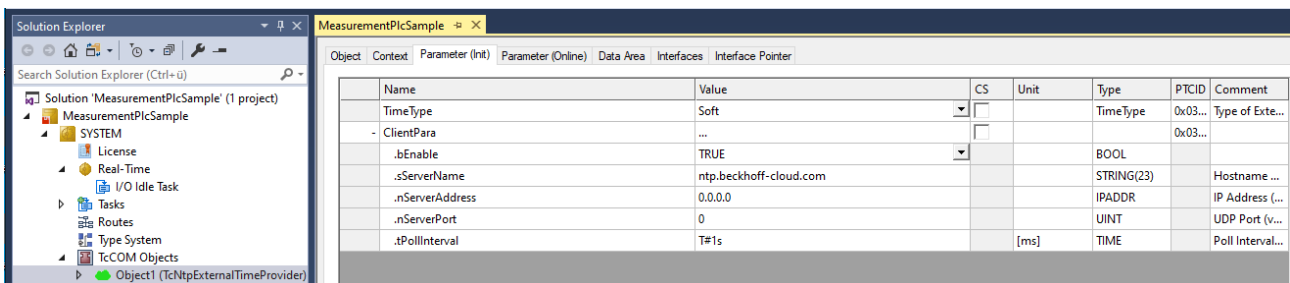
Dieser erscheint dann im Projekt unter dem TcCom-Objects Knoten, zu dem Sie über den Button **Config** springen können.



Hier sollte sichergestellt werden, dass im **Context**-Reiter die **Task** ausgewählt wird, die das Time Provider Objekt steuert.



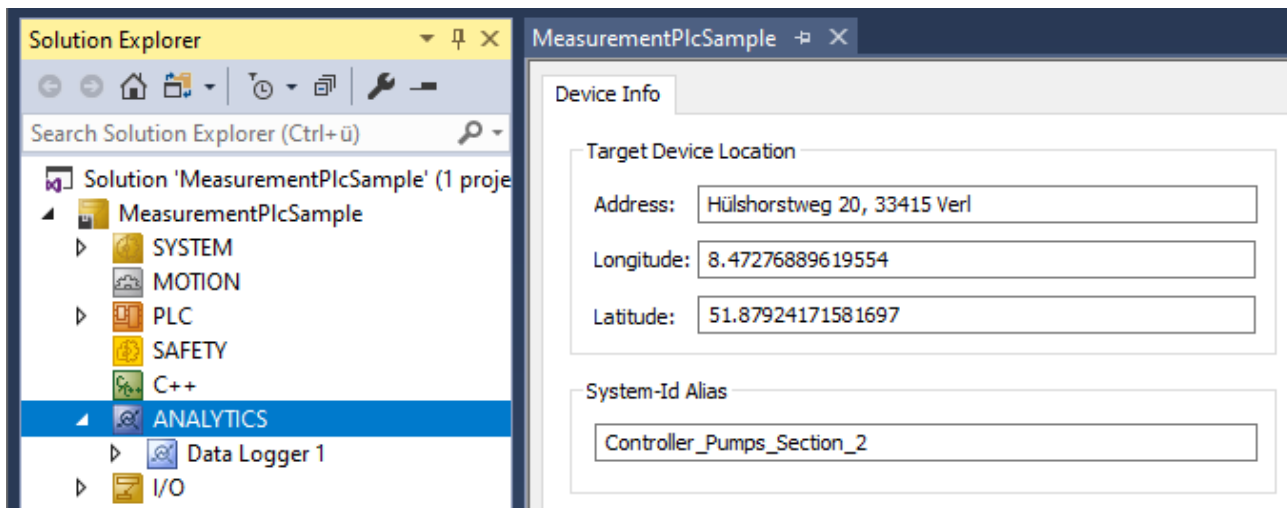
Außerdem lassen sich über die **Init-Parameter** die Zykluszeit der Synchronisation des Providers mit dem NTP-Server und der Server-Hostname einstellen.



Damit sind Zeitstempelkorrekturen, die der Logger verwendet, um einen Offset zu korrigieren, der sich aus der Differenz der TwinCAT-Zeit und der NTP-synchronen Zeit ergibt, erstellt.

6.1.3 Gerätespezifische Informationen

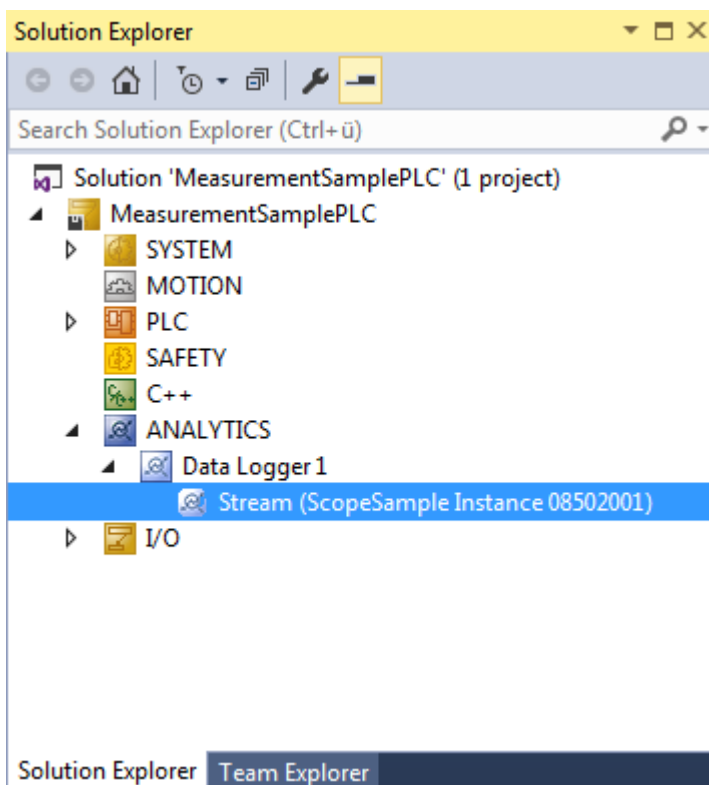
Unter dem Analytics-Projektknoten im Reiter **Device Info** können Sie Informationen zu dem Steuerungsgerät angeben. Hierzu gehören die Adresse, Koordinaten und einen Aliasnamen für die System-ID, der es dem Nutzer im Analytics-Workflow erleichtert, das Gerät zu identifizieren.



Diese Informationen werden als Teil der Stream Description auf dem entsprechenden MQTT-Topic versandt und sind für alle Data Logger und Streams gleich.

