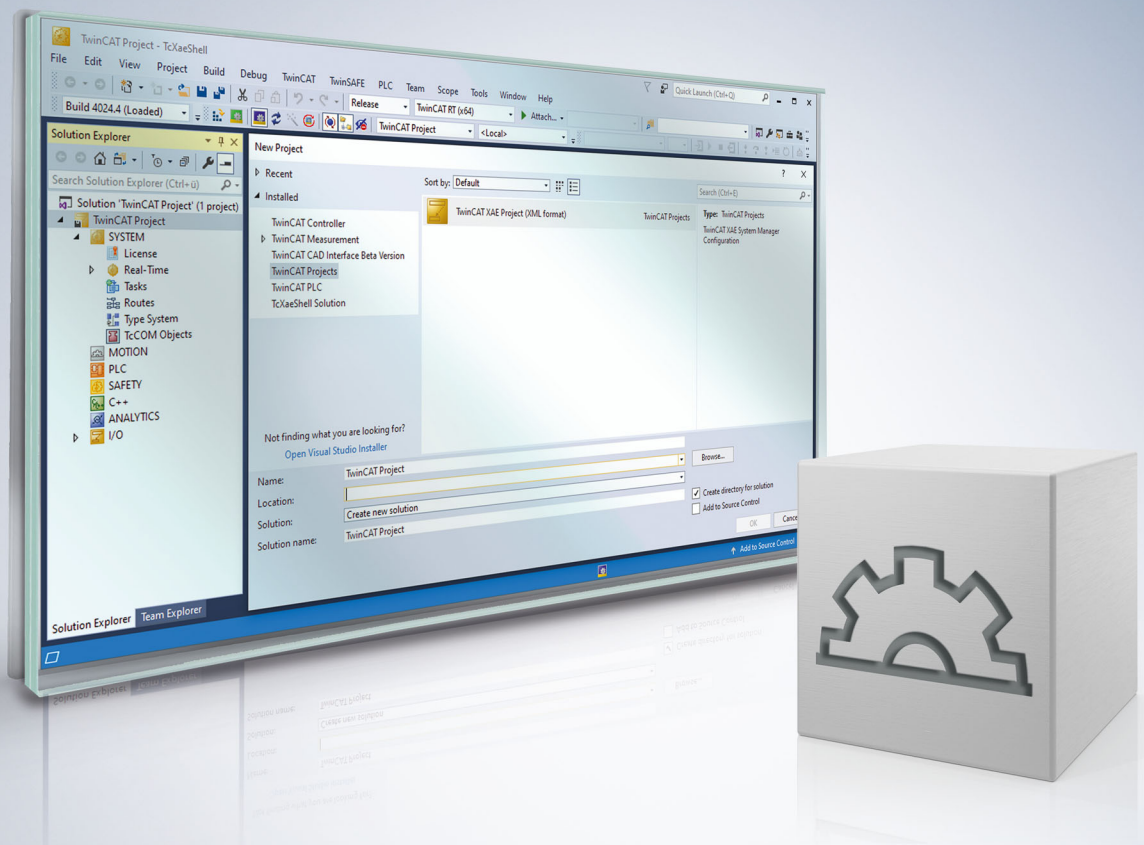
BECKHOFF New Automation Technology

Handbuch | DE

TF5110 - TF5113

TwinCAT 3 | Kinematic Transformation



24.08.2022 | Version: 2.2

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

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

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

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

EtherCAT®

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

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

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

Haftungsausschluss

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

Qualifikation des Personals

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

Erklärung der Symbole

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

⚠ GEFAHR

Akute Verletzungsgefahr!

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

⚠ WARNUNG

Verletzungsgefahr!

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

⚠ VORSICHT

Schädigung von Personen!

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

HINWEIS

Schädigung von Umwelt oder Geräten

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

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Hinweise zur Informationssicherheit

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

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

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

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

2 Einführung

Das TF5110 - TF5113 TwinCAT Kinematic Transformation Software Paket wird zusammen mit dem TF5400 Software Paket installiert.

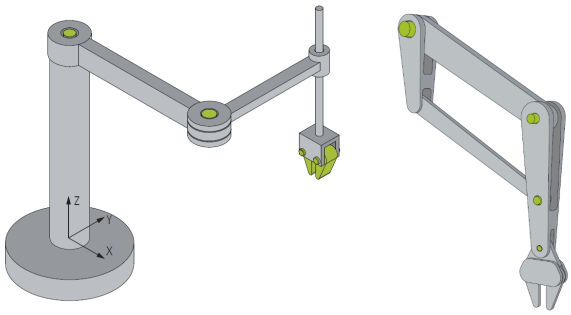
TwinCAT Kinematic Transformation

Die TF5110 - TF5113 TwinCAT Kinematic Transformation ist eine Software Lösung, die Robotersteuerung und konventionelle SPS in einem System vereint (siehe https://www.beckhoff.de/tf5113/). Durch die Implementierung der gesamten Steuerung in einem System entfallen Schnittstellenverluste zwischen verschiedenen CPUs für SPS, Motion Control und Robotik. In der Praxis führt diese Implementierung zu einer Reduktion von Engineeringkosten und zu einer Verkürzung von Takzzeiten im Fertigungsprozess. Zusätzlich zum Wegfall von Schnittstellen und Komponenten führt das Verschmelzen von SPS, Robotik und Motion Control zu einer Applikation zu einem homogenen System. Deshalb gibt es für den Anwender keinen erkennbaren Unterschied in der Behandlung der einzelnen Funktionalitäten. Auf diese Weise ist es problemlos möglich, ein Teil auf einem Förderband, das mit Standard-Motion-Control betrieben wird, mit dem Roboter zügig und geschickt zu greifen und wegzulegen.

Da der Aufbau und die Anzahl der Achsen den Arbeitsraum des Roboters bestimmen, ist dieser von verschiedenen Parametern abhängig: Armlängen, Reichweitenwinkel, Schwerpunkt, max. Last etc. Die Anordnung der Arme und Gelenke bestimmt die kinematische Struktur, die in zwei Hauptklassen geteilt wird – die serielle Kinematik und die parallele Kinematik.

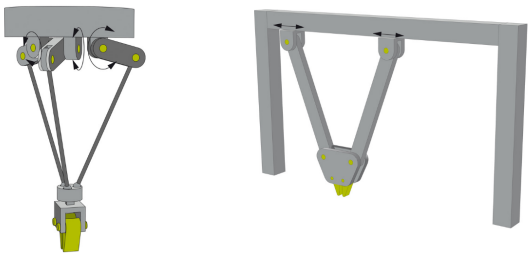
Serielle Kinematik

Die aktuelle Position einer beliebigen Achse ist immer von der Position der vorhergehenden Achse abhängig, das heißt alle Achsen sind nacheinander angeordnet.
Beispiele: SCARA und Kran-Kinematik



Parallele Kinematik

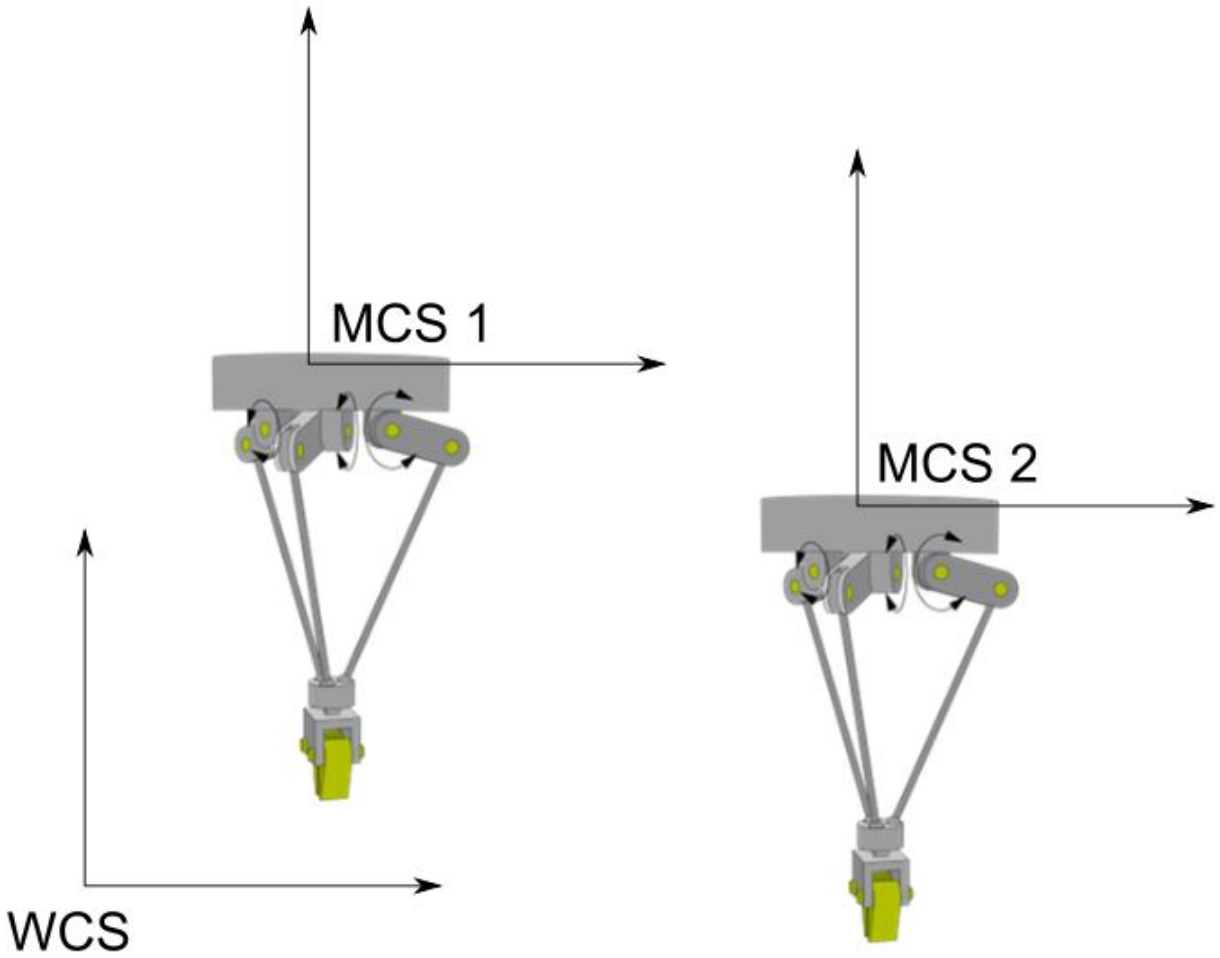
Alle Achsen greifen über die Kinematik direkt an der Arbeitsplattform an.
Beispiele: Delta-Kinematik, Scheren-Kinematik



Koordinatensysteme

Zur Beschreibung des Positionierverhaltens eines Systems benötigt man Koordinatensysteme. Zur Programmierung können unterschiedliche Koordinatensysteme als Grundlage genutzt werden:

- Das Maschinenkoordinatensystem (MCS) ist ein robotergebundenes kartesisches Koordinatensystem, welches seinen Ursprung üblicherweise im Sockel des Roboters hat.
- Das Weltkoordinatensystem (WCS) ist ein kartesisches Koordinatensystem, welches die gesamte modellierte Welt beschreibt. Es bezieht sich also nicht auf einen speziellen Roboter, sondern auf die gesamte Anlage. Der Ursprung eines roboterbezogenen Maschinenkoordinatensystems (MCS) liegt in einem Punkt des WCS. Der Anwender kann also festlegen, an welchem Ort seiner „Welt“ sich ein Industrieroboter befindet und wie dieser ausgerichtet ist. Somit ist es auch möglich, dass sich mehrere Roboter in einem WCS befinden. Bei der Verwendung eines Roboters können Weltkoordinaten für eine bessere Übersichtlichkeit aber auch mit den Maschinenkoordinaten zusammenfallen.
- Das Anwenderkoordinatensystem (UCS) kann vom Anwender frei an einer beliebigen Position und mit beliebiger Ausrichtung im Weltkoordinatensystem platziert werden.
- Das Achskoordinatensystem (ACS) beschreibt die Stellung der physikalischen Achsen und ist im Allgemeinen kein kartesisches Koordinatensystem. Viele Robotergelenkachsen vollführen rotatorische Bewegungen. Die Verwendung des ACS erleichtert die Berücksichtigung der Grenzwerte für Winkel, Geschwindigkeit und Beschleunigung. Vollführen Roboterachsen jedoch rotatorische Bewegungen, so ist der Bahnverlauf für den Anwender häufig schwer abschätzbar und kontrollierbar. In der Regel wird das Achskoordinatensystem für Referenzierung bzw. Homing eingesetzt.



