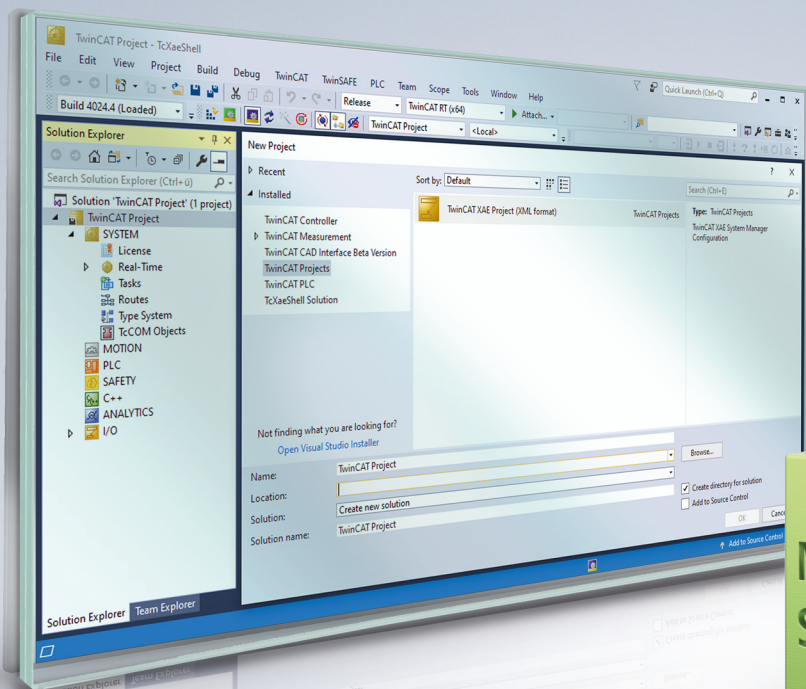


**BECKHOFF** New Automation Technology

Handbuch | DE

# TwinCAT 3

MATLAB®/Simulink®





# Inhaltsverzeichnis

<b>1</b>	<b>Vorwort</b> .....	<b>5</b>
1.1	Hinweise zur Dokumentation .....	5
1.2	Zu Ihrer Sicherheit.....	6
1.3	Hinweise zur Informationssicherheit .....	7
1.4	Ausgabestände dieser Dokumentation .....	8
<b>2</b>	<b>Übersicht</b> .....	<b>9</b>
<b>3</b>	<b>Blockdiagramm</b> .....	<b>11</b>
3.1	Simulink®-TcCOM .....	12
3.1.1	Bedienung des Blockdiagramms.....	12
3.1.2	Anzeigen von Signalverläufen.....	13
3.1.3	Modul-Parametrierung im Blockdiagramm.....	14
3.1.4	Debuggen.....	16
3.2	MATLAB®-TcCOM.....	20
3.2.1	Bedienung des Blockdiagramm-Fensters .....	21
3.2.2	Anzeige von Signalverläufen.....	22
3.3	Einbinden des Blockdiagramm-Controls .....	24
<b>4</b>	<b>TwinCAT Automation Interface: Verwendung in MATLAB®</b> .....	<b>26</b>
4.1	Beispiel: Tc3AutomationInterface .....	26



# 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®

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

### Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

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

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

## 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>.

## 1.4 Ausgabestände dieser Dokumentation

Version	Änderungen
1.4.x	NEU: <ul style="list-style-type: none"><li>• Versionsnummer des Block Diagram einsehen.</li><li>• Informationen zur Installation des TwinCAT 3.1 Build 4026, siehe <a href="#">Blockdiagramm</a> [► 11].</li></ul>



## 2 Übersicht

### MATLAB® und Simulink®

MATLAB® und Simulink® haben sich, auch bei angehenden Ingenieurinnen und Ingenieuren, global zu etablierten Umgebungen für verschiedenste Anwendungen entwickelt. Die Gründe dafür sind vielfältig. MATLAB® und Simulink® liefern Umgebungen, in denen man sich ganz auf die Engineering-Aufgabe konzentrieren kann. Das ist ideal für didaktische Konzepte in der Lehre und effizient in industriellen Anwendungen. MATLAB® liefert mit seinen zahlreichen Toolboxen eine ideale Umgebung zur Entwicklung von Algorithmen und zur Analyse von Daten. MATLAB® bietet zahlreiche Funktionen zum einfachen Zugriff auf unterschiedliche Datenformate und harmonisiert mit den unterschiedlichen Data-Logging-Mechanismen von TwinCAT. Simulink® ist fokussiert auf die durchgängige Unterstützung von Model Based Design (MBD). Hierbei wird anhand eines Systemmodells entwickelt, getestet und verifiziert. Die anschließende, automatische Code-Generierung für Plattformen wie TwinCAT stellt eine ideale Lösung dar, um den getesteten Code in der Produktion anzuwenden. Simulink® stellt alle Mittel zur Modellierung von Multi-Physik-Simulationen und Erstellung von Steuerungs-, Regelungs- und KI-Algorithmen bereit. Somit kommt auf Ihren Steuerungen nur qualitativ hochwertiger und an Modellen getesteter Code zum Einsatz.

### Technologien und Produkte für TwinCAT 3

#### TE1400 TwinCAT 3 Target for Simulink®

Mit dem TwinCAT 3 Target for Simulink® ist es möglich, in Simulink® entwickelte Modelle in TwinCAT 3 nutzbar zu machen. Dabei können in Simulink® diverse Toolboxen, z. B. SimScape™, Stateflow™ oder DSP System Toolbox™ eingebunden werden. Auch eingebettete MATLAB®-Funktionsbausteine werden unterstützt. Die Modelle werden automatisch mithilfe des Simulink Coder™ in C/C++-Code übersetzt und mit dem TwinCAT 3 Target for Simulink® in echtzeitfähige TwinCAT-Objekte überführt. Aus Simulink® erstellte TwinCAT-Objekte tragen dieselben Interfaces und Eigenschaften wie alle anderen TwinCAT-Objekte. Sie lassen sich vollständig im TwinCAT 3 Engineering verwenden, z. B. mit SPS-Quellcode zu einem Gesamtprojekt erweitern, debuggen und mit Feldbusteilnehmern verknüpfen, siehe Dokumentation [TE1400 TwinCAT 3 Target for Simulink®](#).

#### TE1401 TwinCAT 3 Target for MATLAB®

Mit TwinCAT 3 Target for MATLAB® können MATLAB®-Funktionen in TwinCAT 3 genutzt werden. Die Funktionen werden automatisch in TwinCAT-Objekte überführt und fließend im TwinCAT 3 Engineering verwendet. Die automatisch generierten Module können einerseits als TcCOM-Objekt und andererseits als SPS-Funktionsbaustein, auch als Bestandteil einer eigens erstellen SPS-Bibliothek, in die TwinCAT-Solution eingebunden werden. Die eingefügten Module werden mit dem gesamten TwinCAT-Projekt in die TwinCAT-3-Laufzeit heruntergespielt und dort, wie alle anderen Objekte, innerhalb der Echtzeitumgebung ausgeführt, siehe Dokumentation [TE1401 TwinCAT 3 Target for MATLAB®](#).

#### TE1410 TwinCAT 3 Interface for MATLAB®/Simulink®

Mit TwinCAT 3 Interface for MATLAB®/Simulink® stellen Sie eine performante bidirektionale Kommunikation zwischen der TwinCAT Runtime und MATLAB® oder Simulink® her. In der Engineeringphase einer Anlage können Sie das Werkzeug z. B. zur Software-in-the-Loop-Simulation mit Simulink® nutzen. Während der Maschinenlaufzeit realisieren Sie mit diesem Produkt beispielsweise:

- Ein MATLAB® App-basiertes Human-Maschine-Interface (HMI)
- Einen MATLAB® Compiler Runtime basierten Compute Server für z. B. prädiktive Wartung oder Parameteroptimierung

[Zur Dokumentation vom TwinCAT 3 Interface for MATLAB®/Simulink®](#)

### TwinCAT Basistechnologien für MATLAB® und Simulink®

Die im Folgenden vorgestellten Tools sind lizenzkostenfrei und Bestandteil des TwinCAT 3 XAE bzw. Full Setups.

#### TwinCAT 3 Block Diagram

Der Block Diagram Viewer ermöglicht die Anzeige, das Debugging und die Parametrierung eines TcCOM-Moduls über ein im TwinCAT Engineering System dargestelltes Block Diagram. Dieser Reiter ist nur für TcCOM-Module verfügbar, welche mit dem Target for MATLAB® oder Target for Simulink® generiert wurden. Der Export des Block Diagrams ist bei beiden Targets optional.