6.2 Daten-Streams

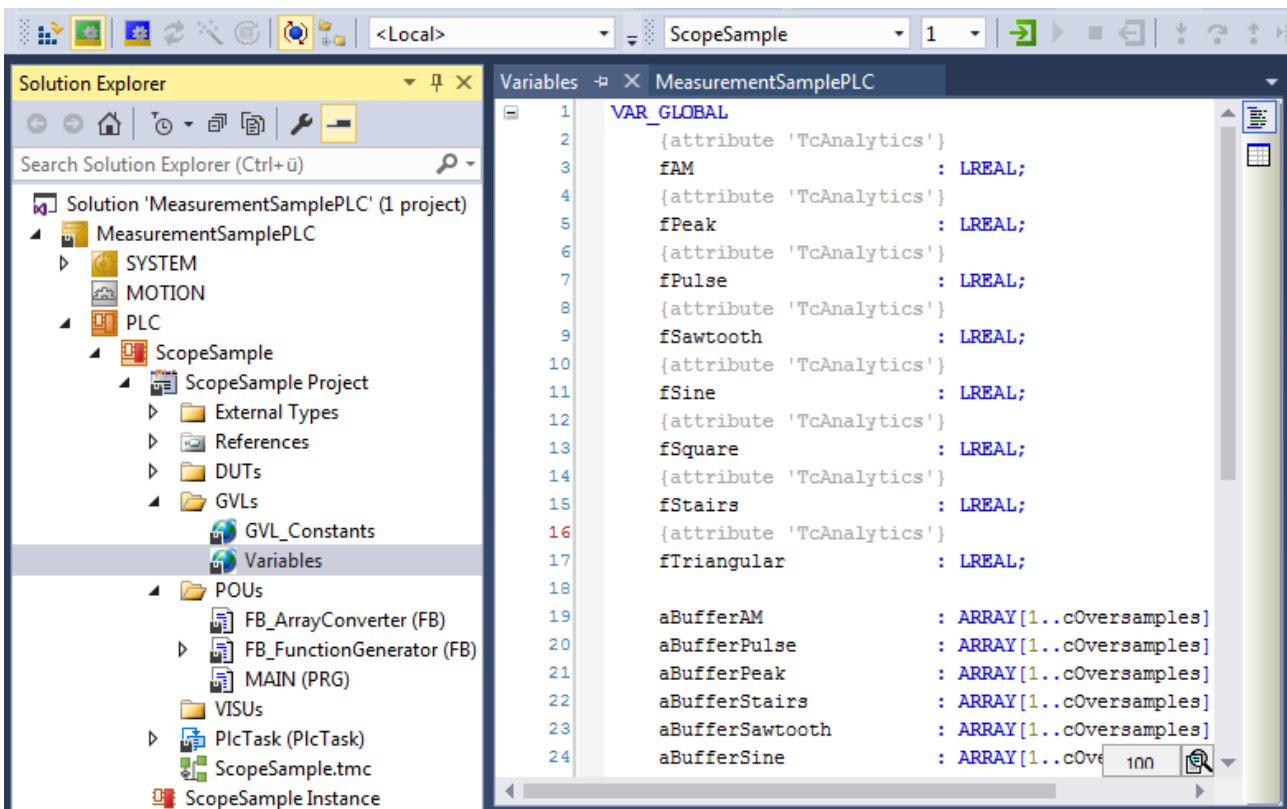
Falls Variablen zur Aufzeichnung vorhanden sind, werden sie automatisch als sogenannter Stream angezeigt. Derzeit können Daten direkt aus dem EtherCAT-Prozessabbild oder aus der SPS-Anwendung aufgezeichnet werden.



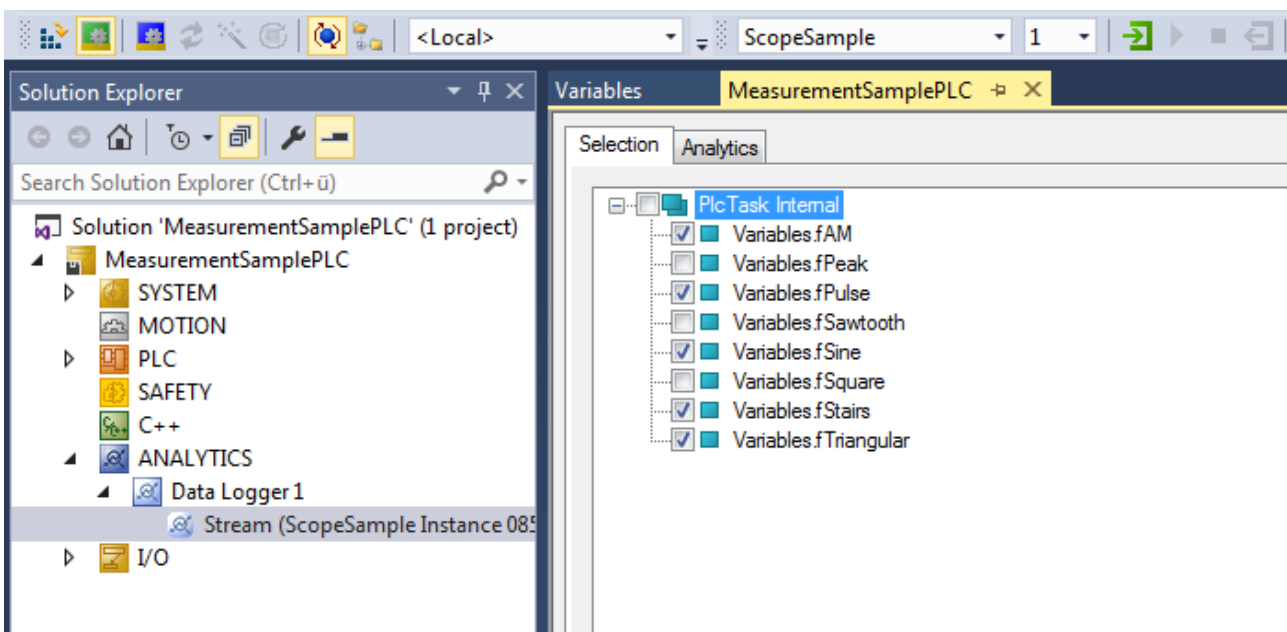
SPS-Anwendung:

Wenn Variablen aus der SPS vom Analytics Logger aufgezeichnet werden sollen, muss der Benutzer vor jede Variable in der Variablendeklaration ein Attribut setzen.

Die Attribut-Syntax lautet: {attribute 'TcAnalytics'}

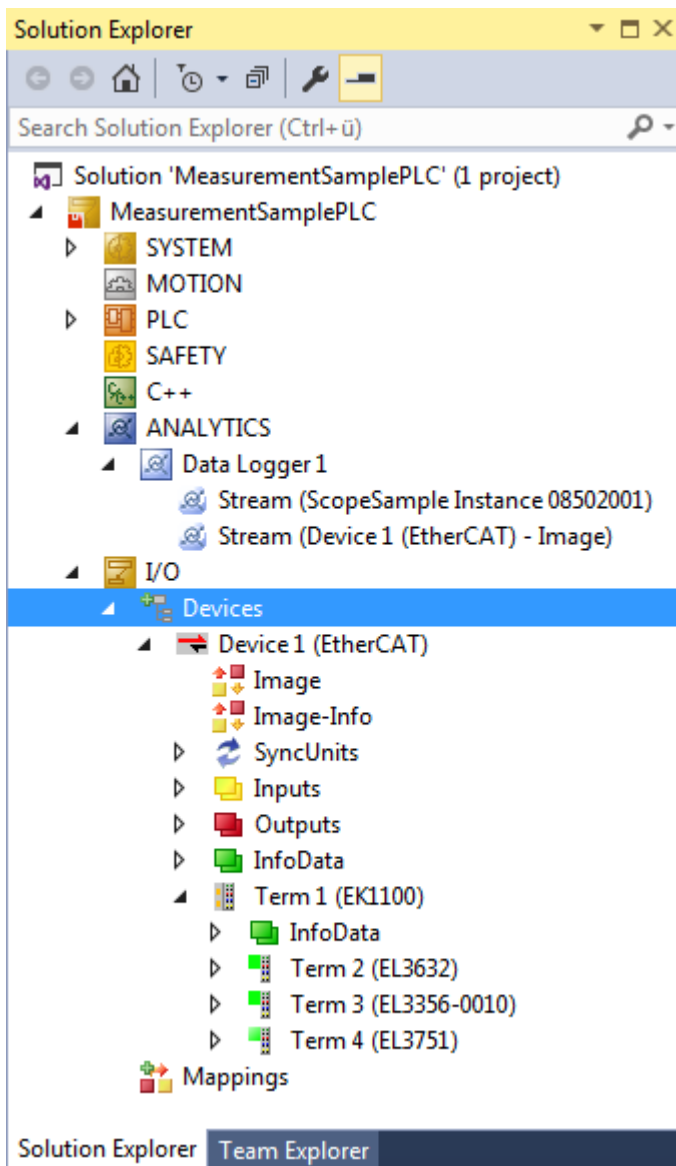


Wie bereits beschrieben, wird der Analytics Logger-Konfiguration **nach dem Rebuild** des SPS-Projekts automatisch ein neuer Stream hinzugefügt. In dem Stream gibt es einen Karteireiter *Selection*, wo schließlich die aufzuzeichnenden Variablen durch Checkboxes ausgewählt werden können.