Kinematische Transformation

Die Programmierung der Roboter erfolgt häufig im MCS. Da Bewegungen aufgrund des menschlichen Vorstellungsvermögens meist in kartesischen Koordinatensystemen programmiert werden, ist zur Ausführung einer solchen Bewegung eine Umrechnung zwischen Achskoordinatensystem und kartesischem Raum notwendig.

Die Transformation beschreibt, im Zusammenhang mit der Kinematik, die notwendige Berechnung, um von einem Koordinatensystem in ein anderes zu wechseln. Bei der Betrachtung der Kinematiken von Robotern stellen sich grundsätzlich zwei Probleme:

- Die Umrechnung vom Achskoordinatensystem (ACS) in ein kartesisches Koordinatensystem wird als Vorwärtstransformation bezeichnet. Dabei wird die kartesische Position des Tool-Center-Points (TCP) aus den achsspezifischen Gelenkkoordinaten des Roboters berechnet.
- Die Umrechnung von kartesischen Koordinaten des TCP in Achskoordinaten, die notwendig ist, um die realen Roboterachsen zu bewegen, wird als Rückwärtstransformation bezeichnet.

Realisierung in TwinCAT

Mit TwinCAT Kinematic Transformation lassen sich Robotikapplikationen realisieren. Alle SPS- und NC-Eigenschaften können auf einer gemeinsamen Hard- und Softwareplattform kombiniert werden. TwinCAT Kinematische Transformation realisiert diverse Roboterkinematiken (z. B. H-Bot, Delta-Roboter, 6-Achs-Roboter) auf dem PC, die Ansteuerung der Achsen erfolgt direkt aus dem TwinCAT-Motion-Control-System heraus.

Roboterbewegungen werden dabei vom Anwender direkt im kartesischen Koordinatensystem, die Software rechnet in jedem Zyklus die Transformation in das Achskoordinatensystem des Roboters. Zur Minimierung der Schwingungsneigung und zur Erhöhung der Positioniergenauigkeit ist für viele Kinematiken noch eine Stromvorsteuerung aktivierbar, wenn die Antriebsverstärker und der Feldbus schnell genug sind und Schnittstellen für eine zusätzliche Stromvorsteuerung zur Verfügung stehen. EtherCAT und die Beckhoff Servoverstärker vom Typ AX5000 erfüllen diese Anforderungen.

Die TwinCAT Function fügt sich dabei nahtlos in die Motion-Control-Welt von Beckhoff ein. Durch TwinCAT NC I ist die Programmierung sowohl über G-Code (DIN 66025) als auch direkt aus der SPS (PlcInterpolation-Bibliothek) möglich. Die Functions TF5055 TC3 NC Flying Saw und TF5050 TC3 NC Camming ermöglichen z. B. die Synchronisation mit Förderbändern für das Greifen und Absetzen von Werkstücken. Darüber hinaus können standardmäßige PTP-Funktionen aus den bekannten Beckhoff PTP-Motion-Bibliotheken verwendet werden.

Die Konfiguration des Roboters erfolgt komplett in der TwinCAT 3 Engineering Umgebung (XAE).

3 Übersicht der neuen Funktionen

Ab V3.1.10.66:

- Neue Kinematiken:
  - 3D-Tripod Type 1 (P_3Z)
  - 3D-Tripod Type 2 (P_3L)

Ab V3.1.10.63:

- Benötigt TwinCAT V3.1.4024.24 oder höher

Ab V3.1.10.30:

- Neue Kinematiken:
  - 3D-Kinematics Type 7 (PXX_SZ)
  - 3D-Delta T-Type 3 (P_3C3)
  - 3D-Cable Kinematic Type 2 (P_3L)
  - 4D-Kinematics Type 6 (S_XCZC)
  - 4D-Cable Kinematics (P_4L)

Ab V3.1.10.1:

- Neue Funktionsblöcke für Extended Rotation Range sind implementiert:
  - FB_KinPresetRotation
  - FB_KinExtendedRotationRange
- Neue Funktion F_KinAxesInTolerance
- Benötigt TwinCAT V3.1.4024.7 oder höher

Ab V3.1.6.3:

- Die TF511x TC3 Kinematic Transformation werden ein Teil des TF5400 Installationspakets.

4 Installation

Das Installationspaket TF5400 TC3 Advanced Motion Pack enthält die notwendigen Komponenten für die TwinCAT Funktionen TF5110 - TF5113 TC3 Kinematic Transformation, TF5410 TC3 Motion Collision Avoidance, TF5420 TC3 Motion Pick-and-Place und TF5430 TC3 Planar Motion.

Achten Sie darauf, das Advanced Motion Pack sowohl auf dem Entwicklungsrechner als auch auf dem Target-System zu installieren.

Installationsanforderungen

Die Installation TF5400 TC3 Advanced Motion Pack erfordert TwinCAT 3.

Zielsystem

Windows 7, Windows 10

Funktions-Level für TF5110-TF5113

Die Funktion TF5110 - TF5113 TC3 Kinematic Transformation ist in vier verschiedene Level unterteilt, in Abhängigkeit der Anzahl der Transformationsachsen. Ein höherer Level beinhaltet alle Sub-Level.

Für das Level 4 ist eine separate Installationsdatei erforderlich, die Sie auf Anfrage vom Support erhalten.

Level 1: Unterstützt die statische Transformation. Diese beinhaltet eine Translation und Rotation des Koordinatensystems.

Level 2: Unterstützt Level 1 und einfache (hauptsächlich 2D) kinematische Transformationen, wie H-Bot und 2D-Parallelkinematik.

Level 3: Unterstützt Level 2 und komplexere (3D, 4D) kinematische Transformationen, wie Delta-Roboter.

Level 4: Unterstützt Level 3 und komplexe kinematische Transformationen (bis zu 6D).

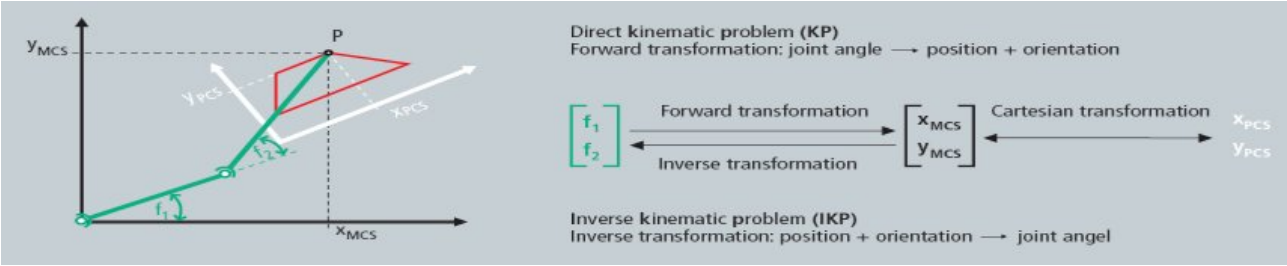
Zusätzliche Lizenzanforderungen

TF5110 - TF5113 TC3 Kinematic Transformation benötigt die Lizenz TC1260.

5 Konfiguration

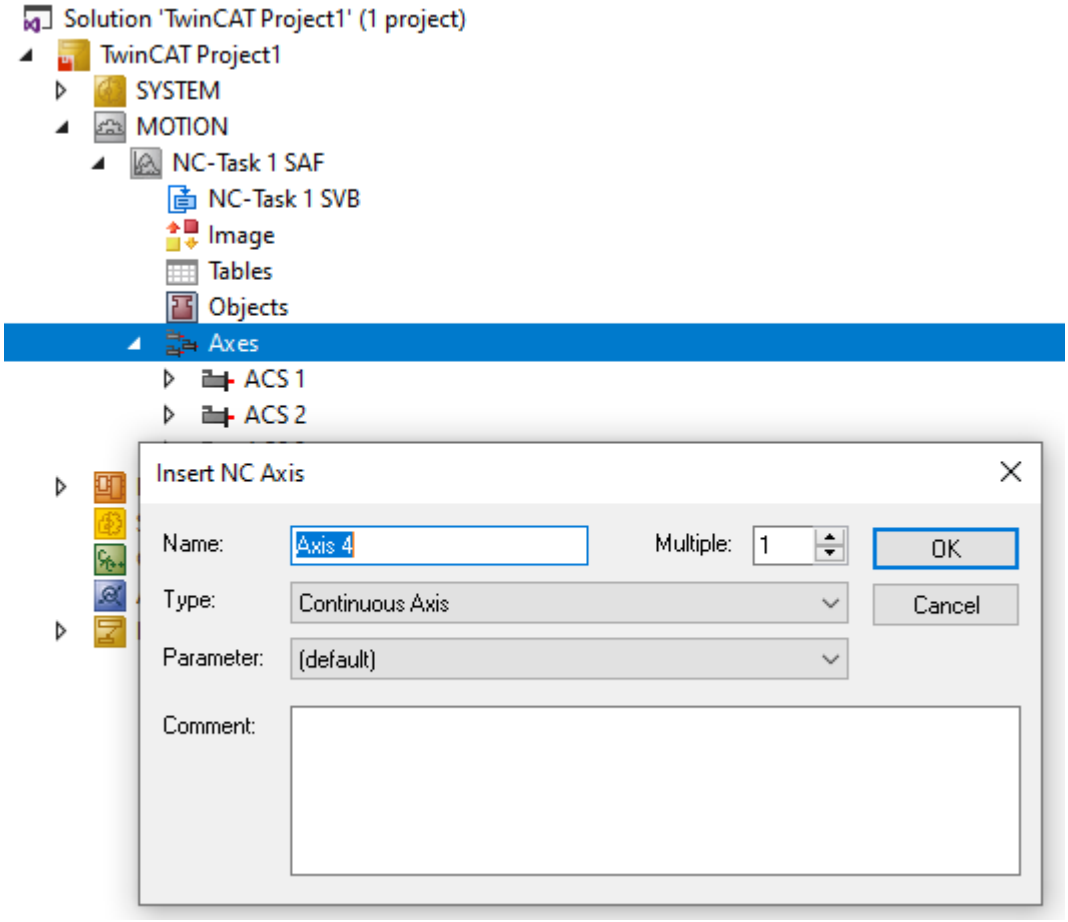
Basierend auf PLCopen unterscheiden wir zwischen zwei Hauptkoordinatensystemen (Einführung [▶ 8]):