[Zur Dokumentation von TwinCAT 3 Block Diagram \[► 11\]](#)

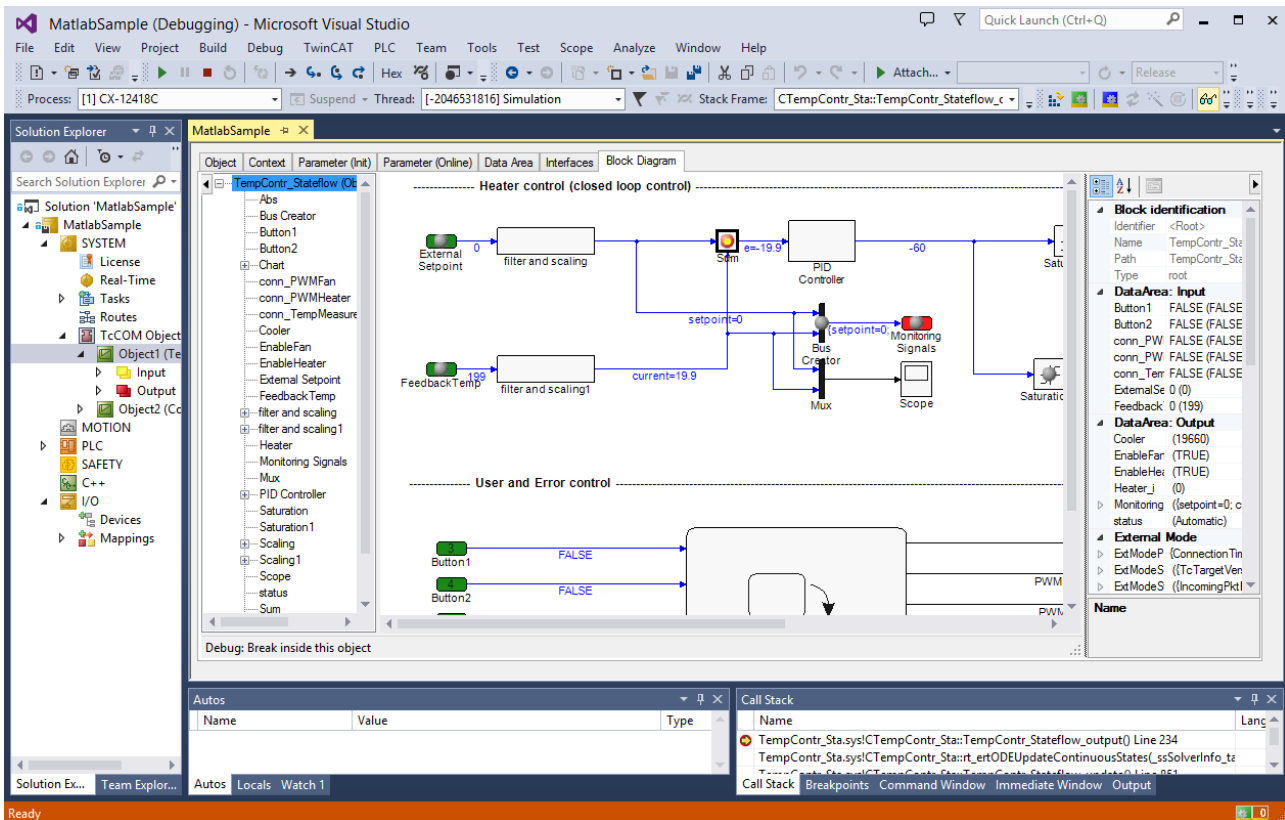
### **TwinCAT 3 Automation Interface**

Das TwinCAT 3 Automation Interface ist eine Programmierschnittstelle für TwinCAT Projekte, d. h. Sie können eine Konfiguration im System Manager anlegen, eine PLC erstellen und Code einfügen sowie TcCOM-Objekte instanziiieren, ohne TwinCAT über Visual Studio oder das TwinCAT XAE zu öffnen. Diese API ist unter anderem auch für MATLAB® verfügbar.

[Zur Dokumentation von TwinCAT 3 Automation Interface mit MATLAB® \[► 26\]](#)

### 3 Blockdiagramm

Bei der Generierung eines TwinCAT-Objekts aus MATLAB® oder Simulink® kann optional das Blockdiagramm (Target for Simulink®) oder der MATLAB®-Code (Target for MATLAB®) mit exportiert werden. In diesem Fall kann das Blockdiagramm oder der MATLAB®-Code in der TwinCAT-Entwicklungsumgebung unter dem Karteireiter **Block Diagram** der TcCOM-Instanz angezeigt werden:



Mithilfe dieses Controls ist es nicht nur möglich, durch die gesamte Struktur des Blockdiagramms zu navigieren oder den zugrundeliegenden MATLAB®-Code zu betrachten, sondern auch Parameterwerte einzusehen und zu ändern, Signalwerte und -Verläufe darzustellen sowie im Debugging Mode auch mithilfe von Breakpoints durch das Modul zu debuggen. Das Control ist so gestaltet, dass es auch in einer eigenen Visualisierung verwendet werden kann.

Die Versionsnummer des Block Diagram können Sie durch Rechtsklick in das Control und Auswahl **About TC3 BlockDiagram** einsehen. Alternativ können Sie auch im **Control Panel** von Windows unter **Programs and Features** die Versionsnummer einsehen (Eintrag: Beckhoff TwinCAT 3 BlockDiagram).

#### Installation

##### TwinCAT 3.1 Build 4024

Das TwinCAT Block Diagram Setup wird mit dem TwinCAT 3 XAE oder Full Setup ausgeführt und muss nicht separat ausgeführt werden. Ebenso wird das TwinCAT Block Diagram Setup über das TE14xx TwinCAT Tools for MATLAB® mitgeliefert.

##### TwinCAT 3.1 Build 4026

Installieren Sie über den TwinCAT Package Manager folgenden Workload:



### TwinCAT Block Diagram Classic

block diagram view for TcCom modules generated from Simulink or via FMI interface



Command line: `tcpkg install TwinCAT.XAE.BlockDiagramClassic`

## 3.1 Simulink®-TcCOM

Wenn mit dem TwinCAT Target for Simulink® ein TwinCAT-Objekt erzeugt wurde und der Blockdiagramm Export dabei ausgeführt wurde, kann das Block Diagramm des Simulink®-Modells als Control in der TwinCAT XAE dargestellt werden.

### 3.1.1 Bedienung des Blockdiagramms

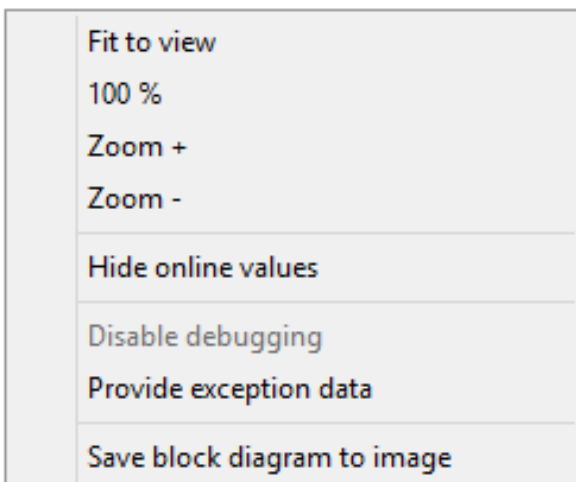
Bei der Generierung eines TcCOM-Moduls aus MATLAB® oder Simulink® kann der Export des Blockdiagramms konfiguriert werden. Wenn der Export aktiviert wurde, findet man das Blockdiagramm in der TwinCAT-Entwicklungsumgebung unter dem Karteireiter „Block Diagram“ der Modul-Instanz.

Unter Verwendung von Shortcuts, Drag&Drop sowie einem Kontextmenü, kann man durch die Hierarchie des TcCOM-Moduls navigieren, Parameterwerte ansehen, Signalwerte darstellen und optional zusätzliche Debug-Informationen erhalten.

#### Shortcut-Funktionen:

Shortcut	Funktion
Space	Zoom auf die aktuelle Größe des Blockdiagramm-Reiters
Backspace	Wechseln auf die nächst höhere Hierarchiestufe
ESC	Wechseln auf die nächst höhere Hierarchiestufe
STRG + "+"	Herein zoomen
STRG + "-"	Heraus zoomen
F5	Attach Debugger (System- > Real-Time -> C++ Debugger -> Enable C++ Debugger muss aktiviert sein)

#### Kontextmenü-Funktionen:

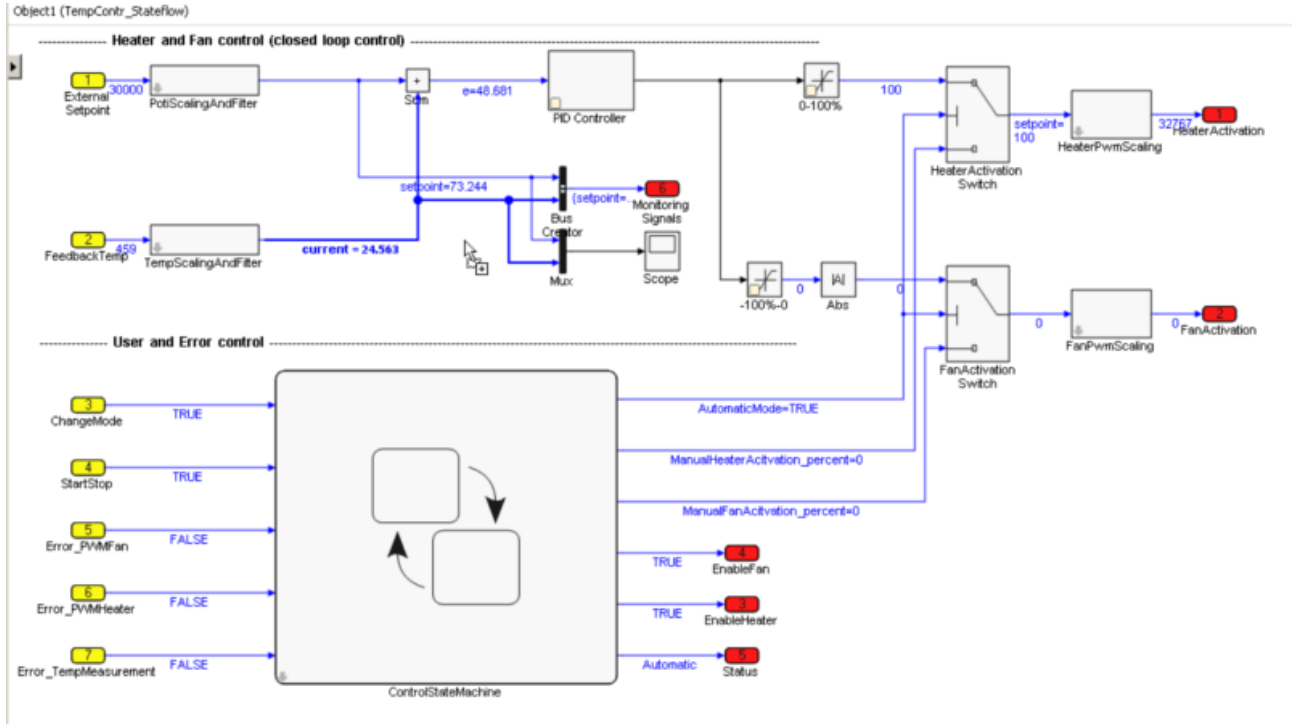


### 3.1.2 Anzeigen von Signalverläufen

Oft ist es hilfreich, sich zur Verifikation und Fehlersuche Signalverläufe anzeigen zu lassen. Das Blockdiagramm bietet hierzu folgende Möglichkeiten:

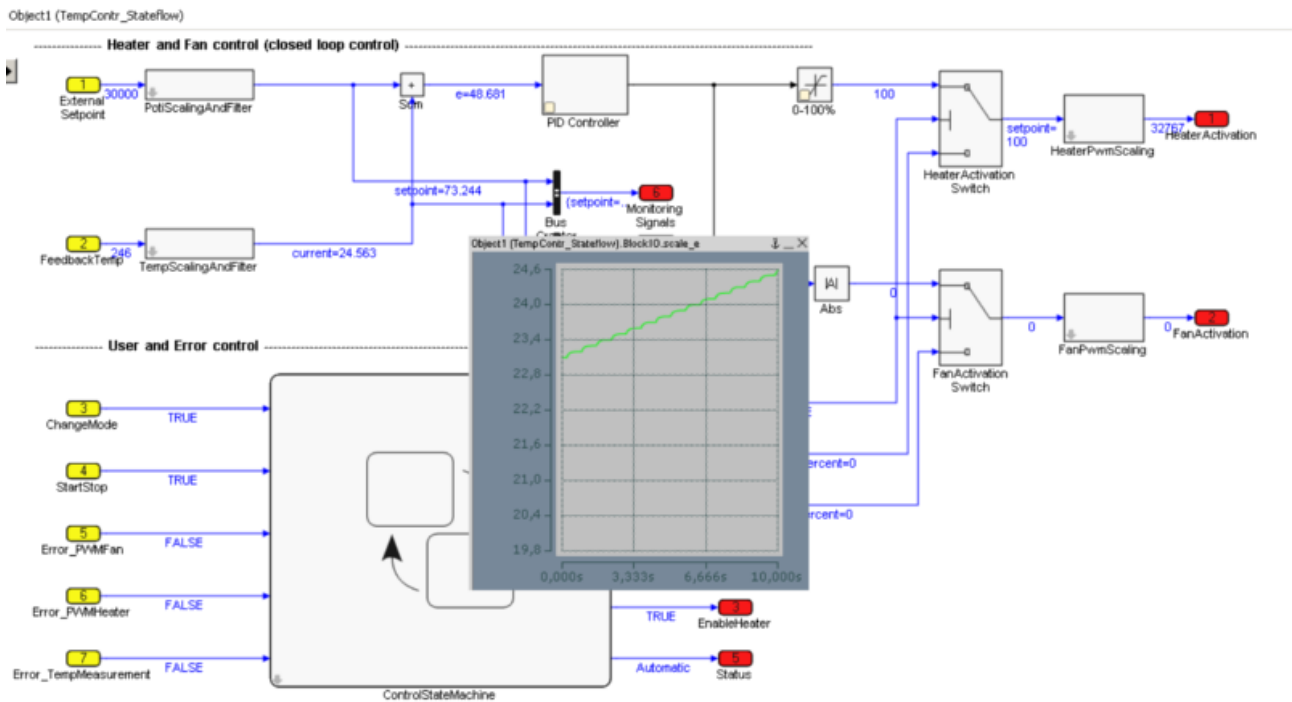
#### Anzeigen von Signalverläufen im Blockdiagramm

Das Blockdiagramm bietet die Möglichkeit, Signalverläufe in einem Fenster anzuzeigen. Hierzu wird ein Signal oder Block durch Drag-and-Drop auf einen freien Bereich des Blockdiagramms gezogen.






Erstellen eines Scopes im Blockdiagramm

Nach dem Drop öffnet sich ein Scope-Fenster im Blockdiagramm.



Anzeige des Scopes im Blockdiagramm

Die Titelleiste vom Scope-Fenster bietet folgende Optionen:

	Fenster schließen
	Fenster über alle Blockdiagrammhierarchien im Vordergrund halten
	Fenster auf die Titelleiste minimieren



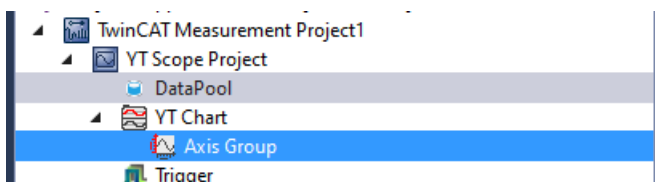
Für die Darstellung des Scopes im [Blockdiagramm-Control](#) [► 24] wird eine Lizenz des Scope View Professional (TE1300) benötigt. Im TwinCAT XAE ist keine Scope View Professional Lizenz notwendig.

Beim Erstellen eines Scope-Fensters im Blockdiagramm für einen Simulink®-Bus werden direkt alle Signale des Bus im Scope-Fenster dargestellt.

Es bietet sich an, das Scope-Fenster im Blockdiagramm für einen schnellen Überblick zu verwenden. Für genauere Analysen empfiehlt es sich, die Signale in einem TwinCAT Measurement-Projekt zu analysieren.

### Anzeigen von Signalverläufen im TwinCAT 3 Scope

Erfolgt der Drop nicht auf das Blockdiagramm Control sondern auf eine Axis Group in einem TwinCAT Measurement Projekt, wird das Signal dort hinzugefügt.



Hinzufügen eines Signals in ein TwinCAT 3 Scope

## 3.1.3 Modul-Parametrierung im Blockdiagramm

Zur Parametrierung einer TcCOM-Instanz kann das **Parameter-Fenster** direkt im Blockdiagramm verwendet werden. Außerdem kann die Eigenschaftstabelle genutzt werden, die am rechten Rand des Blockdiagramms ein- und ausgeklappt werden kann. Grundsätzlich wird zwischen unterschiedlichen Parameter-Werten unterschieden:

### „Default“, „Startup“, „Online“ und „Prepared“

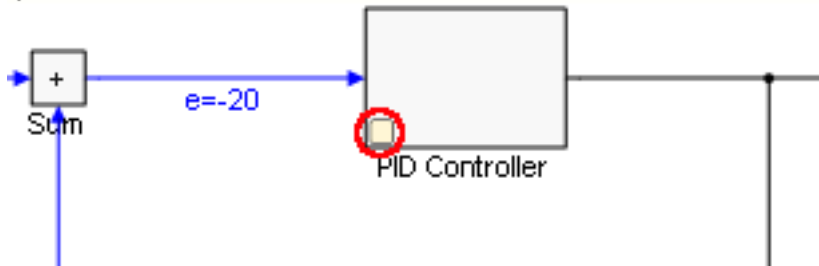
Im Drop-down-Menü der Eigenschaftstabelle des Blockdiagramms findet man folgende Wertetypen:

- **Default-Werte** sind die Parameterwerte beim Generieren des Codes. Sie sind unveränderlich in der Modulbeschreibungsdatei gespeichert und ermöglichen es, nach Parameteränderungen die "manufacturing settings" wiederherzustellen.
- **Startup-Werte** werden in der TwinCAT-Projektdatei gespeichert und in die Modulinstanz heruntergeladen, sobald TwinCAT die Modulinstanz startet. In Simulink®-Modulen können auch Startwerte für das Eingangs-Prozessabbild festgelegt werden. Dadurch kann das Modul mit Eingangswerten ungleich Null gestartet werden, ohne dass die Eingänge mit anderen Prozessabbildern verknüpft werden müssen. Interne Signale und Ausgangssignale haben keine Startwerte, da sie in jedem Fall im ersten Zyklus überschrieben würden.
- **Onlinewerte** sind nur verfügbar, wenn das Modul auf dem Zielsystem gestartet wurde. Sie zeigen den aktuellen Wert des Parameters im laufenden Modul. Dieser kann auch während der Laufzeit geändert werden. Das entsprechende Eingabefeld muss dazu allerdings erst über das Kontextmenü freigeschaltet werden, um versehentliche Eingaben zu vermeiden.

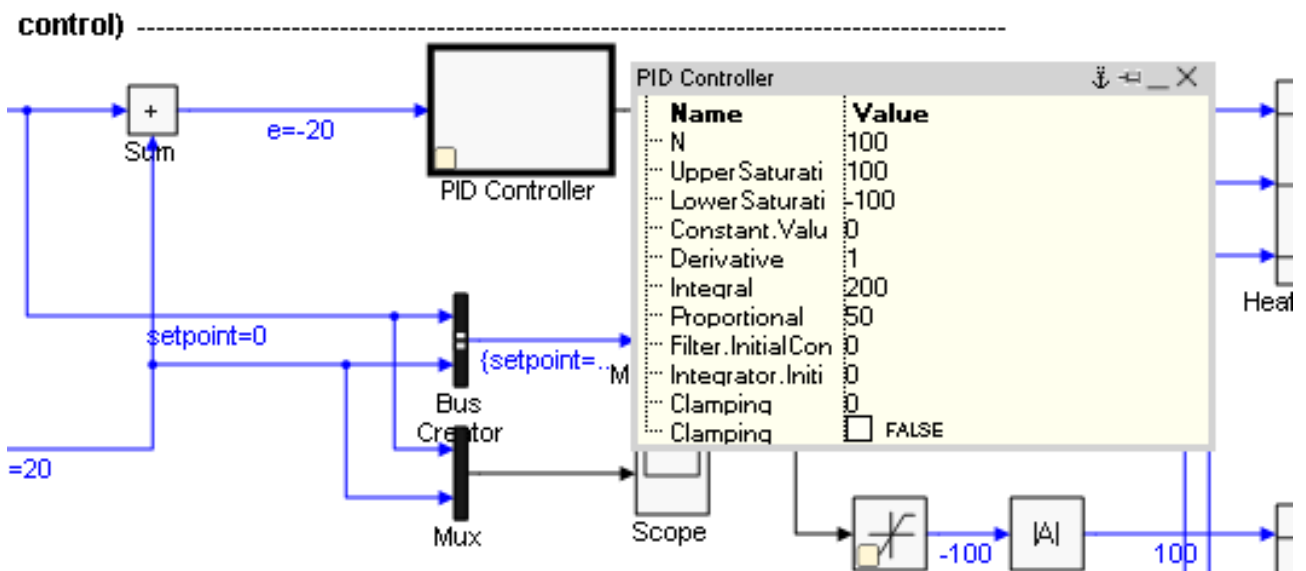
- **Prepared-Werte** können immer dann festgelegt werden, wenn auch Onlinewerte verfügbar sind. Mit ihrer Hilfe können verschiedene Werte gespeichert werden, um sie konsistent in das Modul zu schreiben. Wenn vorbereitete Werte festgelegt wurden, sind diese in einer Tabelle unterhalb des Blockdiagramms zu sehen. Mit den Schaltflächen rechts neben der Liste können die vorbereiteten Werte als Onlinewert heruntergeladen und/oder als Startwert gespeichert oder auch gelöscht werden.