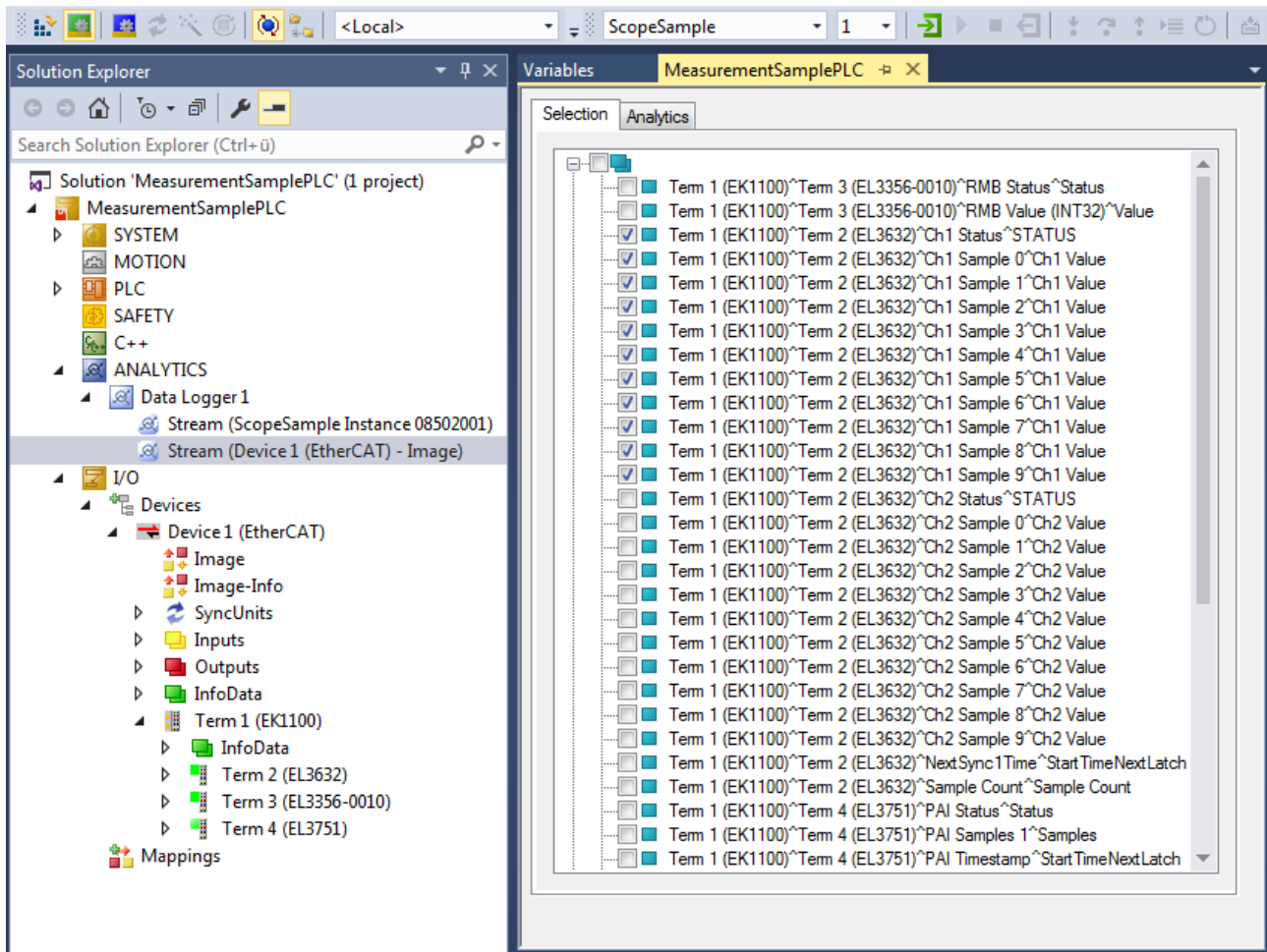


Prozessabbild:

Wenn in der gegebenen Konfiguration ein EtherCAT-Prozessabbild vorhanden ist, wird unter dem Daten-Logger-Baumknoten ein zusätzlicher Stream angezeigt.



Auf dem Karteireiter *Selection* kann der Benutzer auch hier die Werte auswählen, die vom Analytics Logger aufgezeichnet werden sollen.

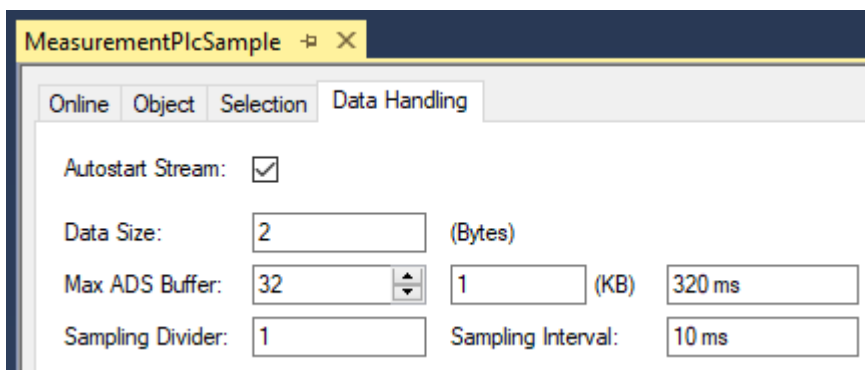


Aufzeichnung starten

Mit der Aktivierung der TwinCAT-Konfiguration startet der Analytics Logger das Loggen, das abhängig von den Basiseinstellungen an einen MQTT-Message-Broker übermittelt oder in einer lokalen Binärdatei gespeichert wird. Je nach gegebenem Format gibt es verschiedene Einstellungsmöglichkeiten auf dem Karteireiter **Data Handling** jedes Streams. Siehe hierzu das folgende Kapitel.

6.2.1 Data Handling

In dem Karteireiter Data Handling können allgemeine Einstellungen für die Paketgröße der aufgezeichneten Daten vorgenommen werden. Abhängig von dem gegebenen Datenformat enthält sie zusätzliche Einstellungen.



Autostart Stream: Legt fest, ob der Stream automatisch starten soll, sobald TwinCAT in den Run-Modus übergeht. Ist diese Option nicht ausgewählt kann der Stream über ein SPS Programm gestartet werden.

Data Size: Dies ist ein schreibgeschützter, automatisch ermittelter Wert, der die Größe der gegebenen Variablenauswahl für diesen Stream angibt d.h. die Sample-Größe.

Max ADS Buffer: Hier können Sie die Anzahl der gepufferten Samples vor dem Schreiben in eine Datei oder Senden an den Message-Broker einstellen. Bei einer Samplerate von 1 ms und 32 Samples pro Puffer, braucht der Analytics Logger 32 ms, bevor er den Puffer per MQTT versendet, oder in eine Datei schreibt. Dies ist eine individuelle Einstellung, die von den Systemressourcen abhängt.

Sampling Divider: Dieser ermöglicht eine Verringerung der Samplingrate, welche man durch Division der Inversen der Taskzykluszeit mit dem hier angegebenen Wert erhält.