- Achskoordinatensystem (ACS „Axis Coordinate System“)
- Maschinenkoordinatensystem (MCS „Machine Coordinate System“)

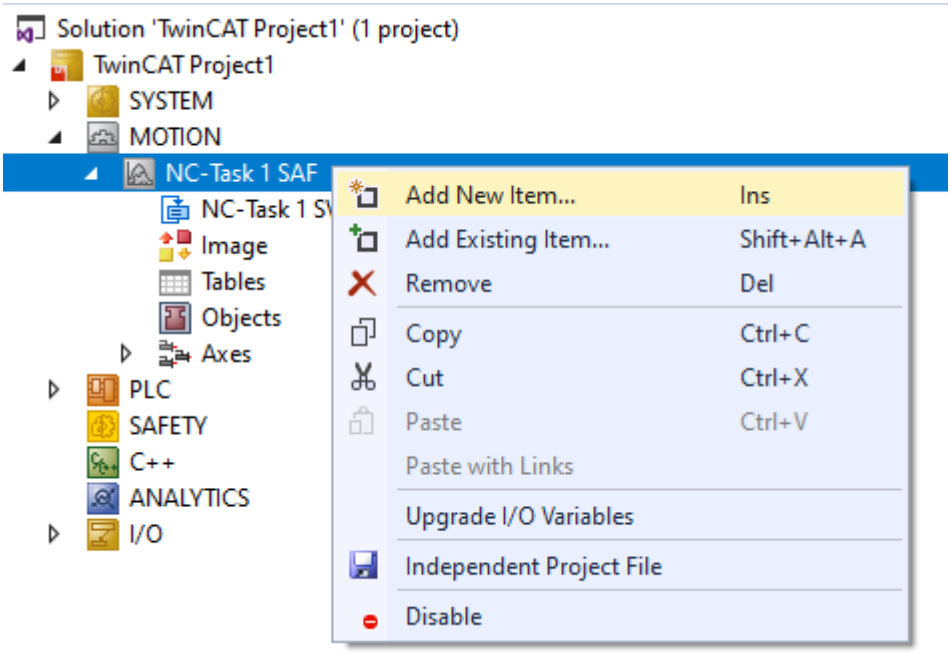


Kinematischen Transformationskanal konfigurieren

1. Alle Achsen (ACS und MCS) zur NC-Konfiguration in der XAE hinzufügen, genau wie PTP-Achsen. Die Achsen des ACS sind Hardwareachsen und werden mit Antrieben verknüpft, die Achsen des Maschinenkoordinatensystem (MCS) sind reine Softwareachsen des Typs Simulationsencoder. Alle ACS - und MCS -Achsen, die in einem kinematischen Transformationskanal verwendet werden, müssen in der XAE erzeugt werden. So hat z.B. ein Delta-Roboter [▶ 31] 3 ACS-Achsen (M1...M3) und 3 MCS-Achsen (X, Y, Z).
2. Dazu auf „Axes“ einen Rechtsklick und „Add new item“ auswählen.
3. Dann im Fenster „Insert NC Axis“ die Achsen entsprechend der Kinematik anlegen.

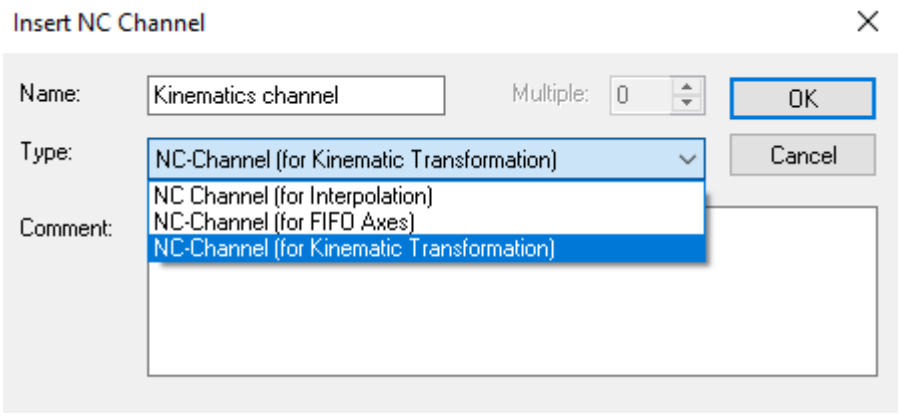


4. Einen Kinematikkanal zur XAE Konfiguration hinzufügen.

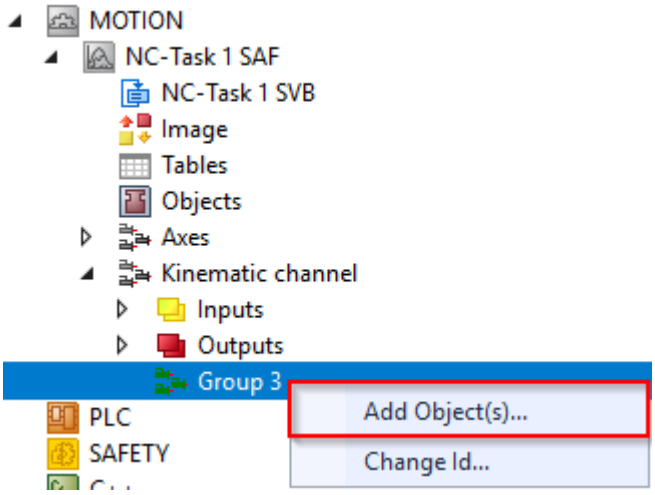


⇨ Durch das Hinzufügen eines Kanals wird eine Instanz einer Kinematikgruppe erzeugt.

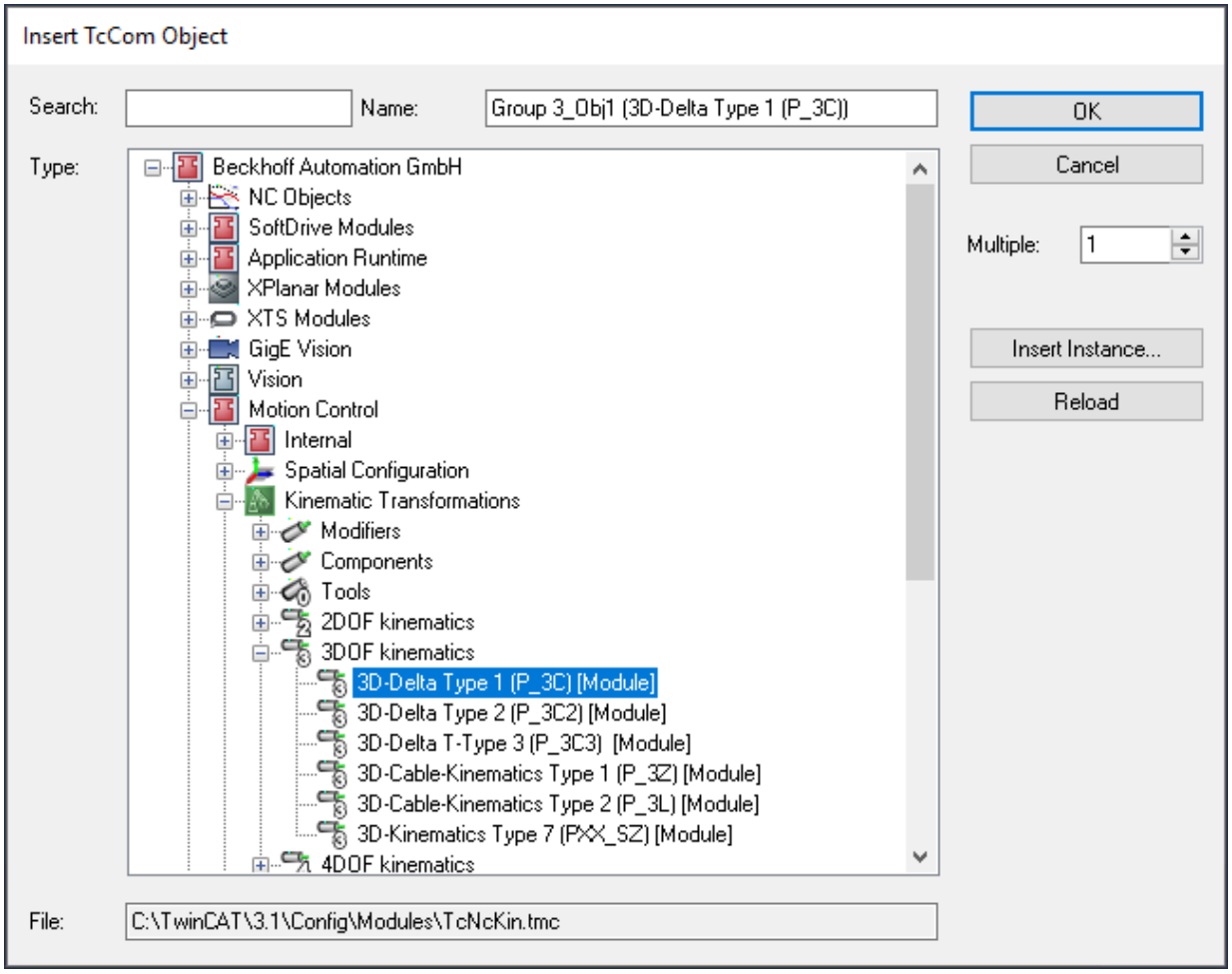
5. Den Kanaltyp auswählen: **NC-Kanal (für Kinematic Transformation)** um eine kinematische Transformation auszuführen.



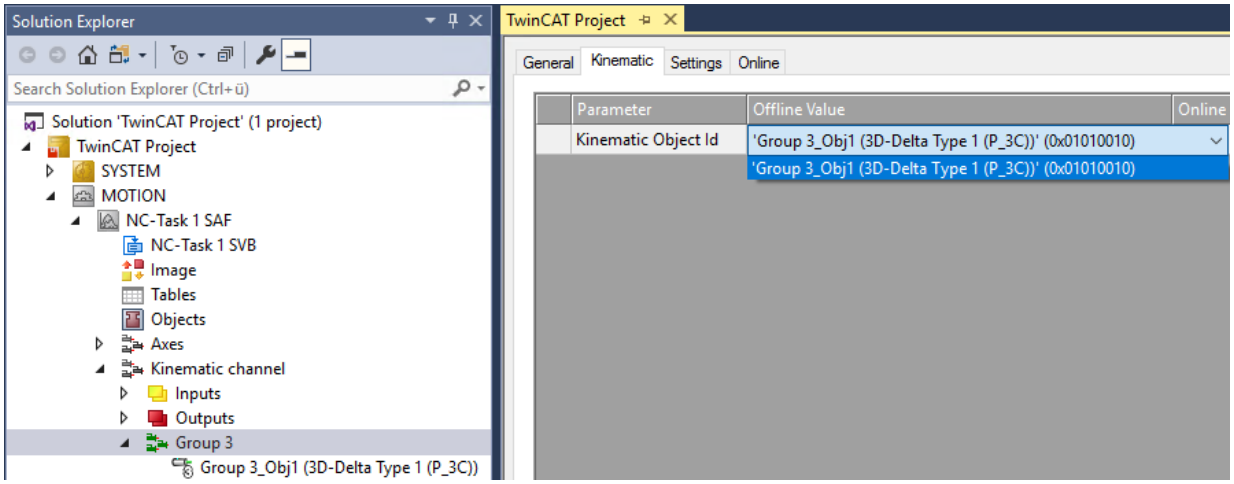
6. Die Objekte unter der Gruppe hinzufügen, die die kinematische Konfiguration des Benutzers darstellen.