**Parametrieren im Blockdiagramm**

Parametrierbare Blöcke werden im Blockdiagramm mit einem gelben Kasten markiert.



Durch Doppelklick auf den Block oder durch einen einzelnen Klick auf den gelben Kasten wird ein Fenster mit den veränderbaren Parametern angezeigt.



Wird ein Wert geändert, kann dieser mit folgenden Tastenbefehlen übernommen werden:

STRG + Enter	Onlinewert direkt setzen
SHIFT + Enter	Startup-Wert setzen
Enter	Prepared-Wert setzen

Die Symbole in der Titelleiste haben folgende Funktionen:

	Fenster schließen
	Fenster über alle Blockdiagrammhierarchieebenen im Vordergrund halten
	Fenster auf der aktuellen Blockdiagrammhierarchieebene offen halten
	Fenster auf Titelleiste minimieren

### 3.1.4 Debuggen

Um Fehler innerhalb eines mit MATLAB®/Simulink® erstellten TcCOM-Moduls zu finden bzw. das Verhalten des Moduls in der Gesamtarchitektur des TwinCAT-Projekts zu analysieren, stehen unterschiedliche Wege zur Verfügung.

#### Debuggen im Blockdiagramm

Wurde bei der Generierung des TcCOM-Moduls das Blockdiagramm exportiert, kann dieses in der TwinCAT-Entwicklungsumgebung angezeigt und unter anderem zum Debuggen innerhalb der entsprechenden Modulinstanz verwendet werden. Dazu nutzt das Blockdiagramm den Microsoft Visual Studio Debugger, der über den TwinCAT Debugger-Port mit der TwinCAT-Laufzeit verbunden werden kann. Das Verbinden (Attachen) des Debuggers erfolgt wie im C++-Bereich unter Debuggen beschrieben.

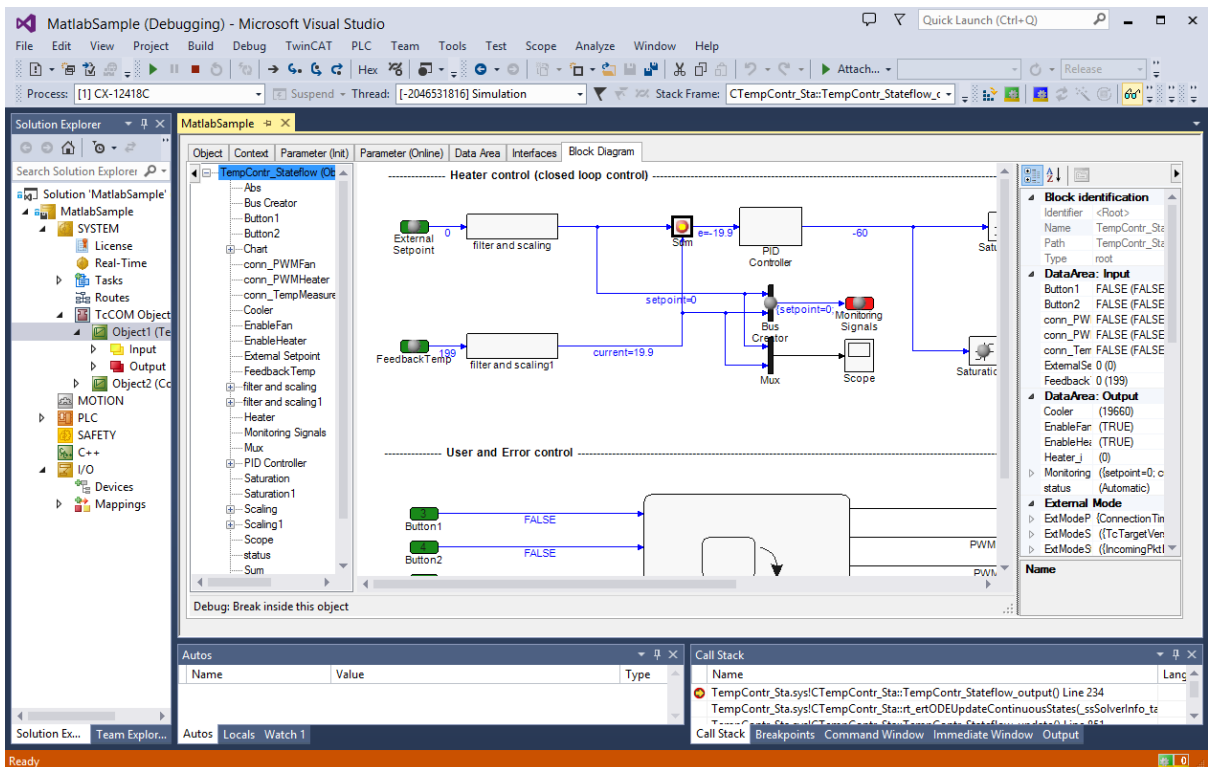
Voraussetzungen für das Debuggen innerhalb des Blockdiagramms sind:

- Der C/C++-Quellcode des TcCOM-Moduls muss auf dem Engineering-System vorhanden sein und vom Visual Studio-Debugger gefunden werden können. Idealerweise sollte das Debuggen daher auf dem System stattfinden, auf dem auch die Codegenerierung ausgeführt wurde. Wurde das Modul auf einem anderen System erstellt, kann der zugehörige C/C++-Quellcode i.d.R. durch Einbindung des Visual Studio Projektes in den C++ Bereich von TwinCAT bekannt gemacht werden. Die Datei <ModelName>.vcxproj liegt im build-Verzeichnis, siehe Welche Dateien werden automatisch bei der Codegenerierung und dem Publish erstellt?
- Das Modul muss mit der Konfiguration **Debug** erstellt worden sein. Beim Publish direkt im Anschluss an die Codegenerierung, muss im Bereich Publish-Mechanismus unter **publish configuration** die Einstellung **Debug** ausgewählt werden. Beim publish des Moduls aus dem C++ Bereich in TwinCAT muss der Debugger im C++ Knoten der Solution freigegeben sein, siehe Dokumentation C/C++ - Debuggen.
- Bei der Codegenerierung müssen in den Coder-Einstellungen unter **Tc Advanced** die Optionen **Export block diagram** und **Export block diagram debug information** aktiviert sein.
- Im TwinCAT-Projekt muss der Debugger-Port aktiviert sein, wie unter TwinCAT 3 C++ Enable C++ debugger beschrieben.

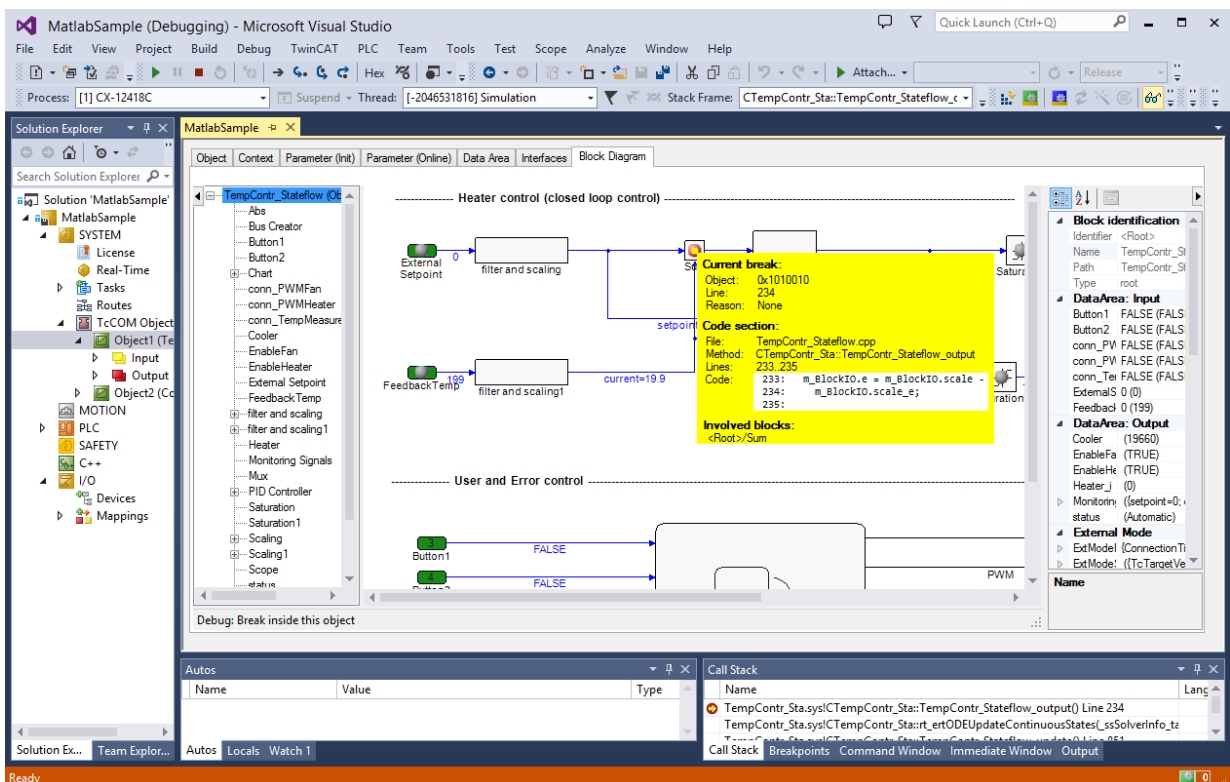
#### Setzen von Breakpoints im Blockdiagramm

1. Nach dem Verbinden (Attachen) des Debuggers mit der TwinCAT-Laufzeit, werden die möglichen Breakpoints im Blockdiagramm den Blöcken zugeordnet und als Punkte dargestellt. Durch Anklicken des gewünschten Breakpoints kann dieser aktiviert werden, um die Ausführung der Modulinstanz bei der nächsten Abarbeitung des zugehörigen Blocks zu stoppen. Die Farbe des Punktes gibt Auskunft über den aktuellen Zustand des Breakpoints:
  - Grau: Breakpoint inaktiv
  - Rot: Breakpoint aktiv. Bei der nächsten Abarbeitung dieses Blocks wird der Programmablauf angehalten
  - Gelber Punkt in der Mitte: Breakpoint Hit. Die Programmabarbeitung ist im Augenblick an dieser Stelle angehalten
  - Blauer Punkt in der Mitte: Breakpoint Hit (wie gelb), allerdings in einer anderen Instanz des Moduls.