Data format: MQTT

Ist das Datenformat *IOT_FORMAT_BINARY*, dann kann eine zusätzliche Checkbox aktiviert werden, um Meldungen bei einer Unterbrechung der Verbindung zu puffern.

The screenshot shows the 'MeasurementPlcSample' configuration window with the 'Data Handling' tab selected. The settings are as follows:

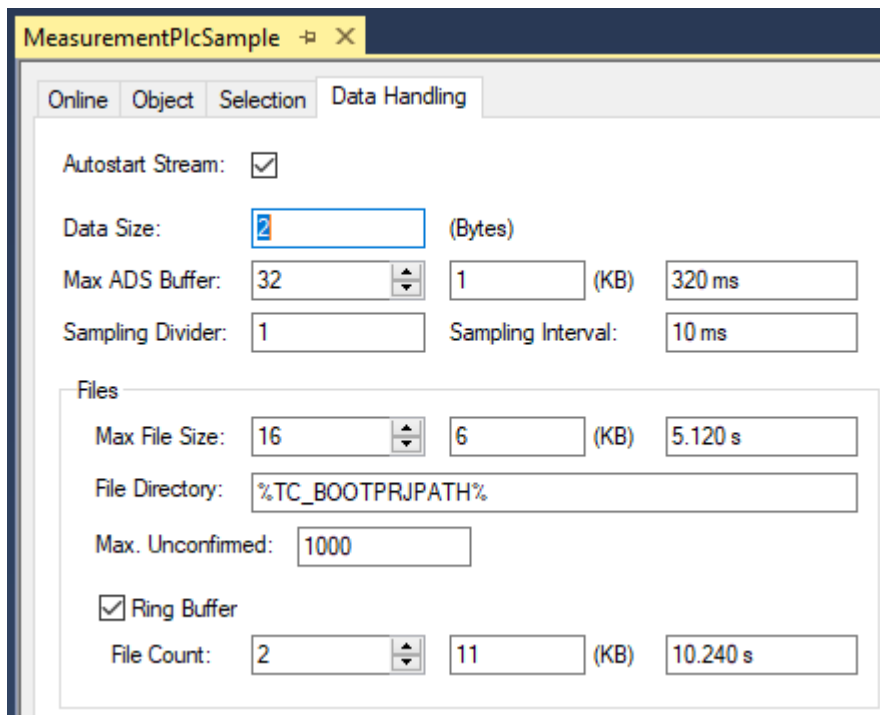
- Autostart Stream:**
- Data Size:** 2 (Bytes)
- Max ADS Buffer:** 32 (Samples), 1 (KB), 320 ms
- Sampling Divider:** 1, **Sampling Interval:** 10 ms
- Files:**
 - Max File Size:** 16 (Samples), 6 (KB), 5.120 s
 - File Directory:** %TC_BOOTPRJPATH%
 - Max. Unconfirmed:** 1000
 - Ring Buffer**
 - File Count:** 2, 11 (KB), 10.240 s
- MQTT:**
 - Queue messages when disconnected**
 - Queue Size:** 62 (Samples), 26 (KB), 19.840 s
 - Store in file**

File Store: Wenn diese Option aktiviert ist, werden die gepufferten Meldungen in einer temporären Datei auf der Festplatte gespeichert. Anderenfalls werden die Daten im RAM gespeichert, wenn die Verbindung zum Message-Broker unterbrochen wird.

Queue Size: Dies ist die Anzahl der konfigurierten ADS-Puffer, die im Falle eines Verbindungsabbruchs gespeichert werden sollen.

Data format: File

Ist das Datenformat *IOT_FORMAT_FILE* ausgewählt, stehen eine Reihe weiterer Einstellungsmöglichkeiten zur Verfügung.



MeasurementPlcSample

Online Object Selection **Data Handling**

Autostart Stream:

Data Size: (Bytes)

Max ADS Buffer: (KB)

Sampling Divider: Sampling Interval:

Files

Max File Size: (KB)

File Directory:

Max. Unconfirmed:

Ring Buffer

File Count: (KB)

Max File Size: Hier kann die maximale Anzahl an Puffern angegeben werden, die in eine Datei geschrieben werden, bevor mit einer neuen Datei begonnen wird. Daraus ergibt sich eine maximale Dateigröße. Die Dateien werden unter C:\TwinCAT\3.1\Boot\Analytics gespeichert.

File Directory: Standardmäßig wird pro Stream ein Analytics Unterordner in C:\TwinCAT\3.1\Boot\ (%TC_BOOTPRJPATH%) angelegt. Hier wiederum wird pro Stream ein Unterordner angelegt in dem sich eine .tas-Datei (Symbolik) und die .tay-Dateien (Daten) befinden. Hier kann der Pfad zum Ordner nutzerdefiniert bestimmt werden, in den das Analytics-Verzeichnis rein soll.

Max. unconfirmed: Gibt an, wie viele ADS-Requests sich während des Schreibens von Dateien ansammeln dürfen, für die keine entsprechende ADS-Confirmation vorliegt. Dieser Parameter ermöglicht ein Flow-Control, um ein overflow der ADS-Router queues zu verhindern. Er kann in der Regel auf dem voreingestellten Wert bleiben. Treten allerdings sehr viele Schreibvorgänge in kurzer Zeit auf, bei gleichzeitiger anderweitiger Belastung des Routers, kann es zu überfüllten Queues führen, was über Fehlermeldungen angezeigt wird.

Ring Buffer: Aktiviert einen Ringpuffer in dem die Binärdateien gepuffert werden. Bei jedem Neustart von TwinCAT wird der aktuelle Inhalt in einen Backup-Ordner übertragen und ein neuer Ringpuffer gestartet, wobei der Inhalt des Backup-Ordners überschrieben wird.

File Count: Mit dem Parameter File Count kann eine Anzahl von Dateien angegeben werden, die Teil des Ringpuffers sein sollen. Die Ringpufferzeit hängt von der angegebenen maximalen Dateigröße (Max File Size) ab.

7 API

7.1 SPS

7.1.1 Analytics Communication Library

7.1.1.1 Übersicht

Die TwinCAT Analytics Communication Library ist eine SPS-Bibliothek, die dem Nutzer eine Schnittstelle zum Analytics Logger und seinem Gegenstück, dem Analytics Stream Helper bereitstellt. Sie sind damit in der Lage während der Laufzeit aus dem SPS-Code heraus einen Analytics Stream zu starten, zu stoppen und umzukonfigurieren. Beispielsweise können Sie den Endpunkt, ein Verzeichnis oder einen Message Broker, aus der Applikation heraus ändern. Das eröffnet viele neue Anwendungsmöglichkeiten für Ihre Analytics Applikation.

Produkt Komponenten

- Treiber TcAnalytics.sys
- SPS-Bibliothek Tc3AnalyticsCommunication.compiled-library

7.1.1.2 Installation

Die TwinCAT Analytics Communication Library wird mit der TwinCAT XAE und XAR Version ≥ 4024.47 installiert. Daher sollte sie immer verfügbar sein. Um sie zu benutzen ist keine eigenständige Lizenz nötig, jedoch erfordert der Analytics Logger eine TF3500 TwinCAT 3 Analytics Logger Lizenz, entweder unbefristet oder als 7-Tage Lizenz.