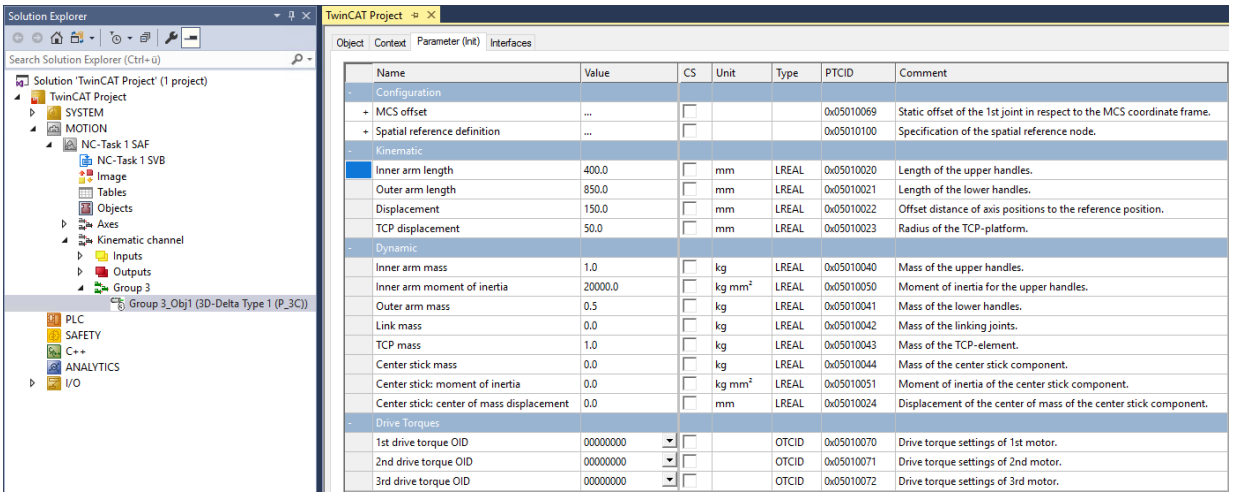
7. Zum Starten der Transformation für einen Delta-Roboter, wählen Sie z.B.
   - Delta Type 1
   Zusätzlich können optional Werkzeuge [▶ 51] und Koordinatensysteme [▶ 53] (UCS) angelegt werden.



8. Die Transformationsgruppe muss wissen, welches Root-Modul aufzurufen ist. Deshalb muss die Objekt-ID der Kinematik (in diesem Fall Delta Type1) ausgewählt werden. Das Kinematik-Objekt definiert die Anzahl der in der SPS zu verwendenden ACS - und MCS -Achsen (siehe ST_KinAxes [▶ 81]).



9. Die Objektparameter entsprechend der verwendeten Kinematik parametrieren.

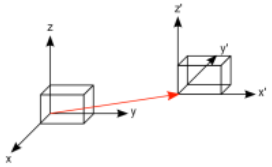
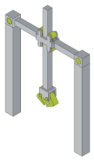
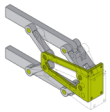
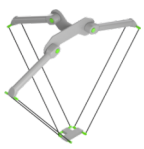


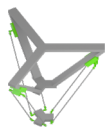
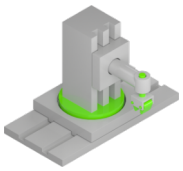
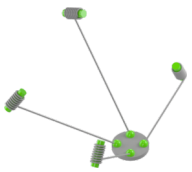
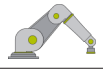
10. Die Transformation kann jetzt von der SPS aus aktiviert werden (siehe SPS-Bibliothek [▶ 65]). Zum Ansprechen der Transformation ein zyklisches Kanalinterface in der SPS definieren und dieses mit den I/O des Kinematikkanals verknüpfen.

```
in_stKinToPlc       AT %I*      : NCTOPLC_NCICHANNEL_REF;
out_stPlcToKin      AT %Q*      : PLCTONC_NCICHANNEL_REF;
```

6 Unterstützte Transformationen

Übersicht

Transformationstyp	Schema	Erforderliche TwinCAT Funktion (Level)
Static Transformation [▶ 21]		TF5110 TC3 Kinematic Transformation (Level 1)
2D-Kinematics Type 1 (P_2C) [▶ 23]		TF5111 TC3 Kinematic Transformation (Level 2)
2D-Kinematics Type 2 (P_2C2) [▶ 25]		TF5111 TC3 Kinematic Transformation (Level 2)
2D-Kinematics Type 3 (S_CC) [▶ 26]		TF5111 TC3 Kinematic Transformation (Level 2)
2D-Kinematics H-Bot (P_2Y) [▶ 27]		TF5111 TC3 Kinematic Transformation (Level 2)
2D-Kinematics Type 5 (S_CC) [▶ 28]		TF5111 TC3 Kinematic Transformation (Level 2)
2D-Kinematics Type 6 (P_2X) [▶ 30]		TF5111 TC3 Kinematic Transformation (Level 2)
3D-Delta Type 1 (P_3C) [▶ 31]		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Delta Type 2 (P_3C2)		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Delta T Type 3 (P_3C3) [▶ 33]		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Tripod Type 1 (P_3Z) [▶ 35]		TF5112 TC3 Kinematic Transformation (Level 3)

Transformationstyp	Schema	Erforderliche TwinCAT Funktion (Level)
3D-Tripod Type 2 (P_3L) [▶ 36]		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Cable Type 1 (P_3Z)		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Cable Type 2 (P_3L) [▶ 37]		TF5112 TC3 Kinematic Transformation (Level 3)
3D-Kinematics Type 7 (PXX_SZ) [▶ 39]		TF5112 TC3 Kinematic Transformation (Level 3)
4D-SCARA (S_CCZC) [▶ 41]		TF5112 TC3 Kinematic Transformation (Level 3)
4D-Kinematics Type 6 (S_XCZC) [▶ 42]		TF5112 TC3 Kinematic Transformation (Level 3)
4D-Cable (4D_P_4L) [▶ 43]		TF5112 TC3 Kinematic Transformation (Level 3)
5D-Kinematics Type 2 (XYZab) [▶ 44]		TF5113 TC3 Kinematic Transformation (Level 4)
5D-Kinematics Type 3 (XYZAB) [▶ 46]		TF5113 TC3 Kinematic Transformation (Level 4)
Stewart Platform (P_6L) [▶ 48]		TF5113 TC3 Kinematic Transformation (Level 4)
Six Axis Articulated (S_CBBCBC) [▶ 50]		TF5113 TC3 Kinematic Transformation (Level 4)

Zusätzliche Objekte

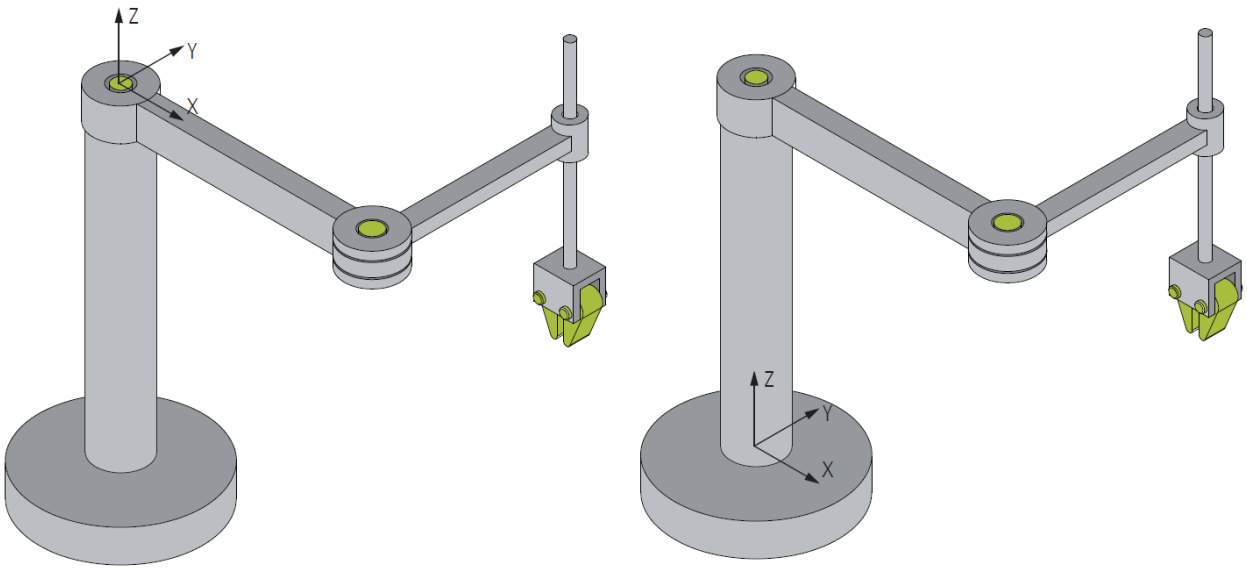
Die folgenden Objekte können angelegt und in der zugehörigen Kinematik ausgewählt werden. Die Auswahl geschieht über eine Dropdown-Parameter-Liste in der Kinematik. Dort muss die entsprechende Objekt-ID (OTCID) ausgewählt werden.

Objekttyp	Beschreibung	Erforderliches Level und Version
Tool Offset [▶ 51]	Werkzeugversatz - beschreibt ein Werkzeug auf Höhe des Kinematikflansches.	TF5110 TC3 Kinematic Transformation (Level 1)
Tool Linear [▶ 53]	Werkzeug Linear - beschreibt ein am Kinematikflansch befestigtes 1D-Werkzeug, das die Möglichkeit bietet, den TCP in Richtung des Werkzeugs zu bewegen.	TF5110 TC3 Kinematic Transformation (Level 1)
Drive Torque [▶ 51]	Antriebsdrehmoment - stellt die Trägheit und die Effizienz des Motors und des Getriebes dar, um das dynamische Modell genauer berechnen zu können.	TF5110 TC3 Kinematic Transformation (Level 1)
Coordinate Frame [▶ 53]	Koordinatensystem - beschreibt ein benutzerdefiniertes Koordinatensystem.	TF5110 TC3 Kinematic Transformation (Level 1)

6.1 Allgemeine Parameter für die Kinematik

MCS Offset

Mit dem MCS offset können zusätzliche Offset-Parameter vor der ersten Achse (bzw. vor der Basis) der Kinematik parametriert werden. Beispielsweise liegt bei der SCARA-Kinematik der Ursprung des MCS im ersten Gelenk (M1). Der Parameter Z-shift von dem MCS offset kann dazu genutzt werden, die zusätzliche Stablänge zu parametrieren, so dass der Ursprung des MCS im Fuß des Roboters liegt.