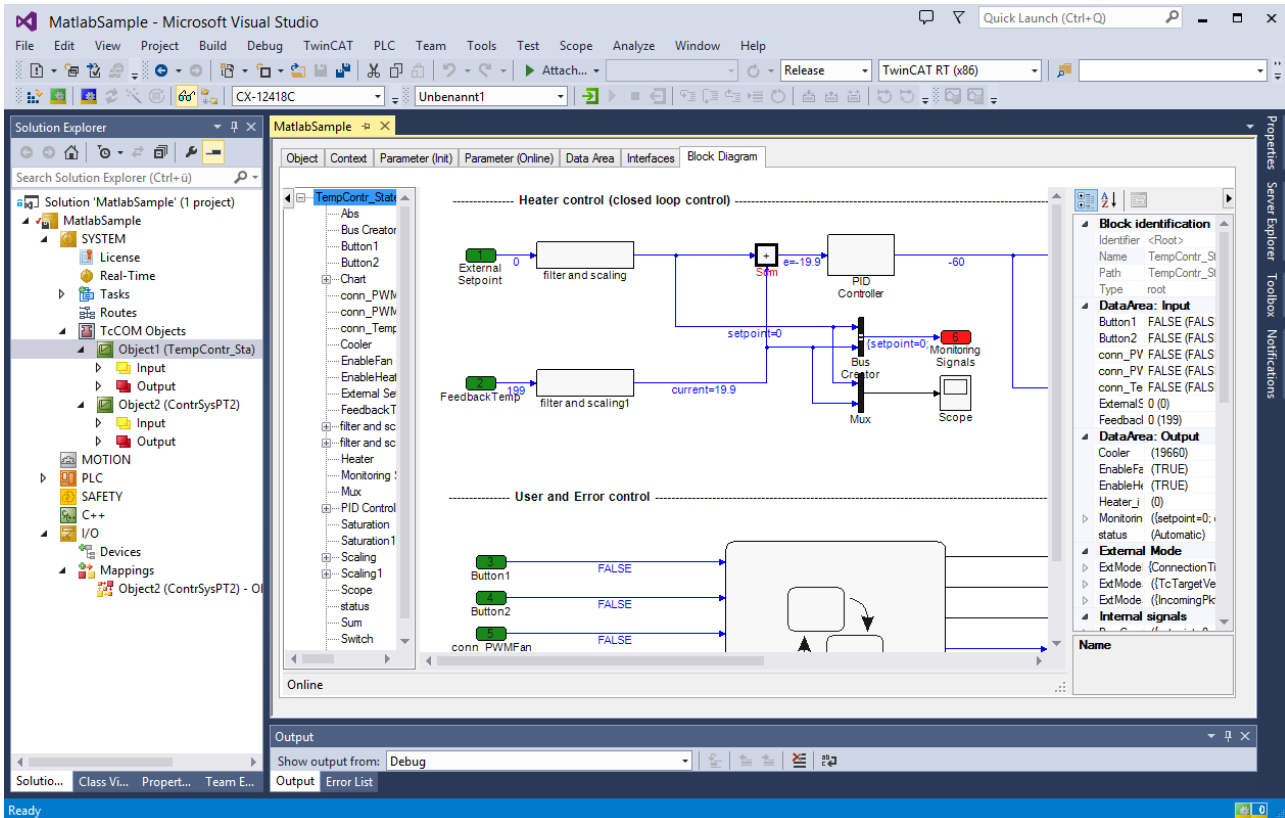
2. Im Tool-Tip des Breakpoints findet man zusätzliche Informationen, wie z. B. den zugehörigen C++-Codeabschnitt:



**i** Breakpoints werden nicht immer einem einzelnen Block zugeordnet. Im zugrundeliegenden C++-Code sind häufig Funktionalitäten mehrerer Blöcke in einem Codeabschnitt oder sogar einer Zeile zusammengefasst. Weil sich daher oft mehrere Blöcke den gleichen Breakpoint teilen, ändert sich bei der Aktivierung eines Breakpoints im Blockdiagramm häufig auch die Darstellung der Punkte an anderen Blöcken.

**Auswertung von Exceptions**

Treten während der Abarbeitung eines TcCOM-Modules Exceptions, wie z.B. eine Division durch Null, auf, so kann die Stelle an der diese Exception verursacht wurde im Blockdiagramm dargestellt werden. Dazu muss das TcCOM-Modul die oben genannten Voraussetzungen erfüllen und der C++-Debugger muss im TwinCAT-Projekt aktiviert sein (TwinCAT 3 C++ Enable C++ debugger). Nachdem der Debugger verbunden (attached) wurde, was vor aber auch noch nach dem Auftreten der Exception erfolgen kann, wird der verursachende Block im Blockdiagramm hervorgehoben, sofern die verursachende Codezeile einem Block zugeordnet werden kann. Der Name des Blockes wird rot dargestellt und der Block selbst wird fett markiert.



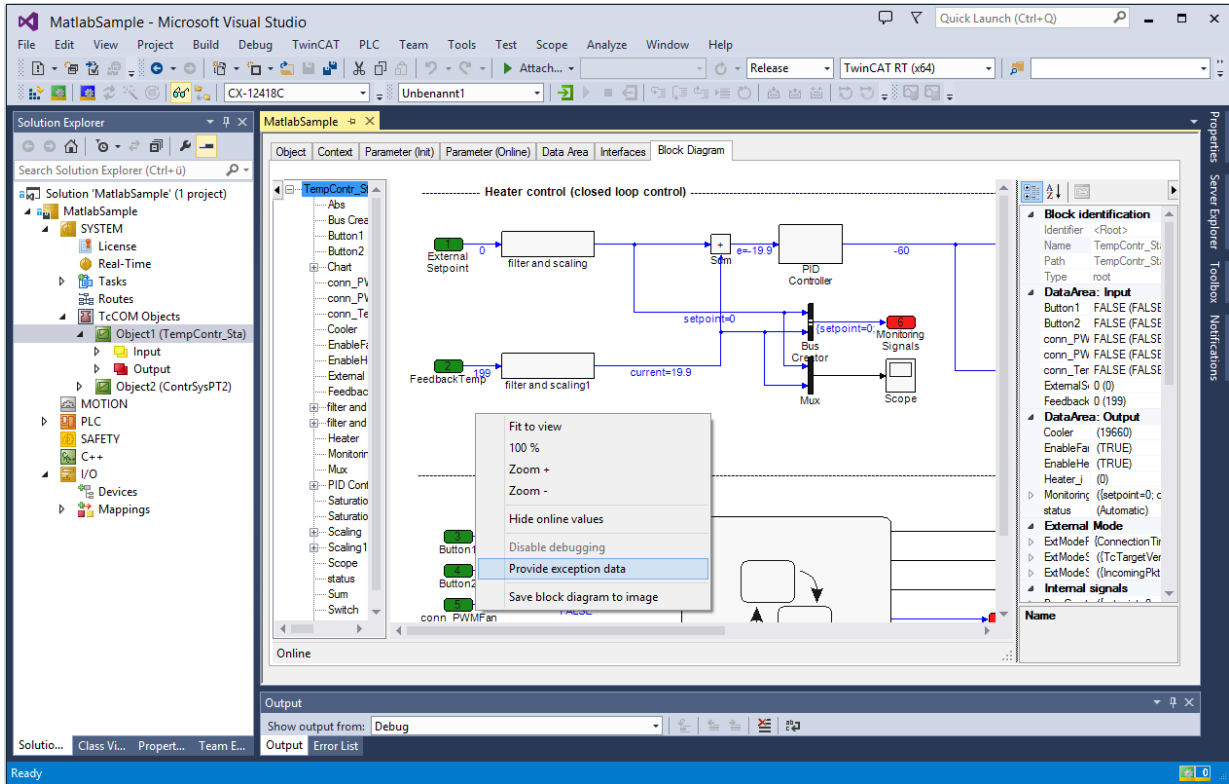
### Manuelle Auswertung von Exceptions ohne Quellcode

Auch wenn auf dem Engineering-System nicht der Source Code des Modules verfügbar ist oder der C++-Debugger nicht aktiviert wurde, kann man nach Auftreten einer Exception die Fehlerstelle im Blockschaltbild hervorheben.