7.1.1.3 SPS-API

7.1.1.3.1 Funktionsbausteine

7.1.1.3.1.1 FB_ALYC_MqttStream

Dieser Funktionsbaustein repräsentiert einen Analytics MQTT-Stream. Über die ObjectID als Eingangsvariable kann der Zusammenhang zu einem Stream hergestellt werden. Voraussetzung ist ein existierender Stream, z. B. im System Manager unter dem Data Logger Knoten im Projektbaum angelegt. Über die Methoden Start/Stop kann der Stream gesteuert werden und mit Reconfigure rekonfiguriert werden. Dazu wird der Methode eine Struktur des Typs ST_ALYC_MqttStreamConfig übergeben, die die neuen Konfigurationsparameter enthält. Bei der Rekonfiguration werden, ausgehend vom OP-Zustand, die TcCom-Zustände SAFEOP, PREOP, SAFEOP, OP in dieser Reihenfolge durchlaufen. Da alle Zustände unterhalb von SAFEOP nicht mehr im Echtzeit-Modus laufen, die restliche TwinCAT-Laufzeit aber schon, muss die Rekonfiguration asynchron zum Task-Zyklus erfolgen, wobei die Methode Reconfig zyklisch aufgerufen werden soll, solange nicht der OP-Zustand wieder erreicht wird. Die Eigenschaften bConnected, bStarted etc. geben Auskunft über den aktuellen Zustand des Streams. Fehler können über den Ausgang bError und die dazugehörige ipResultMessage erfasst werden.

Definition:

```
FUNCTION_BLOCK FB_ALYC_MqttStream
VAR_INPUT
    {attribute 'tcinitsymbol'}
    nObjectID : OTCID := 0;
END_VAR
VAR_OUTPUT
    bInitialized : BOOL := FALSE;
    bError : BOOL := FALSE;
    ipResultMessage : I_TcMessage := fbResult;
    eReconfigState : E_ALYC_ReconfigState := E_ALYC_ReconfigState.DONE;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nObjectID	OTCID	TcCom-Object ID des referenzierten Streams. Diese kann unter dem Reiter ‚Init-Symbole‘ des entsprechenden PLC-Instanz Knotens im System Manager Projektbaum initialisiert werden.

 **Ausgänge**

Name	Typ	Beschreibung
bInitialized	BOOL	TRUE, wenn der Funktionsbaustein initialisiert ist und benutzt werden kann. Die Initialisierung erfolgt automatisch nach dem Setzen von TwinCAT in den Run-Modus.
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Nachricht EventLogger
eReconfigState	E_ALYC_ReconfigState	Der Zustand der Zustandsmaschine während der Rekonfiguration.

 **Methoden**

Name	Rückgabe Typ	Beschreibung
Reconfigure	BOOL	Rekonfiguriere den Stream. Parameter: ST_ALYC_MqttStreamConfig. TRUE, wenn erfolgreich.
Start	BOOL	Starte den Stream. TRUE, wenn erfolgreich.
Stop	BOOL	Stoppe den Stream. TRUE, wenn erfolgreich.

 **Eigenschaften**

Name	Typ	Richtung	Beschreibung
bConnected	BOOL	get	TRUE, wenn MQTT-Verbindung besteht.
bStarted	BOOL	get	TRUE, wenn Stream gestartet.
nDataSize	UDINT	get	Sample Datengröße
nSamplesIssued	ULINT	get	Anzahl Samples, die geschrieben wurden.
nSamplesLost	ULINT	get	Anzahl der verworfenen Samples.
tCycleTime	LTIME	get	Zykluszeit in ns
nCompDataSaving	DINT	get	Prozentanteil der eingesparten Datenmenge im Vergleich zu der unkomprimierten Alternative. Wenn negativ, liegt ein Mehraufwand statt einer Ersparnis an Daten vor.

7.1.1.3.1.2 FB_ALYC_FileStream

Dieser Funktionsbaustein repräsentiert einen Analytics Stream im Filemodus. Über die ObjectID als Eingangsvariable kann der Zusammenhang zu einem Stream hergestellt werden. Voraussetzung ist ein existierender Stream, der im Filemodus ist, z. B. im System Manager unter dem Data Logger Knoten im Projektbaum angelegt. Die Symbole, die geloggt werden sollen, müssen per System Manager ebenfalls konfiguriert werden.

Über die Methoden Start/Stop kann der Stream gesteuert werden und mit Reconfigure rekonfiguriert werden. Dazu wird der Methode eine Struktur des Typs ST_ALYC_FileStreamConfig übergeben, die die neuen Konfigurationsparameter enthält. Bei der Rekonfiguration werden, ausgehend vom OP-Zustand, die TcCom-Zustände SAFEOP, PREOP, SAFEOP, OP in dieser Reihenfolge durchlaufen. Da alle Zustände unterhalb von SAFEOP nicht mehr im Echtzeit-Modus operieren, die restliche TwinCAT-Laufzeit aber schon, muss die Rekonfiguration asynchron zum Task-Zyklus erfolgen, wobei die Methode Reconfigure zyklisch aufgerufen

werden soll, solange nicht der OP-Zustand wieder erreicht ist. Die Eigenschaften bStarted, nSampleIssued etc. geben Auskunft über den aktuellen Zustand des Streams. Fehler können über den Ausgang bError und die dazugehörige ipResultMessage erfasst werden.

Definition:

```
FUNCTION_BLOCK FB_ALYC_FileStream
VAR_INPUT
  {attribute 'tcinitsymbol'}
  nObjectID : OTCID := 0;
END_VAR
VAR_OUTPUT
  bInitialized : BOOL := FALSE;
  bError : BOOL := FALSE;
  ipResultMessage : I_TcMessage := fbResult;
  eReconfigState : E_ALYC_ReconfigState := E_ALYC_ReconfigState.DONE;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
nObjectID	OTCID	TcCom-Object ID des referenzierten Streams. Diese kann unter dem Reiter ‚Init-Symbole‘ des entsprechenden PLC-Instanz Knotens im System Manager Projektbaum initialisiert werden.

Ausgänge

Name	Typ	Beschreibung
bInitialized	BOOL	TRUE, wenn der Funktionsbaustein initialisiert ist und benutzt werden kann. Die Initialisierung erfolgt automatisch nach dem Setzen von TwinCAT in den Run-Modus.
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Nachricht EventLogger
eReconfigState	E_ALYC_ReconfigState	Der Zustand der Zustandsmaschine während der Rekonfiguration.

Methoden

Name	Rückgabe Typ	Beschreibung
Reconfigure	BOOL	Rekonfiguriere den Stream. Parameter: ST_ALYC_FileStreamConfig. TRUE, wenn erfolgreich.
Start	BOOL	Starte den Stream. TRUE, wenn erfolgreich.
Stop	BOOL	Stoppe den Stream. TRUE, wenn erfolgreich.

 **Eigenschaften**

Name	Typ	Richtung	Beschreibung
bStarted	BOOL	get	TRUE, wenn Stream gestartet.
nDataSize	UDINT	get	Sample Datengröße
nSamplesIssued	ULINT	get	Anzahl Samples, die geschrieben wurden.
nSamplesLost	ULINT	get	Anzahl der verworfenen Samples.
tCycleTime	LTIME	get	Zykluszeit in ns
nCompDataSaving	DINT	get	Prozentanteil der eingesparten Datenmenge im Vergleich zu der unkomprimierten Alternative. Wenn negativ, liegt ein Mehraufwand statt einer Ersparnis an Daten vor.
nUnconfFileWrites	ULINT	get	Anzahl der Dateischreib-Anfragen, die noch nicht vom AMS Router bestätigt wurden. Kann ein Overflow der Router Nachrichten-Queue verhindern.