Parameter	Beschreibung	Typ	Einheit
X-shift	Statischer X-offset im MCS.	LREAL	mm
Y-shift	Statischer Y-offset im MCS.	LREAL	mm
Z-shift	Statischer Z-offset im MCS.	LREAL	mm

MCS to Spatial reference

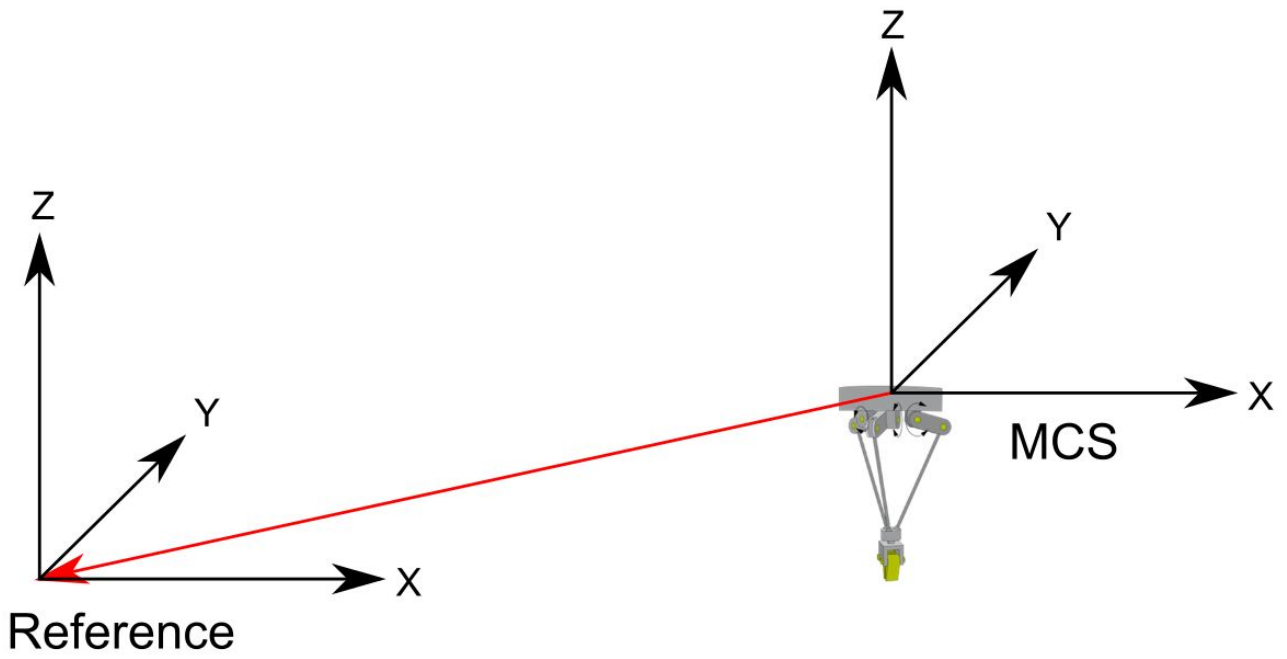
Mit dem Parameter Spatial reference kann das MCS in einem Referenzkoordinatensystem verschoben werden. Alle Koordinatensysteme sind rechtsdrehend (gegen den Uhrzeigersinn).

Parameter	Beschreibung	Typ	Einheit
Translation X	Verschiebung in X-Richtung.	LREAL	mm
Translation Y	Verschiebung in Y-Richtung.	LREAL	mm

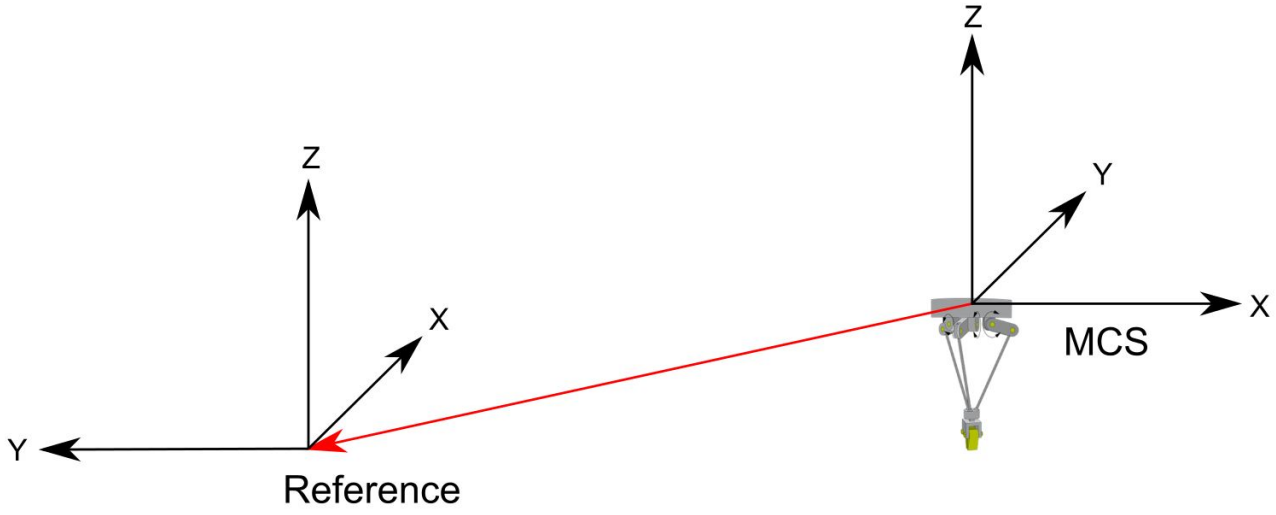
Parameter	Beschreibung	Typ	Einheit
Translation Z	Verschiebung in Z-Richtung.	LREAL	mm
Rotation 1	Winkel, um den zuerst rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation 2	Winkel, um den als zweites rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation 3	Winkel, um den als drittes rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation convention	Die Rotationskonvention gibt an, in welcher Reihenfolge um die Achsen rotiert werden soll (Parameter Rotation 1-3). Dabei geben die Buchstaben (X, Y, Z) von links nach rechts die Reihenfolge an, in der um die entsprechenden Achsen rotiert wird. Die nachfolgende Zahl gibt an, auf welchen Parameter (Rotation 1-3) der Wert zu parametrieren ist. Die translatorische Verschiebung wird immer vor der Rotation ausgeführt.	MC.CoordInterpretation_SO3	
Spatial reference	Der Parameter Spatial reference gibt an, auf welches Koordinatensystem als Basis sich das MCS bezieht. Wenn hier der Wert 0 eingestellt ist, dann wird das WCS als Basis verwendet. Wenn ein anderes Koordinatensystem als Ausgangspunkt für die Verschiebung genutzt werden soll, so kann ein Objekt Coordinate Frame [▶ 53] angelegt werden. Die Objekt-ID dieses Koordinatensystems kann über das Dropdown-Menü ausgewählt werden.	OTCID	
Definition direction	Gibt die Richtung an, in der die Verschiebung programmiert wird (aus Sicht des Bezugssystems oder aus Sicht des MCS), siehe Beispiel unten.	MC.ReferenceDefDir	

Beispiel: Definition Direction

Wenn man die Definition Direction MCS -> Reference verwendet, so wird die unten dargestellte Verschiebung vom Ausgangskoordinatensystem (MCS) zum Zielkoordinatensystem (Reference) mit negativen Vektoren angegeben.



Wenn zusätzlich zur Translation eine positive Rotation um die Z-Achse (hier 90°) angegeben wird, so wird zuerst die Translation ausgeführt und anschließend das Zielkoordinatensystem gedreht (hier +90° um die Z-Achse).



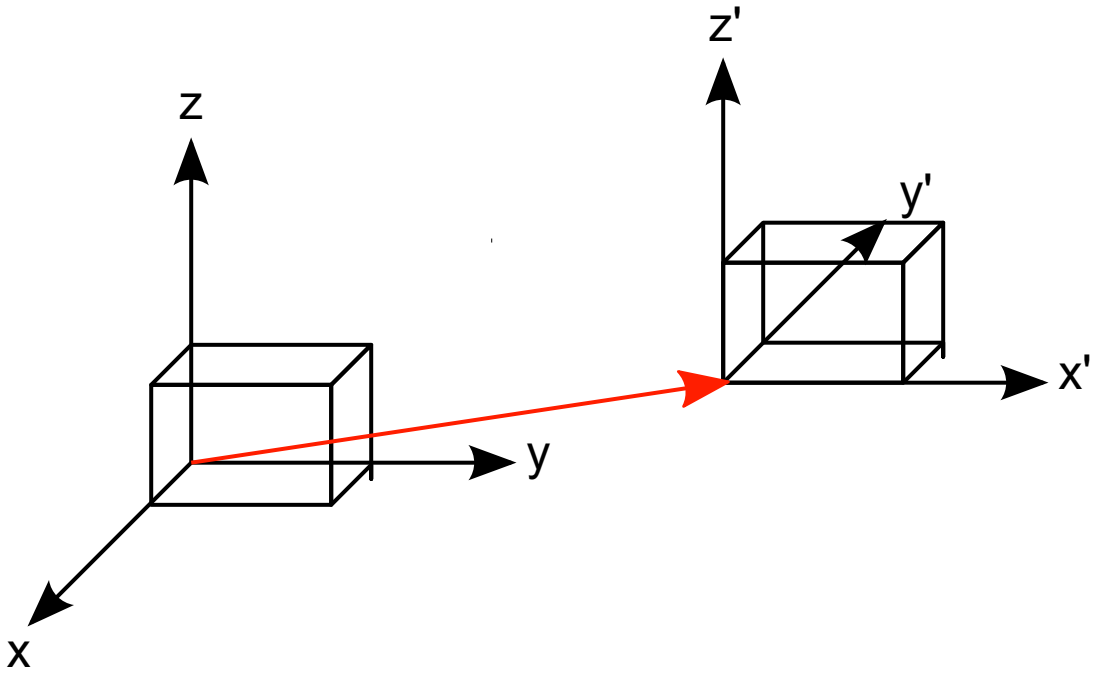
Tool offset OID

Parameter	Beschreibung	Typ	Einheit
Tool offset OID	Soll für die Kinematik ein Werkzeug definiert werden, so muss zunächst ein Werkzeugversatz (Tool Offset) [▶ 51] Objekt oder ein Werkzeug Linear (Tool Linear) [▶ 53] Objekt angelegt werden. Die Objekt-ID dieses Werkzeugs kann über das Dropdown-Menü ausgewählt werden.	OTCID	

6.2 Static Transformation

Die statische Transformation ermöglicht das Anlegen eines Kartesischen Portals. Diese unterstützt Translation und Rotation zwischen zwei kartesischen Koordinatensystemen.

Die statische Transformation bildet die Kinematik, sodass die kinematische Gruppe mit der statischen Transformation gebildet wird. Im Gegensatz dazu ergänzt ein Koordinatensystem (Coordinate Frame) [▶ 53] eine existierende Kinematik um eine Translation und eine Rotation.