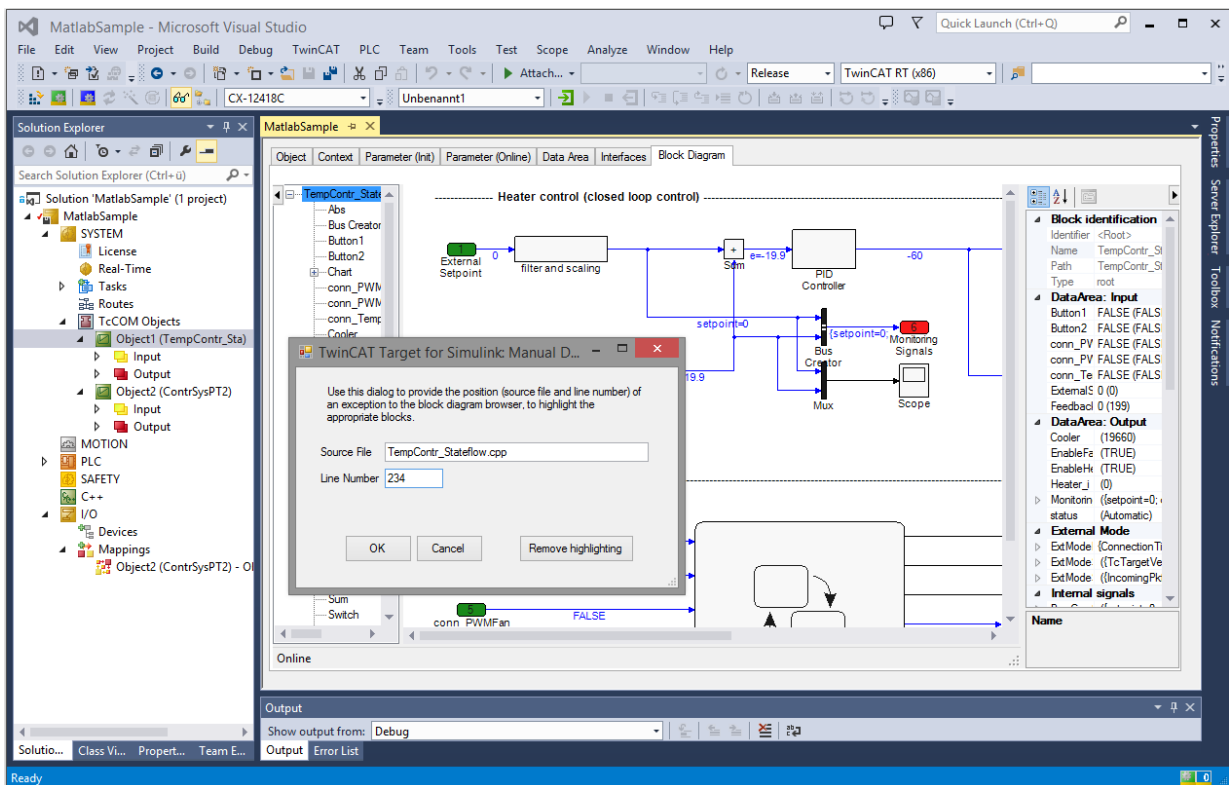
Typischerweise wird beim Auftreten eines Fehlers immer eine Fehlermeldung generiert, in der die Quellcode-Datei sowie die Zeile im Quellcode angegeben ist. Über diese Information lässt sich eine Exception in vielen Fällen einem Block des Blockschaltbildes zuordnen. Dazu kann man wie folgt vorgehen:

- ✓ Voraussetzung für das Hervorheben der Fehlerstelle innerhalb des Blockdiagramms ist, dass die Debuginformationen erzeugt wurden (Option **Export block diagram debug information** in den Coder-Einstellungen unter **Tc Advanced**).

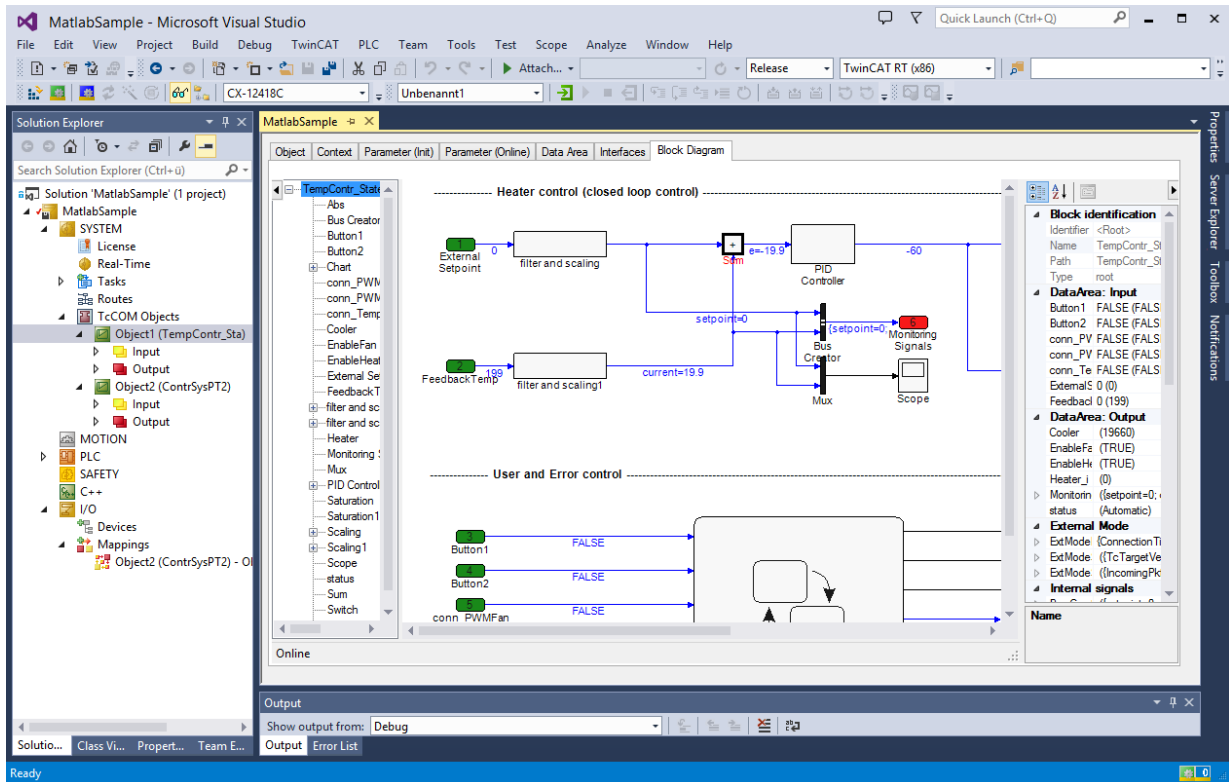
3. Aus dem Kontext-Menü des Blockschaltbildes ist der Eintrag **Provide exception data** zu wählen:



4. In dem sich öffnenden Dialog sind die in der Fehlermeldung bereitgestellte Quellcode-Datei und Zeilennummer einzutragen:



5. Der Name des Blockes, welchem die Zeilennummer zugeordnet ist, wird rot dargestellt und der Block selbst wird fett markiert:



## 3.2 MATLAB®-TcCOM

Wurde mit dem TwinCAT Target for MATLAB® ein TwinCAT-Objekt erzeugt und der MATLAB® Code Export dabei ausgeführt, kann der MATLAB® Code der MATLAB®-Function als Control in der TwinCAT XAE dargestellt werden.

```

TwinCAT Project50
Object Context Parameter (Init) Parameter (Online) Data Area Interfaces Block Diagram
Object1 (BaseStatisticlterative) > BaseStatisticlterative
% iterative implementation of mean and standard deviation
% persistent variable -> FB with internal state in PLC

function [ mean_x [ 0 ], std_x [ 0 ] ] = BaseStatisticlterative( x [ 0 ] )

persistent n [ 2385 ] Mn [ 0 ] Sn

% init of persistent vars
if isempty( n [ 2384 ] )
    n [ 2383 ] = 0;
    Mn [ 0 ] = 0;
    Sn [ 0 ] = 0;
end

% update standard deviation
if n [ 2382 ] > 0
    Sn [ 0 ] = Sn [ 0 ] + ( n [ 2380 ] * x [ 0 ] - Mn [ 0 ] )^2 / ( n [ 2379 ]^2 + n [ 2378 ] );
end

% update mean
Mn = Mn [ 0 ] + x [ 0 ] ;

% update population count
n = n [ 2377 ] + 1;

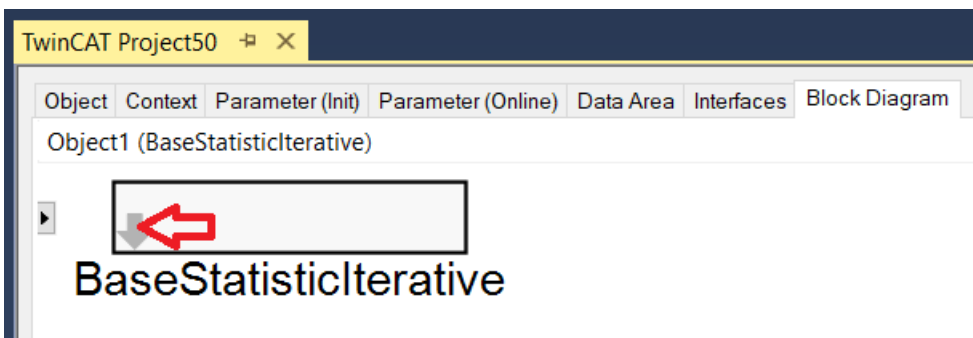
% output scaled values
if n [ 2376 ] > 1
    std_x [ 0 ] = sqrt( Sn [ 0 ] / ( n [ 2375 ] - 1 ) );
else
    std_x [ 0 ] = 0;
end
mean_x = Mn [ 0 ] / n [ 2381 ] ;

Online
    
```

### 3.2.1 Bedienung des Blockdiagramm-Fensters

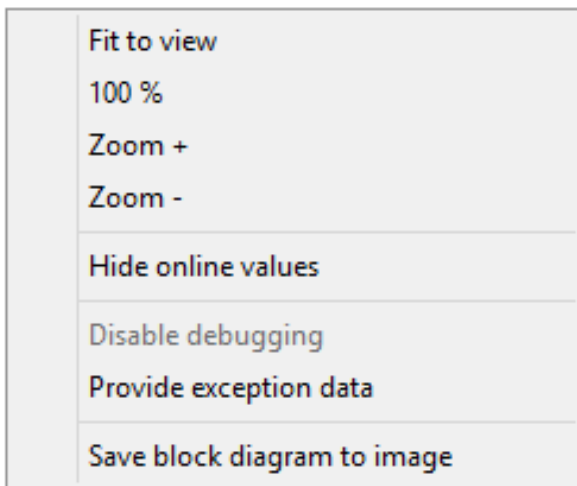
Bei der Generierung eines TcCOM-Moduls aus MATLAB® kann der Export des MATLAB®-Codes konfiguriert werden. Wurde der Export aktiviert, findet man den Code in der TwinCAT-Entwicklungsumgebung unter dem Karteireiter **Block Diagram** der Modul-Instanz.