7.1.1.3.1.3 FB_ALYC_MqttStreamHelper

Dieser Funktionsbaustein repräsentiert einen Analytics Stream Helper im MQTT-Modus. Über die ObjectID als Eingangsvariable kann der Zusammenhang zu einem bestehenden Stream Helper hergestellt werden. Dieser muss im MQTT-Modus konfiguriert sein. Über die Methoden Start/Stop kann der Stream gesteuert werden und mit Reconfigure rekonfiguriert werden. Dazu wird der Methode eine Struktur des Typs ST_ALYC_MqttStreamHelperConfig übergeben, die die neuen Konfigurationsparameter enthält. Bei der Rekonfiguration werden, ausgehend vom OP-Zustand die TcCom-Zustände SAFEOP, PREOP, SAFEOP, OP in dieser Reihenfolge durchlaufen. Da alle Zustände unterhalb von SAFEOP nicht mehr im Echtzeit-Modus laufen, die restliche TwinCAT-Laufzeit aber schon, muss die Rekonfiguration asynchron zum Task-Zyklus erfolgen, wobei die Methode Reconfigure zyklisch aufgerufen werden soll, solange nicht der OP-Zustand wieder erreicht ist. Die Eigenschaften bConnected, bStarted etc. geben Auskunft über den aktuellen Zustand des Stream Helpers. Fehler können über den Ausgang bError und die dazugehörige ipResultMessage erfasst werden.

Definition:

```
FUNCTION_BLOCK FB_ALYC_MqttStreamHelper
VAR_INPUT
    {attribute 'tcinitsymbol'}
    nObjectID : OTCID := 0;
    nNumInputBuffer : UDINT := 20;
END_VAR
VAR_OUTPUT
    ipResultMessage : I_TcMessage := fbResult;
    bError : BOOL := FALSE;
    bNewResult : BOOL := FALSE;
    bInitialized : BOOL := FALSE;
    nNumElements : UDINT;
    eReconfigState : E_ALYC_ReconfigState := E_ALYC_ReconfigState.DONE;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nObjectID	OTCID	TcCom-Object ID des referenzierten StreamHelpers. Diese kann unter dem Reiter ‚Init-Symbole‘ des entsprechenden PLC-Instanz Knotens im System Manager Projektbaum initialisiert werden.
nNumInputBuffer	UDINT	Maximale Anzahl an gepufferten Symbol-Werten (Samples) in den Symbol Queues.

Ausgänge

Name	Typ	Beschreibung
bInitialized	BOOL	TRUE, wenn der Funktionsbaustein initialisiert ist und benutzt werden kann. Die Initialisierung erfolgt automatisch nach dem Setzen von TwinCAT in den Run-Modus.
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Nachricht EventLogger
eReconfigState	E_ALYC_ReconfigState	Der Zustand der Zustandsmaschine während der Rekonfiguration.
bNewResult	BOOL	TRUE, wenn neue Werte in die Symbol Queues eingelesen wurden.
nNumElements	UDINT	Anzahl an neuen Werten, in den Symbol Queues

Methoden

Name	Rückgabe Typ	Beschreibung
Reconfigure	BOOL	Reconfigure the stream helper with a ST_ALYC_MqttStreamHelperConfig as parameter. Rückgabe von TRUE, wenn erfolgreich.
Call	BOOL	Hauptmethode, die stets zyklisch aufgerufen werden sollte. TRUE, wenn erfolgreich.
AddlotSymbol	BOOL	Füge ein Symbol des Typs I_ALYC_lotSymbol zur internen Symbol-Liste hinzu, dessen Werte empfangen werden sollen. TRUE, wenn erfolgreich.
ReleaselotSymbol	BOOL	Entferne ein Symbol des Typs I_ALYC_lotSymbol von internen Symbol-Liste. TRUE, wenn erfolgreich.
ReleaseAlllotSymbols	BOOL	TRUE, wenn erfolgreich.
ContainslotSymbol	BOOL	TRUE, wenn Symbol des Typs I_ALYC_lotSymbol in der internen Symbol-Liste.

Eigenschaften

Name	Typ	Richtung	Beschreibung
bConnected	BOOL	get	TRUE, wenn eine MQTT-Verbindung besteht.
bReconnect	BOOL	get/set	Wenn TRUE, unterbreche die MQTT-Verbindung, wenn FALSE, erneuere die MQTT-Verbindung.
sStream	STRING(255)	get/set	MQTT-Empfangs-Topic im Format <MainTopic>/<StreamTopic>
nNumlotSymbolsRegistered	UDINT	get/set	Anzahl der hinzugefügten lot-Symbole.

7.1.1.3.1.4 FB_ALYC_FileStreamHelper

Dieser Funktionsbaustein repräsentiert einen Analytics Stream Helper im Filemodus. Über die ObjectId als Eingangsvariable kann der Zusammenhang zu einem bestehenden Stream Helper hergestellt werden. Dieser muss im Filemodus konfiguriert sein. Über die Methoden Start/Stop kann der Stream gesteuert werden und mit Reconfigure rekonfiguriert werden. Dazu wird der Methode eine Struktur des Typs ST_ALYC_FileStreamHelperConfig übergeben, die die neuen Konfigurationsparameter enthält. Bei der Rekonfiguration werden, ausgehend vom OP-Zustand die TcCom-Zustände SAFEOP, PREOP, SAFEOP, OP in dieser Reihenfolge durchlaufen. Da alle Zustände unterhalb von SAFEOP nicht mehr im Echtzeit-Modus laufen, die restliche TwinCAT-Laufzeit aber schon, muss die Rekonfiguration asynchron zum Task-Zyklus erfolgen, wobei die Methode Reconfigure zyklisch aufgerufen werden soll, solange nicht der OP-

Zustand wieder erreicht ist. Die Eigenschaften bStarted etc. geben Auskunft über den aktuellen Zustand des Stream Helpers. Fehler können über den Ausgang bError und die dazugehörige ipResultMessage erfasst werden.

Definition:

```
FUNCTION_BLOCK FB_ALYC_MqttStreamHelper
VAR_INPUT
  {attribute 'tcinitsymbol'}
  nObjectID : OTCID := 0;
  nNumInputBuffer : UDINT := 20;
END_VAR
VAR_OUTPUT
  ipResultMessage : I_TcMessage := fbResult;
  bError : BOOL := FALSE;
  bNewResult : BOOL := FALSE;
  bInitialized : BOOL := FALSE;
  nNumElements : UDINT;
  eReconfigState : E_ALYC_ReconfigState := E_ALYC_ReconfigState.DONE;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nObjectID	OTCID	TcCom-Object ID des referenzierten StreamHelpers. Diese kann unter dem Reiter ‚Init-Symbole‘ des entsprechenden PLC-Instanz Knotens im System Manager Projektbaum initialisiert werden.
nNumInputBuffer	UDINT	Maximale Anzahl an gepufferten Symbol-Werten (Samples) in den Symbol Queues.

 **Ausgänge**

Name	Typ	Beschreibung
bInitialized	BOOL	TRUE, wenn der Funktionsbaustein initialisiert ist und benutzt werden kann. Die Initialisierung erfolgt automatisch nach dem Setzen von TwinCAT in den Run-Modus.
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
ipResultMessage	I_TcMessage	Nachricht EventLogger
eReconfigState	E_ALYC_ReconfigState	Der Zustand der Zustandsmaschine während der Rekonfiguration.
bNewResult	BOOL	TRUE, wenn neue Werte in die Symbol Queues eingelesen wurden.
nNumElements	UDINT	Anzahl an neuen Werten, in den Symbol Queues
stCurrentConfig	ST_ALYC_FileStreamHelper Config_▶ 83]	
stCurrentState	ST_ALYC_FileStreamHelperState	Aktuelle Zustandsinformationen zum Stream Helper.

Methoden

Name	Rückgabe Typ	Beschreibung
Reconfigure	BOOL	Reconfigure the stream helper with a ST_ALYC_MqttStreamHelperConfig as parameter. Rückgabe von TRUE, wenn erfolgreich.
Call	BOOL	Hauptmethode, die stets zyklisch aufgerufen werden sollte. TRUE, wenn erfolgreich.
AddlotSymbol	BOOL	Füge ein Symbol des Typs I_ALYC_lotSymbol zur internen Symbol-Liste hinzu, dessen Werte empfangen werden sollen. TRUE, wenn erfolgreich.
ReleaselotSymbol	BOOL	Entferne ein Symbol des Typs I_ALYC_lotSymbol von internen Symbol-Liste. TRUE, wenn erfolgreich.
ReleaseAlllotSymbols	BOOL	TRUE, wenn erfolgreich.
ContainslotSymbol	BOOL	TRUE, wenn Symbol des Typs I_ALYC_lotSymbol in der internen Symbol-Liste.