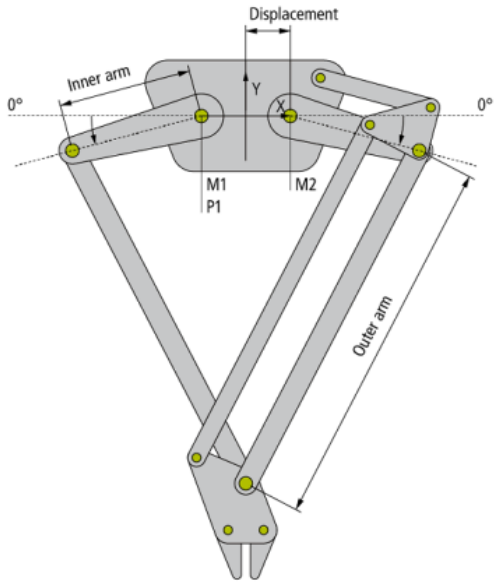


Zuerst wird die Translation berechnet, anschließend die Rotation. Die Reihenfolge der Rotationen beeinflusst die Orientierung des Koordinatensystems. Als Default für die Rotationsreihenfolge wird die in DIN 9300 beschriebene Roll-Pitch-Yaw-Regel verwendet. Die Berechnungssequenz für die Vorwärtstransformation ist Z, Y', X".

Parameter	Beschreibung	Typ	Einheit
Translation X	Verschiebung in der X-Richtung	LREAL	mm
Translation Y	Verschiebung in der Y-Richtung	LREAL	mm
Translation Z	Verschiebung in der Z-Richtung	LREAL	mm
Rotation 1	Winkel, um den als erstes rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation 2	Winkel, um den als zweites rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation 3	Winkel, um den als drittes rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	LREAL	°
Rotation convention	Die Rotationskonvention gibt an, in welcher Reihenfolge um die Achsen rotiert werden soll (Parameter Rotation 1-3). Dabei geben die Buchstaben (X, Y, Z) von links nach rechts die Reihenfolge an, in der um die entsprechenden Achsen rotiert wird. Die nachfolgende Zahl gibt an auf welchen Parameter (Rotation 1-3) der Wert zu parametrieren ist. Die translatorische Verschiebung wird immer vor der Rotation ausgeführt.	MC.CoordInterpretation_SO3	
Spatial reference	Der Parameter Spatial reference, gibt an, auf welches Koordinatensystem als Basis sich dieses Koordinatensystem bezieht. Ist hier der Wert 0 eingestellt wird das WCS als Basis verwendet. Soll ein anderes Koordinatensystem als Ausgangspunkt für die Verschiebung genutzt werden, so kann ein weiteres Objekt	OTCID	

Parameter	Beschreibung	Typ	Einheit
	Koordinatensystem (Coordinate Frame) [▶ 53] angelegt werden. Die Objekt-ID dieses Koordinatensystems kann über das Dropdown-Menü ausgewählt werden		
Definition direction	Gibt die Richtung an, in der die Verschiebung programmiert wird (aus Sicht des Bezugssystems oder aus Sicht dieses Koordinatensystems).	`MC.ReferenceDefDir`	

6.3 2D-Kinematics Type 1 (P_2C)



Die 2D-Kinematics Type 1 (P_2C) ist, wie oben im Schema gezeigt, aufgebaut.

Alle Motorachsen sind in Grad skaliert und 0° ist, wie im Schema gezeigt, definiert, wobei der Pfeil die positive Richtung anzeigt.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length	Länge von Drehpunkt zu Drehpunkt des inneren Arms	`LREAL`	mm
Outer arm length	Länge von Drehpunkt zu Drehpunkt des äußeren Arms	`LREAL`	mm
Displacement	Länge vom Mittelpunkt der Grundplatte bis zu den virtuellen Drehachsen des inneren Arms	`LREAL`	mm

Parameter für das dynamische Modell

Parameter	Beschreibung	Type	Einheit
Inner arm mass	Gesamtmasse des inneren Arms	`LREAL`	kg
Inner arm moment of inertia	Trägheitsmoment des inneren Arms in Bezug zum Drehpunkt P1, der mit dem Motor verbunden ist	`LREAL`	kg mm^2
Outer arm mass	Masse des äußeren Arms - die Gelenkmasse kann optional als eigener Parameter beschrieben werden.	`LREAL`	kg
First link mass	Masse des Gelenks, das den inneren und äußeren Arm verbindet - kann verwendet werden, wenn die Gelenkmasse nicht bereits in den äußeren und	`LREAL`	kg

Parameter	Beschreibung	Type	Einheit
	inneren Armen enthalten ist. Die Masse des Gelenks, das die Greiferplatte mit dem äußeren Arm verbindet, wird hier nicht spezifiziert. Diese kann der TcpMass hinzugefügt werden. Die Masse des ersten Gelenks bezieht sich auf den inneren Arm, der mit dem Motor 1 (M1) verbunden ist.		
Second link mass	Siehe FirstLinkMass Die Masse des zweiten Gelenks bezieht sich auf den inneren Arm, der mit dem Motor 2 (M2) verbunden ist.	LREAL	kg
TCP mass	Masse des Tool Center Point, einschließlich Greiferplatte und Greifer. Die Nutzlast wird normalerweise mit Hilfe eines getrennten Parameters beschrieben.	LREAL	kg
First drive torque OID	Objekt-ID des ersten Antriebsdrehmoments (siehe hier [▶ 51]) Wenn die Motoren und Getriebe aller Motoren sich ähnlich verhalten, können alle Antriebsdrehmomente mithilfe einer OID dargestellt werden. Somit verweisen beide Parameter auf die gleiche Objekt-ID.	OTCID	
Second drive torque OID	Objekt-ID des zweiten Antriebsdrehmoments	OTCID	

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

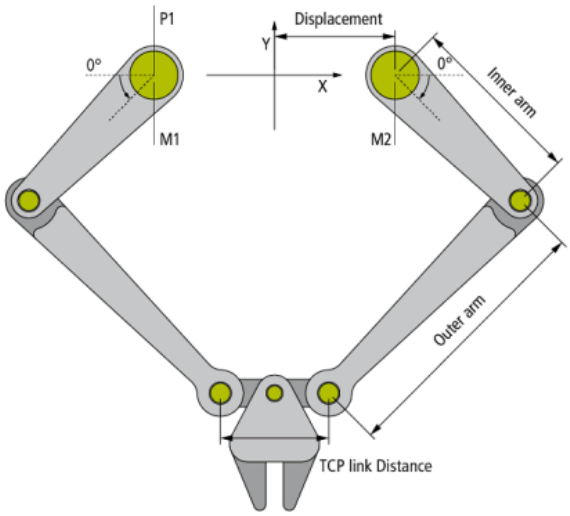
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.4 2D-Kinematics Type 2 (P_2C2)



Die 2D-Kinematics Type 2 (P_2C2) ist, wie oben im Schema gezeigt, aufgebaut.

Alle Motorachsen sind in Grad skaliert und 0° ist, wie im Schema gezeigt, definiert, wobei der Pfeil die positive Richtung anzeigt.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length	Länge von Drehpunkt zu Drehpunkt des inneren Arms	LREAL	mm
Outer arm length	Länge von Drehpunkt zu Drehpunkt des äußeren Arms	LREAL	mm
Displacement	Länge vom Mittelpunkt der Grundplatte bis zu den virtuellen Drehachsen des inneren Arms	LREAL	mm
TCP link distance	Distanz von Drehpunkt zu Drehpunkt des äußeren Arms	LREAL	mm

Parameter für das dynamische Modell

Parameter	Beschreibung	Typ	Einheit
Inner arm mass	Gesamtmasse des inneren Arms	LREAL	kg
Inner arm moment of inertia	Trägheitsmoment des inneren Arms in Bezug zum Drehpunkt P1, der mit dem Motor verbunden ist.	LREAL	kg mm^2
Outer arm mass	Masse des äußeren Arms - die Gelenkmasse kann optional als eigener Parameter beschrieben werden.	LREAL	kg
First link mass	Masse des Gelenks, das den inneren und äußeren Arm verbindet. Kann verwendet werden, wenn die Gelenkmasse nicht bereits in den äußeren und inneren Armen enthalten ist. Die Masse des Gelenks, das die Greiferplatte mit dem äußeren Arm verbindet, wird hier nicht spezifiziert. Diese kann der TCP mass hinzugefügt werden. Die Masse des ersten Gelenks bezieht sich auf den inneren Arm, der mit dem Motor 1 (M1) verbunden ist.	LREAL	kg

Parameter	Beschreibung	Typ	Einheit
Second link mass	siehe First link mass Die Masse des zweiten Gelenks bezieht sich auf den inneren Arm, der mit dem Motor 2 (M2) verbunden ist.	LREAL	kg
TCP mass	Masse des TCP, einschließlich Greiferplatte und Greifer - die Nutzlast wird normalerweise mit Hilfe eines getrennten Parameters beschrieben.	LREAL	kg
First drive torque OID	Objekt-ID des ersten Antriebsdrehmoments (siehe hier [▶ 51]) Wenn die Motoren und Getriebe aller Motoren sich ähnlich verhalten, können alle Antriebsdrehmomente mithilfe einer OID dargestellt werden. Somit verweisen beide Parameter auf die gleiche Objekt-ID.	OTCID	
Second drive torque OID	Objekt-ID des zweiten Antriebsdrehmoments	OTCID	

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

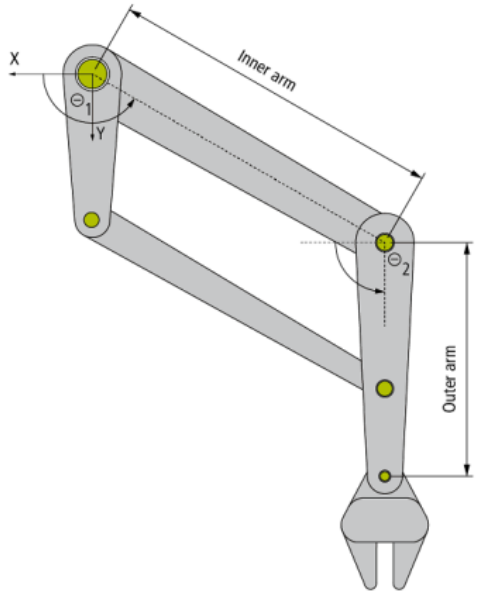
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.5 2D-Kinematics Type 3 (S_CC)



Die 2D-Kinematics Type 3 (S_CC) ist, wie oben im Schema gezeigt, aufgebaut.

Alle Motorachsen sind in Grad skaliert und 0° ist, wie im Schema gezeigt, definiert, wobei der Pfeil die positive Richtung anzeigt.