Auf oberster Ebene finden Sie das erstellte Modul in Block-Darstellung. Wählen Sie den grauen Pfeil im Block an, um den Inhalt dazustellen.



**Shortcut-Funktionen:**

Shortcut	Funktion
Space	Zoom auf die aktuelle Größe des Blockdiagramm-Reiters
Backspace	Wechseln auf die nächst höhere Hierarchiestufe
ESC	Wechseln auf die nächst höhere Hierarchiestufe
STRG + "+"	Herein zoomen
STRG + "-"	Heraus zoomen
F5	Attach Debugger (System- > Real-Time -> C++ Debugger -> Enable C++ Debugger muss aktiviert sein)

**Kontextmenü-Funktionen:****3.2.2 Anzeige von Signalverläufen**

Ausgewählte Variablen können im TwinCAT XAE über ADS abgerufen werden. Somit besteht die Möglichkeit, diese in einem Mini-Scope innerhalb des Block-Diagramms, oder mit dem TwinCAT Scope innerhalb eines Measurement-Projektes, darzustellen.

Variablen, welche im Scope dargestellt werden können, haben einen nachgestellten schwarzen Rahmen in der Code-Darstellung. In diesem Rahmen werden im laufenden Betrieb die Werte in Blau dargestellt.

Durch Drag&Drop einer „blauen Variablen“ auf das Block-Diagramm-Fenster öffnet sich ein Mini.Scope.

```

function [ mean_x [ ... ], std_x [ 1977437994 ] ] = BaseStatisticIterative( x [ 4 ] )

persistent n [ 8765 ] Mn [ 14824 ] Sn

% init of persistent vars
if isempty( n [ 8764 ] )
    n [ 8763 ] = 0;
    Mn [ 14832 ] = 0;
    Sn [ ... ] = 0;
end

% update standard deviation
if n [ 8762 ] > 0
    Sn [ ... ] = Sn [ ... ] + ( n [ ... ] - Mn [ 14828 ] )^2 / ( n [ 8759 ]^2 + n [ 8758 ] );
end

% update mean
Mn = Mn [ 14828 ] + x [ 4 ];

% update population count
n = n [ 8757 ] + 1;

% output scaled values
if n [ 8756 ] > 1
    std_x [ ... ] = sqrt( Sn [ ... ] / ( n [ 8755 ] - 1 ) );
else
    std_x [ ... ] = 0;
end
mean_x = Mn [ 14840 ] / n [ 8761 ];
    
```

Durch Drag&Drop einer „blauen Variablen“ auf die Axis Group eines Charts im TwinCAT Measurement-Projekt, werden die Variablen zum TwinCAT Scope hinzugefügt.

Welche Variablen sind als „blaue Variablen“ im TwinCAT XAE sichtbar?

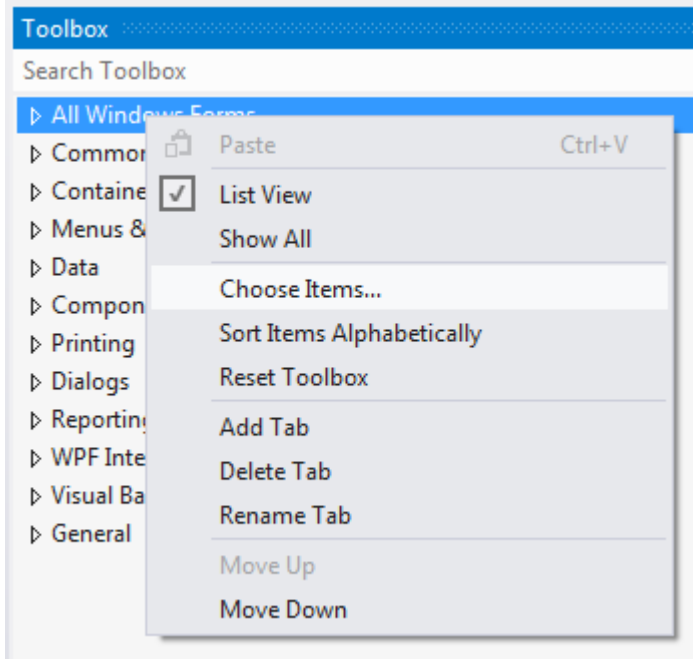
- Die Eingangsgrößen
- Die Ausgangsgrößen
- Persistente Variablen (MATLAB-Definition persistent var1 ... varN)

### 3.3 Einbinden des Blockdiagramm-Controls

Das Control welches das Block-Diagramm in der TwinCAT XAE Umgebung darstellt, kann auch als Control in eigene Visualisierungen eingebunden werden.

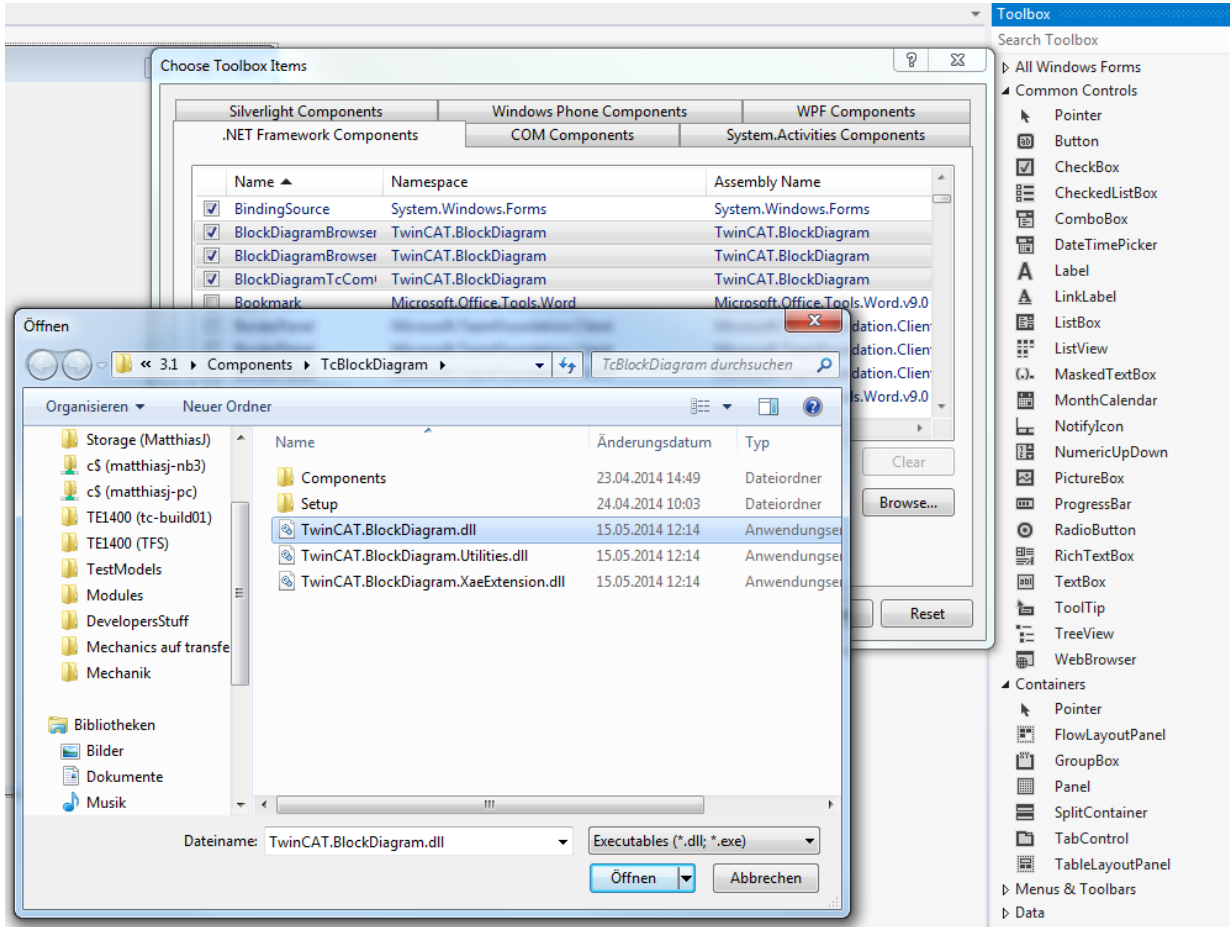
Folgende Schritte sind dafür notwendig:

1. Erstellen Sie eine Windows-Forms-Applikation.
2. Fügen Sie TwinCAT.BlockDiagramm.dll zur Toolbox hinzu:
3. Wählen Sie dafür im Kontextmenü den Eintrag „Choose Items...“ aus.