Eigenschaften

Name	Typ	Richtung	Beschreibung
nNumlotSymbolsRegistered	UDINT	get/set	Anzahl der hinzugefügten lot Symbole.

7.1.1.3.1.5 IoT Symbol

7.1.1.3.1.5.1 FB_ALYC_lotSymbol_BOOL

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_BOOL
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_lotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.2 FB_ALYC_IotSymbol_BYTE

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_BYTE
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.3 *FB_ALYC_IotSymbol_DINT*

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_DINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.4 FB_ALYC_IotSymbol_DWORD

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_DWORD
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.5 *FB_ALYC_IotSymbol_INT*

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_INT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.6 FB_ALYC_IotSymbol_LINT

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_LINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsart	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.7 *FB_ALYC_IotSymbol_LREAL*

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_LREAL
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.8 FB_ALYC_IotSymbol_LWORD

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_LWORD
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.9 *FB_ALYC_IotSymbol_REAL*

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_REAL
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR

```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.10 FB_ALYC_IotSymbol_SINT

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_SINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.11 *FB_ALYC_IotSymbol_STRING*

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_STRING
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.12 FB_ALYC_IotSymbol_UDINT

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_UDINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.13 *FB_ALYC_IotSymbol_UINT*

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_UINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR

```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.14 FB_ALYC_IotSymbol_ULINT

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_ULINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.15 *FB_ALYC_IotSymbol_USINT*

Syntax

Definition:

```

FUNCTION_BLOCK FB_ALYC_IotSymbol_USINT
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR

```

Eingänge

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

 **Ausgänge**

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

 **Methoden**

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.1.5.16 FB_ALYC_IotSymbol_WORD

Syntax

Definition:

```
FUNCTION_BLOCK FB_ALYC_IotSymbol_WORD
VAR_INPUT
    stConfig : ST_ALYC_IotSymbol_Config;
END_VAR
VAR_OUTPUT
    ipResultMessage: I_TcMessage;
    bError: BOOL;
    bNewResult: BOOL;
    bConfigured: BOOL;
    bSymbolHandlerAssigned: BOOL;
    bVariableFound: BOOL;
    sSymbolPath: STRING(255);
    tCycleTime: LTIME;
    nMaxNumElements: UDINT;
    nNumElements: UDINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stConfig	ST_ALYC_IotSymbol_Config	Struktur für die Konfiguration des FB.

Ausgänge

Name	Typ	Beschreibung
ipResultMessage	I_TcMessage	Eventlogger
bError	BOOL	TRUE, wenn ein Fehler aufgetreten ist.
bNewResult	BOOL	TRUE, wenn ein neues Ergebnis berechnet worden ist.
bConfigured	BOOL	TRUE, wenn der FB erfolgreich konfiguriert ist.
bSymbolHandlerAssigned	BOOL	TRUE, wenn der Symbol-Handler zugewiesen wurde.
bVariableFound	BOOL	TRUE, wenn eine Variable im Stream gefunden wurde.
sSymbolPath	STRING(255)	Theam/Stream.Symbol
tCycleTime	LTIME	Zykluszeit des Publishing-Systems.
nMaxNumElements	UDINT	Maximale Anzahl der von StreamHelper beeinflussten gespeicherten Symbole.

Methoden

Name	Definitionsort	Beschreibung
GetValue	Local	Abrufen des Werts des spezifizierten Elements.
GetOversamplingValues	Local	Abrufen der Oversampling-Werte des spezifizierten Elements.
GetArrayValues	Local	Abrufen der Array-Werte des spezifizierten Elements.

Voraussetzungen

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

7.1.1.3.2 Datentypen

7.1.1.3.2.1 ST_ALYC_MqttStreamConfig

```

TYPE ST_ALYC_MqttStreamConfig :
STRUCT
    bAutoStartStream : BOOL := TRUE;
    nAdsBuffer : DINT := 32; // Samples in buffer
    nSamplingDivider : UDINT := 1;
    eCompressionMethod : ANALYTICS_COMPRESSION := ANALYTICS_COMPRESSION.ANALYTICS_COMPRESSION_RL;
    eCompressionWidth : ANALYTICS_COMPRESSION_WIDTH := ANALYTICS_COMPRESSION_WIDTH.ANALYTICS_COMP_W
IDTH_8;
    eExternalTimeType : ETcExternalTimeType := ETcExternalTimeType.SystemTime;
    stDeviceLocation: ST_ALYC_Address := (sAddress := '', sLongitude := '', sLatitude := '');
    sSystemIdAlias : STRING;
    stConnection : ST_ALYC_MqttConnectionSettings;
    nQueueSize : UDINT := 0;
    sMqttTopic : STRING(255) := ''; // Combination of main topic and stream topic: MainTopic/
StreamTopic
    bQueueWhenDisconnected : BOOL := FALSE;
    bQueueInFile : BOOL := FALSE;
END_STRUCT
END_TYPE

```

7.1.1.3.2.2 ST_ALYC_FileStreamConfig

```

TYPE ST_ALYC_FileStreamConfig :
STRUCT
    bAutoStartStream : BOOL := TRUE;
    nAdsBuffer : DINT := 32; // Samples in buffer
    nSamplingDivider : UDINT := 1;
    eCompressionMethod : ANALYTICS_COMPRESSION := ANALYTICS_COMPRESSION.ANALYTICS_COMPRESSION_RL;
    eCompressionWidth : ANALYTICS_COMPRESSION_WIDTH := ANALYTICS_COMPRESSION_WIDTH.ANALYTICS_COMP_W
IDTH_8;

```

```
eExternalTimeType : ETcExternalTimeType := ETcExternalTimeType.SystemTime;
stDeviceLocation : ST_ALYC_Address := (sAddress := '', sLongitude := '', sLatitude := '');
sSystemIdAlias : STRING;
nMaxFileSize : UDINT := 16;
nFilesInRingBuffer : UDINT := 2; // Number of files in ring buffer
sFileDir : STRING(255) := ''; // Optional, Default: %TC_BOOTPRJPATH%
bEnableRingBuffer : BOOL := TRUE;
nMaxUnconfWrites : UDINT := 1000; // Number of file writes
without a confirmation from ADS router
END_STRUCT
END_TYPE
```

7.1.1.3.2.3 ST_ALYC_MqttStreamHelperConfig

```
TYPE ST_ALYC_MqttStreamHelperConfig :
STRUCT
    stConnection : ST_ALYC_MqttConnectionSettings;
    sMqttTopic : STRING(255); // Combination of main topic and stream topic
    bAutostartReceive : BOOL := TRUE;
END_STRUCT
END_TYPE
```

7.1.1.3.2.4 ST_ALYC_FileStreamHelperConfig