Dieser Kinematiktyp ist als linkshändig implementiert. Die Wellen von Motor 1 (M1) und Motor 2 (M2) befinden sich im Ursprung des Koordinatensystems.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length	Länge von der Motorwelle zum Drehpunkt des äußeren Arms	LREAL	mm
Outer arm length	Länge vom Drehpunkt zum Tool Center Point des äußeren Arms	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

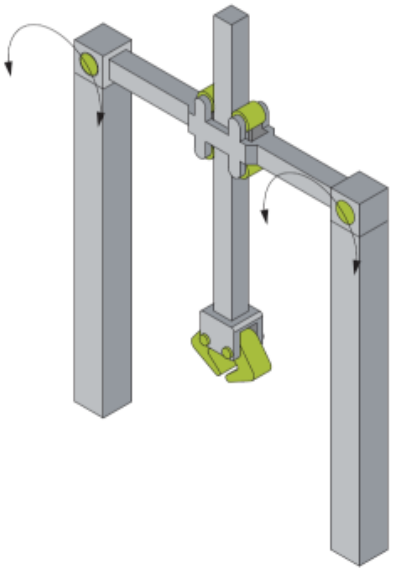
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.6 2D-Kinematics H-Bot (P_2Y)



Die 2D-Kinematics H-Bot (P_2Y) ist, wie oben im Schema gezeigt, aufgebaut.

Die Motorachsen müssen in Millimetern skaliert werden. Alle anderen Positionsparameter ergeben sich aus den kinematischen Zwangsbedingungen.

Der Ursprungspunkt des Maschinenkoordinatensystems MCS ist durch den Punkt definiert, für den die Positionen der beiden Motoren gleich null sind.

Parameter für das dynamische Modell

Parameter	Beschreibung	Typ
FirstDriveTorqueOID	Objekt-ID des ersten Antriebsdrehmoments (siehe hier [▶ 51]). Wenn die Motoren und Getriebe aller Motoren sich ähnlich verhalten, können alle Antriebsdrehmomente mithilfe einer OID dargestellt werden. Somit verweisen beide Parameter auf die gleiche Objekt-ID.	OTCID
SecondDriveTorqueOID	Objekt-ID des zweiten Antriebsdrehmoments	OTCID

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

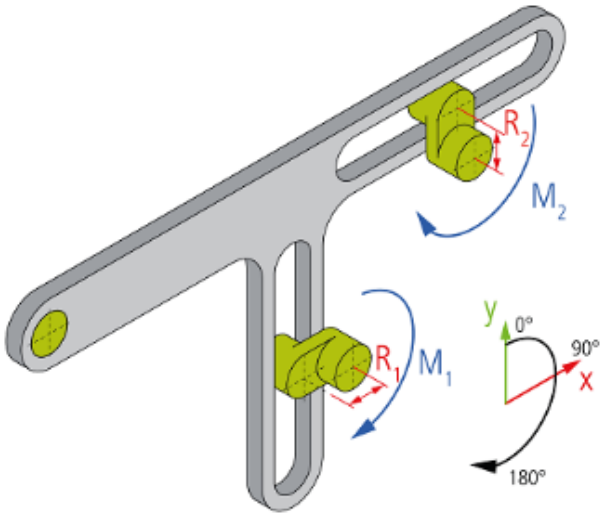
Für alle Kinematiken mit Tool gilt zudem:

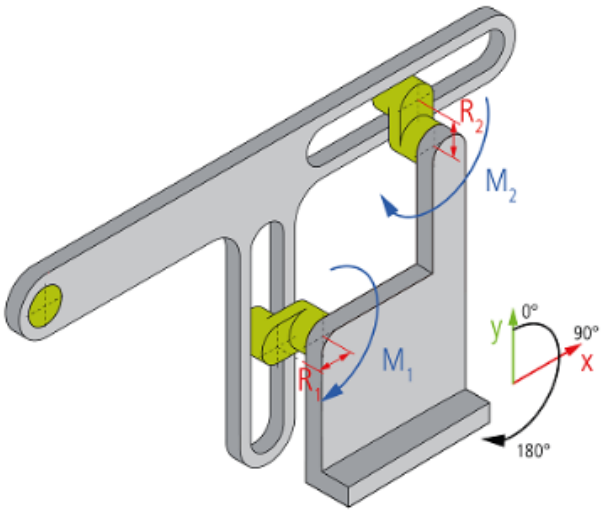
- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.7 2D-Kinematics Type 5 (S_CC)





Eine Kurbel besteht aus einem Rad mit einem exzentrisch angeordnetem Knopf. Zwei Kurbeln, deren Ende in Führungsschienen münden, ermöglichen zweidimensionale Bewegungen des TCP. Die Kurbeln werden von Motoren bewegt, die ortsfest in einem Sockel verbaut sind.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Radius R1	• Die Größe R1 beschreibt den Hebelarm der Kurbel 1. Dieser Hebelarm wird vom Mittelpunkt der Kurbel bis zur Mitte des Knopfes in der Führungsschiene gemessen. • Die Rotationsachse von Kurbel 1 ist fest. • Kurbel 1 wird von Motor M1 angetrieben. • Motor M1 bewirkt Bewegungen in die x-Richtung des TCP.	LREAL	mm
Radius R2	• Die Größe R2 beschreibt den Hebelarm der Kurbel 2. Dieser Hebelarm wird vom Mittelpunkt der Kurbel bis zur Mitte des Knopfes in der Führungsschiene gemessen. • Die Rotationsachse von Kurbel 2 ist fest. • Kurbel 2 wird von Motor M2 angetrieben. • Motor M2 bewirkt Bewegungen in die y-Richtung des TCP.	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

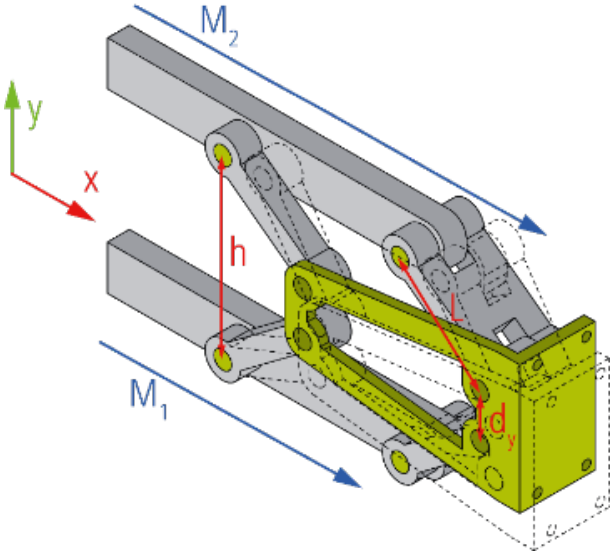
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.8 2D-Kinematics Type 6 (P_2X)



Über die Kinematik ermöglichen die beiden Linearachsen M_1 und M_2 Bewegungen in der XY-Ebene.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Flange translation X	Räumliche Verschiebung in X-Richtung.	LREAL	mm
Arm length (L)	Länge der Verbindungssegmente jeweils vom Motor zum Gelenk des Flansches gemessen.	LREAL	mm
Motor distance (h)	Der Motorabstand h ist der y-Abstand der Gelenke an den Motoren. (Die Motoren M1 und M2 verfahren beide in X-Richtung. Der Motorabstand h wird von Gelenkmittelpunkt zu Gelenkmittelpunkt gemessen.)	LREAL	mm
Link distance (dy)	Die Link Distance dy ist der Abstand zwischen den Gelenken im Flansch.	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

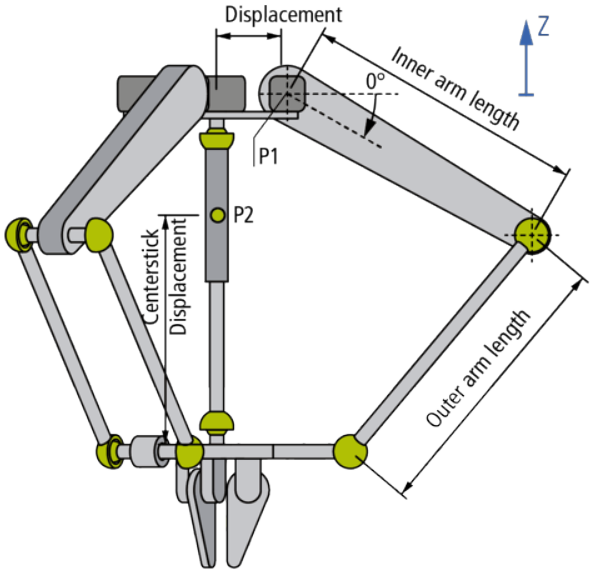
Für alle Kinematiken mit Tool gilt zudem:

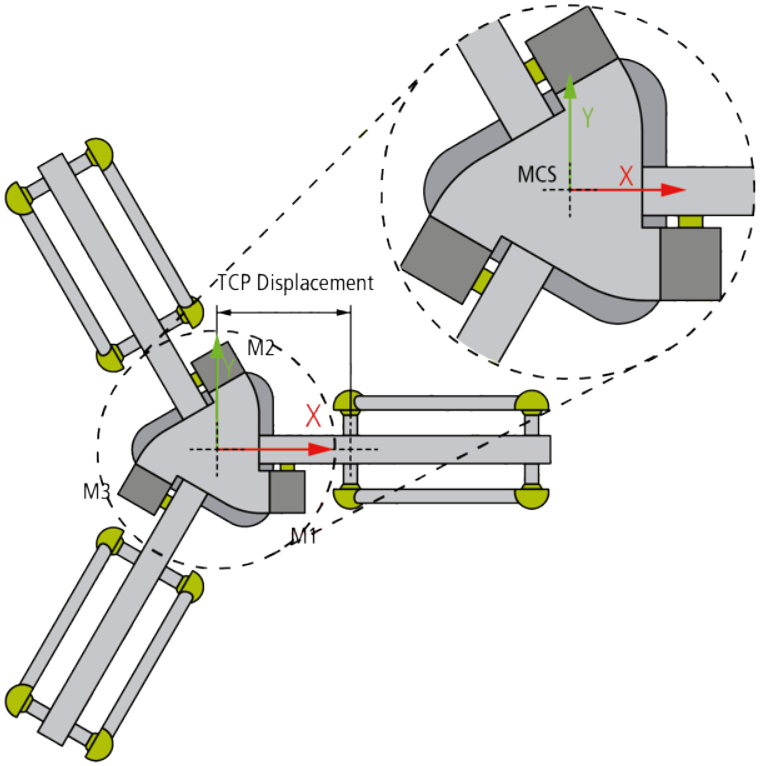
- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TC3 Kinematic Transformation (Level 2)

6.9 3D-Delta Type 1 (P_3C)





Der 3D-Delta Type 1 (P_3C) ist, wie oben im Schema gezeigt, aufgebaut. Die kinematische Transformation erwartet Kugelgelenke (oder Elemente mit gleichem Verhalten) in der Verbindung der Arme und mit der unteren Platte.

Der Zentrierstab für die Ausrichtung des Greifers kann optional parametriert werden.