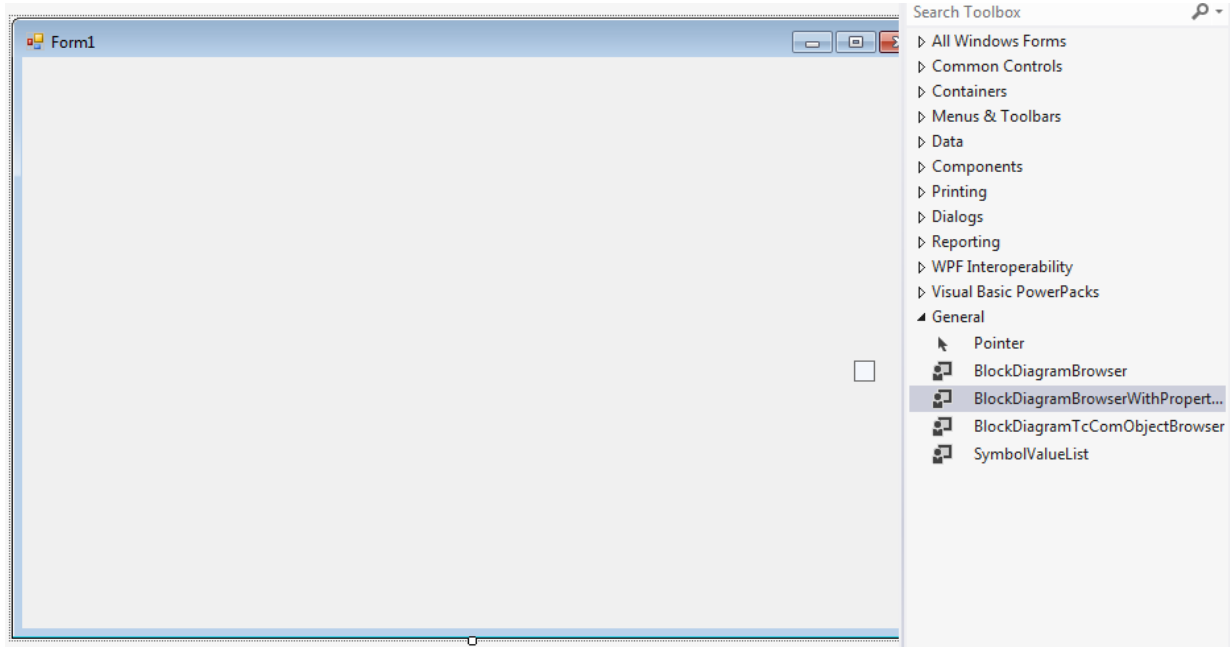




- Browsen Sie zur `TwinCAT.BlockDiagram.dll`, welche sich unter `<TwinCAT-Installationspfad>\3.1\Components\TcBlockDiagram` befindet.



- Fügen Sie eine `TcBlockDiagram-Control-Instanz` zum `Windows-Forms-Object` per `Drag&Drop` hinzu.



## 4 TwinCAT Automation Interface: Verwendung in MATLAB®

### Kurzbeschreibung des Automation Interface

Mit dem TwinCAT Automation Interface können TwinCAT XAE-Konfigurationen per Programmier-/ Skriptcodes automatisch erzeugt und bearbeitet werden. Die Automatisierung einer TwinCAT-Konfiguration steht dank sogenannter Automation Interfaces zur Verfügung, auf die über alle COM-fähigen Programmiersprachen (z. B. C++ oder .NET) und auch über dynamische Scriptsprachen, wie Windows PowerShell, IronPython, oder sogar das (veraltete) Vbscript, zugegriffen werden kann. Ebenfalls ist die Verwendung aus der MATLAB®-Umgebung möglich.

Eine ausführliche Dokumentation der Produkts finden Sie hier: [TwinCAT Automation Interface](#)

### Verwendung in MATLAB®

In MATLAB® können Sie das Automation Interface durch das Kommando `NET.addAssembly` sichtbar machen. Damit sind Sie in der Lage, die in der Produktdokumentation beschriebenen Interfaces ([Automation Interface API](#)) zu nutzen. Ebenfalls finden Sie in der Produktdokumentation viele Programmierbeispiele für die Nutzung aus C# und PowerShell ([Automation Interface Configuration](#)).

Um Ihnen den Einstieg aus MATLAB® zu vereinfachen, finden Sie im Folgenden eine Beispielimplementierung für MATLAB® auf Basis einer MATLAB®-Klasse, welche Sie nutzen, verändern und erweitern können.

## 4.1 Beispiel: Tc3AutomationInterface

### Übersicht

Der Beispiel-Code besteht aus zwei m-Files:

- *Tc3AutomationInterface.m*: MATLAB®-Klasse, welche einige häufig genutzte Methoden implementiert.
- *Tc3AutomationInterfaceGuide.mlx*: MATLAB® Live-Script, welches die MATLAB®-Klasse beispielhaft aufruft.

### Beispiel mit MATLAB® aufrufen

Das TwinCAT Tool for MATLAB® and Simulink® Setup installiert das Beispiel auf Ihrem System. Rufen Sie das Beispiel mit dem MATLAB® Command Window auf:

```
TwinCAT.ModuleGenerator.Samples.Start('AutomationInterface').
```

### Das MATLAB®-Script

Das MATLAB®-Script liefert ein Beispiel, wie Sie eine TwinCAT-Solution erzeugen, den EtherCAT-Master nach I/O scannen, zwei TcCOM-Module instanzieren, verlinken und das Projekt auf einem Target aktivieren können.

Um das Script ausführen zu können, müssen die beiden verwendeten TcCOM in Ihrem *publish directory* `%TwinCATDir%\CustomConfig\Modules\` vorhanden sein. Laden Sie dazu das Beispiel [Temperature Controller](#) aus der TE1400 | Target for MATLAB®/Simulink® herunter. Die Dateidreier aus dem Verzeichnis `.\TE1400Sample_TemperatureController\PrecompiledTcComModules\Actual TwinCAT versions\` kopieren Sie dann in das *publish directory*.

Führen Sie das m-File *Tc3AutomationInterface\_Testbench.m* aus. Es wird im Hintergrund die aktuellste auf Ihrem System verfügbare Visual Studio-Instanz geöffnet und die TwinCAT Solution konfiguriert, speichert und aktiviert.

### Die MATLAB®-Klasse

### Die Properties

In den Properties der Klasse *Tc3AutomationInterface* werden alle zur Instanz der Klasse gehörigen Variablen und Interfaces gehalten. So können mehrere TwinCAT Solutions in einem MATLAB®-Script aufgebaut werden, indem für jede Solution eine Instanz der Klasse erzeugt wird. Damit ergeben sich dann keine Überschneidungen.

### Der Konstruktor

```
function this = Tc3AutomationInterface
```

Der Konstruktor lädt alle notwendigen Assemblies und setzt bei Erfolg das Property *AssembliesLoaded* auf TRUE. Die geladenen Assemblies sind:

- EnvDTE und EnvDTE80: Bibliotheken für das Visual Studio Core Automation. Notwendig zur Konfiguration von Visual Studio.
- Tc3SysManagerLib: TwinCAT Automation Interface Bibliothek zur Konfiguration einer TwinCAT Solution in Visual Studio.
- TwinCAT.Ads: ADS Bibliothek, z. B. zum Lesen und Verändern des XAR state.
- System.Xml: Bibliothek zum Parsen von XML Dateien.

### Ausgewählte Methoden der Klasse

```
function TcComObject = CreateTcCOM(this, Modelname)
```

Nutzen Sie die Hilfsfunktion von MATLAB®, um die Funktion und die Parameter der Methode einzusehen.

```
>> help Tc3_AI.CreateTcCOM  
--- help for Tc3AutomationInterface/CreateTcCOM ---
```

```
CreateTcCOM creates a new instance of a TcCOM
```

```
TcComObject = CreateTcCOM(Modelname)  
Instanciates the TcCOM with the specified name (Modelname).  
Also a task with a matching cycle time is created and linked to  
the TcCOM-Object.
```

```
set properties: TcCOM
```

```
see also:
```

```
Beckhoff Infosys
```

Ebenfalls wird bei einigen Methoden ein Link ins das Beckhoff Infosys angeboten. Diese verweisen auf Dokumentationsbeispiele aus der TwinCAT Automation Interface Dokumentation, sodass Sie direkt einen Vergleich zur Implementierung in MATLAB®, C# und PowerShell einsehen können. Ebenfalls finden Sie in einigen Sektionen im Kommentar einen Link zum Beckhoff Infosys, sodass Sie die Quelle der Information einsehen können.

Die Methode *CreateTcCOM* beginnt zunächst mit dem Parsen der *<modelname>.tmc* Datei. Daraus wird mit der *System.Xml* die ClassID, die Task Zykluszeit sowie Task Priorität extrahiert. Dann wird mit dem Automation Interface ein entsprechendes TcCOM instanziiert sowie eine (oder mehrere) zugehörige Task(s) erzeugt. Abschließend wird (werden) die Task(s) dem TcCOM zugeordnet.

```
function ActivateOnDevice(this, AmsNetId)
```

Um den aktuellen Status einer TwinCAT Runtime, z. B. *config* oder *run*, zu erfragen bzw. zu verändern, wird TwinCAT ADS genutzt. In der Methode *ActivateOnDevice* wird zunächst die XAR mit der spezifizierten *AmsNetId* in den Config-Modus geschaltet und dann die aktuelle TwinCAT-Konfiguration aktiviert sowie das System gestartet. Zwischen den Einzelschritten sind Pausen eingetragen, da dieser Vorgang ggf. etwas Zeit benötigt.


### **Statische Methoden**

Statische Methoden sind auch ohne Instanz der Klasse verfügbar.

```
function vsVersions = GetInstalledVisualStudios
```

Vorbereitet ist hier eine Funktion, welche über Registry Key Einträge die auf dem System verfügbaren Visual Studio Installationen detektiert und auflistet. Die Implementierung ist auf VS 2010 bis VS 2017 limitiert.

### **Dokumente hierzu**

 [https://infosys.beckhoff.com/content/1031/tc3\\_matlab\\_overview/Resources/5776206091.zip](https://infosys.beckhoff.com/content/1031/tc3_matlab_overview/Resources/5776206091.zip)



Mehr Informationen:  
**[www.beckhoff.com/te1000](http://www.beckhoff.com/te1000)**

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Deutschland  
Telefon: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