```
TYPE ST_ALYC_FileStreamHelperConfig :
STRUCT
    nSamplesInReadBuf : UINT := 32; //How many samples to read from file every cycle
    nReadBufsInStreamBuf : UINT := 32; //How many read
    buffers to put into a stream buffer (treated as ring buffer)
    sFileDir : STRING(255) := ''; //Directory in which to find the Analytics data files
    bReadFilesCyclically : BOOL := TRUE; //If all files have been read, start over reading them
from the beginning
END_STRUCT
END_TYPE
```

7.1.1.4 Beispiele

7.1.1.4.1 Analytics Streams und MQTT Stream Helper.

Das Beispiel zeigt, wie ein MQTT-Stream, sowie ein Stream im File Mode konfiguriert werden können. Gleichzeitig wird ein Stream Helper konfiguriert, der die Daten des MQTT-Streams empfängt. Wie die Daten des File Streams ausgewertet werden können, zeigt das nächste Beispiel. Es werden die wichtigsten Funktionsbausteine vom Typ [FB_ALYC_MqttStream](#) [▶ 60], [FB_ALYC_FileStream](#) [▶ 61] und [FB_ALYC_MqttStreamHelper](#) [▶ 63] verwendet.

Das Beispiel steht hier zum Download bereit:

https://infosys.beckhoff.com/content/1031/tf3500_tc3_analytics_logger/Resources/14831744651.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.47	PC oder CX (x64, x86)	Tc3_AnalyticsCommunication

7.1.1.4.2 File Stream Helper

Das Beispiel zeigt, wie ein File-StreamHelper verwendet werden kann, um Analytics-Files einzulesen und in das SPS-Programm einzubinden. Es wird der Funktionsbaustein [FB_ALYC_FileStreamHelper](#) [▶ 64] verwendet. Die Analytics Files des vorhergehenden Beispiels können hier verwendet werden.

Das Beispiel steht hier zum Download bereit:

https://infosys.beckhoff.com/content/1031/tf3500_tc3_analytics_logger/Resources/14831745163.zip

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.47	PC oder CX (x64, x86)	Tc3_AnalyticsCommunication

7.1.2 Obsolete

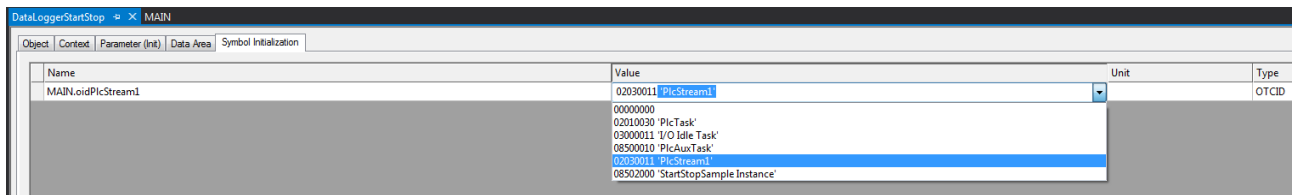
7.1.2.1 Verwendung der Programmierschnittstelle

Wie in der Technischen Einführung beschrieben, kann ein Stream mit Hilfe von Strukturiertem Text aus dem SPS-Code gestartet und gestoppt werden. Hierzu stellen die Streams eines Daten-Loggers, bei denen es sich um TcCom-Objekte handelt, eine Schnittstelle namens *ITcAnalyticsStream* mit zwei Methoden, *StartAnalyticsStream()* und *StopAnalyticsStream()* bereit. Befolgen Sie zur Verwendung der Schnittstelle die folgenden Schritte, wobei auch die Codebeispiele hilfreich sein können:

Deklarieren Sie eine Variable vom Typ *ITcAnalyticsStream* und eine andere vom Typ *OTCID* für die Objekt-ID des entsprechenden Streams. Für Diagnosezwecke empfiehlt sich eine *HRESULT*-Variable.

```
HR : HRESULT := S_OK;
{attribute 'tcinitsymbol'}
oidPlcStream1 : OTCID;
ipPlcStream1 : ITcAnalyticsStream;
```

Als Nächstes fügen Sie das Attribut 'tcinitsymbol' über der OTCID-Variablen hinzu. Auf diese Weise muss es nicht statisch im Quellcode initialisiert werden, sondern wird stattdessen zur Konfigurationszeit initialisiert, indem der SPS-Instanzknoten im Projektbaum doppelt angeklickt wird und der jeweilige Stream in der Combobox wie im folgenden Bild dargestellt ausgewählt wird.



Anschließend erstellen Sie einen Interface Pointer mit Hilfe der TcCom-Objekt-Servermethode und der IID der Schnittstelle.

```
IF ipPlcStream1 = 0 AND oidPlcStream1 <> 0 THEN
  HR := FW_ObjMgr_GetObjectInstance(oidPlcStream1, IID_ITcAnalyticsStream, ADR(ipPlcStream1));
END_IF
```

Nun können Sie mit dem Interface Pointer die Methoden der Schnittstelle aufrufen, wie im folgenden Beispiel dargestellt:

```
IF ipPlcStream1 <> 0 THEN
  IF bStartPlcStream1 THEN
    ipPlcStream1.StartAnalyticsStream();
    bStartPlcStream1 := FALSE;
  END_IF
  IF bStopPlcStream1 THEN
    ipPlcStream1.StopAnalyticsStream();
    bStopPlcStream1 := FALSE;
  END_IF
END_IF
```

Der Stream wird in demselben Zyklus gestartet, in dem *StartAnalyticsStream()* aufgerufen wird, und enthält die geloggtten Variablenwerte. Der Stream wird in demselben Zyklus gestoppt, in dem *StopAnalyticsStream()* aufgerufen wird, enthält jedoch *nicht* die Variablenwerte dieses Zyklus.

Im Abschnitt „Beispiele“ dieser Dokumentation finden Sie ein Beispielprogramm, das die hier dargestellten Code-Ausschnitte enthält.

7.2 Automation Interface

Beachten Sie dazu bitte die Dokumentation [Automation Interface: Erstellen von und Umgang mit Data Logger und Stream Helper](#)

8 Beispiele

Beispiel „Starten/Stoppen Daten-Logger Stream aus SPS-Code“:

https://infosys.beckhoff.com/content/1031/tf3500_tc3_analytics_logger/Resources/6904617099.zip

9 Anhang

9.1 FAQ – Häufig gestellte Fragen und Antworten

In diesem Abschnitt werden häufig gestellte Fragen beantwortet, um Ihnen die Arbeit mit dem TwinCAT Analytics Logger zu erleichtern. Falls Sie weitere Fragen haben, wenden Sie sich bitte an unser Support-Team support@beckhoff.com.

Sollte ich immer TLS mit MQTT verwenden? [► 86]

Ist es möglich, mehrere Verbindungen zu haben? [► 86]

Ist es möglich, den Analytics Logger durch einen SPS-Funktionsbaustein zu steuern? [► 86]

Sollte ich immer TLS mit MQTT verwenden?

!Ja, das sollten Sie, wenn Sie können. Wenn Sie sich den Overhead in CPU und Bandbreite leisten können, dann ist ein sicherer Kommunikationskanal überaus wertvoll. Je nach allgemeiner CPU-Leistung könnte eine merkliche Verringerung der Kommunikationsleistung möglich sein.

Ist es möglich, mehrere Verbindungen zu haben?

!Ja, Sie können den Analytics Logger gleichzeitig mit verschiedenen Message-Brokern verbinden, indem Sie einfach eine neue Instanz des Loggers hinzufügen. Es ist auch möglich, eine Instanz des Loggers für eine MQTT Kommunikation an einen Message-Broker und gleichzeitig eine Instanz zum Schreiben von Daten in die Analytics File des lokalen Systems zu haben.

Ist es möglich, den Analytics Logger durch einen SPS-Funktionsbaustein zu steuern?

!Es gibt keinen speziellen Funktionsbaustein für die Steuerung des Analytics Logger. Sie können jedoch eine Schnittstelle des Analytics Logger verwenden, um ihn mit einfachen Befehlen wie Start und Stopp aus der SPS zu steuern. Die Vorgehensweise finden Sie in [diesem \[► 84\]](#) Kapitel.

Mehr Informationen:
www.beckhoff.com/tf3500

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