Alle Motorachsen sind in Grad skaliert und 0° ist, wie oben im Schema gezeigt, definiert, wobei der Pfeil die positive Richtung anzeigt. Das gilt für alle 3 Motoren.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length	Länge von Drehpunkt zu Drehpunkt des inneren Arms - das ist der Arm, der direkt mit dem Motor verbunden ist.	LREAL	mm
Outer arm length	Länge von Drehpunkt zu Drehpunkt des äußeren Arms	LREAL	mm
Displacement	Länge vom Mittelpunkt der Grundplatte bis zu den virtuellen Drehachsen des inneren Arms	LREAL	mm
TCP displacement	Länge vom Mittelpunkt der Greiferplatte bis zu den virtuellen Drehachsen des äußeren Arms	LREAL	mm

Parameter für das dynamische Modell

Parameter	Beschreibung	Typ	Einheit
Inner arm mass	Gesamtmasse des inneren Arms	LREAL	kg
Inner arm moment of inertia	Trägheitsmoment des inneren Arms in Bezug zum Drehpunkt P1, der mit dem Motor verbunden ist	LREAL	kg mm^2
Outer arm mass	Masse des äußeren Arms. Sind zwei Stäbe vorhanden, wird die Gesamtmasse benötigt. Die Gelenkmasse kann optional als eigener Parameter beschrieben werden.	LREAL	kg
Link mass	Masse des Gelenks, das den inneren und äußeren Arm verbindet. Kann verwendet werden, wenn die Gelenkmasse nicht bereits in den äußeren und inneren Armen enthalten ist. Die Masse des Gelenks, das die Greiferplatte mit dem äußeren Arm verbindet, wird hier nicht spezifiziert. Diese kann der TcpMass hinzugefügt werden.	LREAL	kg
TCP mass	Masse des TCP, einschließlich Greiferplatte und Greifer. Die Nutzlast wird normalerweise mit Hilfe eines getrennten Parameters beschrieben.	LREAL	kg
Center stick mass	Gesamtmasse des Zentrierstabs	LREAL	kg
Center stick: moment of inertia	Trägheitsmoment des Zentrierstabs in Bezug auf den Schwerpunkt (P2)	LREAL	kg mm^2
Center stick: center of mass displacement	Länge von Greiferplatte zum Schwerpunkt des Stabs	LREAL	mm
First drive torque OID	Objekt-ID des ersten Antriebsdrehmoments (siehe hier) Wenn die Motoren und Getriebe aller Motoren sich ähnlich verhalten, können alle Antriebsdrehmomente mithilfe einer OID dargestellt werden. Somit verweisen alle 3 Parameter auf die gleiche Objekt-ID.	OTCID	
Second drive torque OID	Objekt-ID des zweiten Antriebsdrehmoments	OTCID	
Third drive torqueOID	Objekt-ID des dritten Antriebsdrehmoments	OTCID	

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

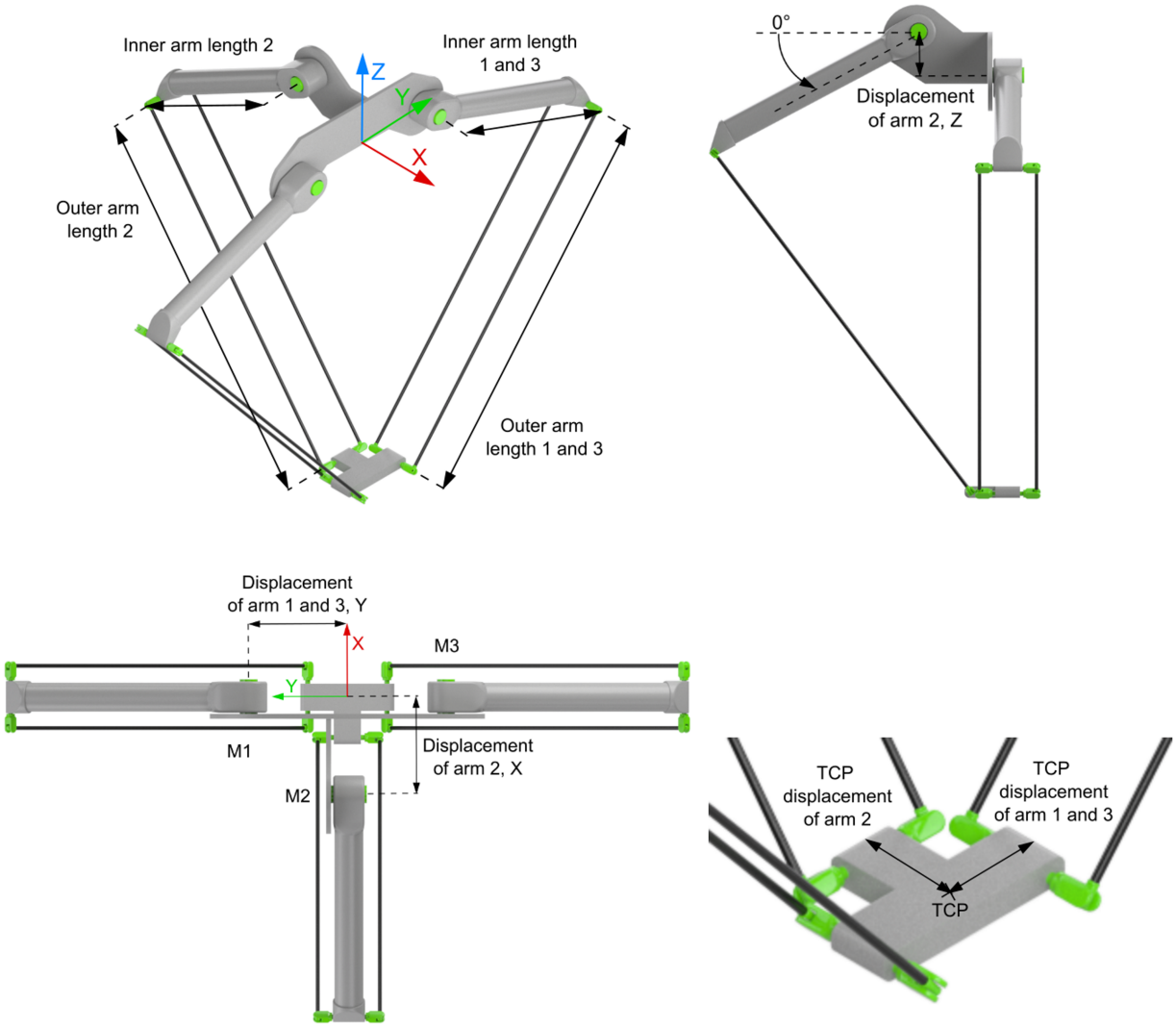
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.10 3D-Delta T Type 3 (P_3C3)



Der 3D-Delta T Type 3 (P_3C3) ist, wie oben im Schema gezeigt, aufgebaut. Zwei Arme sind direkt gegenüberliegend, der dritte Arm im 90 Grad Winkel zu diesen. Durch die Armanordnung ist es möglich, zwei Roboter dieses Typs sehr nah aneinander zu platzieren.

Das Maschinenkoordinatensystem (MCS) befindet sich mittig zwischen den beiden gegenüberliegenden Armen auf Höhe der Motoren M1 und M3.

Alle Motorachsen sind in Grad skaliert und 0° ist, wie im Schema gezeigt, definiert, wobei der Pfeil die positive Drehrichtung anzeigt. Das gilt für alle drei Motoren.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length 1 and 3	Arm 1, Arm 3: Länge von Drehpunkt zu Drehpunkt des inneren Arms (direkt mit dem Motor verbunden)	LREAL	mm
Inner arm length 2	Arm 2: Länge von Drehpunkt zu Drehpunkt des inneren Arms (direkt mit dem Motor verbunden)	LREAL	mm
Outer arm length 1 and 3	Arm 1, Arm 3: Länge von Drehpunkt zu Drehpunkt des äußeren Arms	LREAL	mm
Outer arm length 2	Arm 2: Länge von Drehpunkt zu Drehpunkt des äußeren Arms	LREAL	mm
Displacement of arm 1 and 3, Y	Arm 1, Arm 3: Jeweils der Abstand vom MCS-Ursprung zur Motorachse	LREAL	mm
Displacement of arm 2, X	Arm 2: Der Abstand vom MCS-Ursprung zur Motorachse	LREAL	mm
Displacement of arm 2, Z	Arm 2: Der Abstand vom MCS-Ursprung zur Motorachse	LREAL	mm
TCP displacement of arm 1 and 3	Arm 1, Arm 3: Länge vom Mittelpunkt der Greiferplatte bis zu den virtuellen Drehachsen des äußeren Arms	LREAL	mm
TCP displacement of arm 2	Arm 2: Länge vom Mittelpunkt der Greiferplatte bis zu der virtuellen Drehachse des äußeren Arms	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

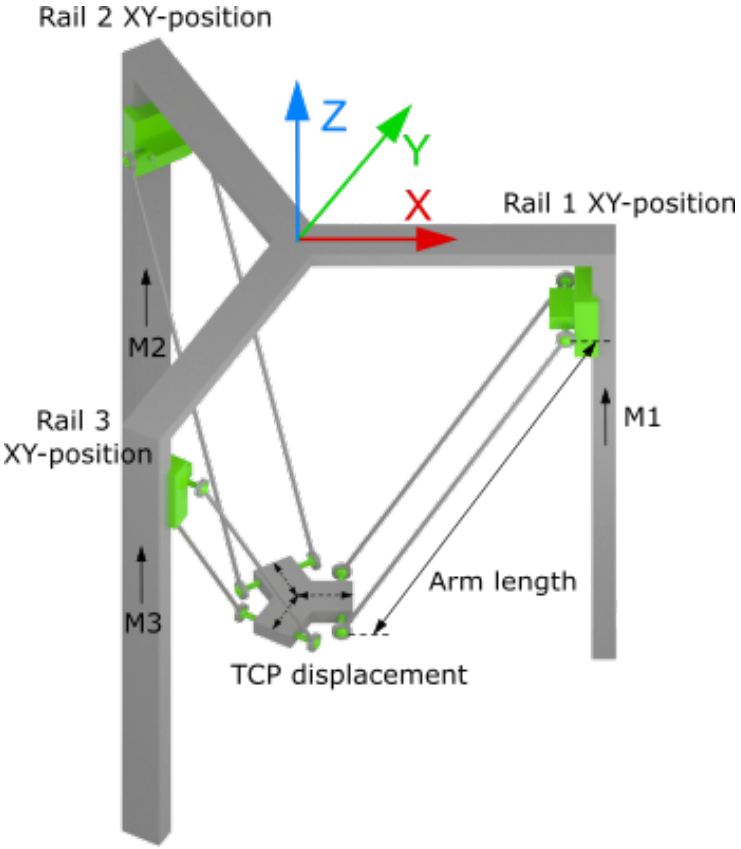
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspaket	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.7 TF5400 TC3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.11 3D-Tripod Type 1 (P_3Z)



Der 3D-Tripod Type 1 (P_3Z) ist, wie oben im Schema gezeigt, aufgebaut.

Alle linear Achsen (ACS) sind in mm skaliert.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Arm length	Armlänge von Drehpunkt zu Drehpunkt	LREAL	mm
Rail 1 X-position	X-Position des 1. Rails in Relation zum MCS	LREAL	mm
Rail 1 Y-position	Y-Position des 1. Rails in Relation zum MCS	LREAL	mm
Rail 2 X-position	X-Position des 2. Rails in Relation zum MCS	LREAL	mm
Rail 2 Y-position	Y-Position des 2. Rails in Relation zum MCS	LREAL	mm
Rail 3 X-position	X-Position des 3. Rails in Relation zum MCS	LREAL	mm
Rail 3 Y-position	Y-Position des 3. Rails in Relation zum MCS	LREAL	mm
TCP displacement	Länge vom Mittelpunkt der Greiferplatte bis zu der virtuellen Drehachse des Arms	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

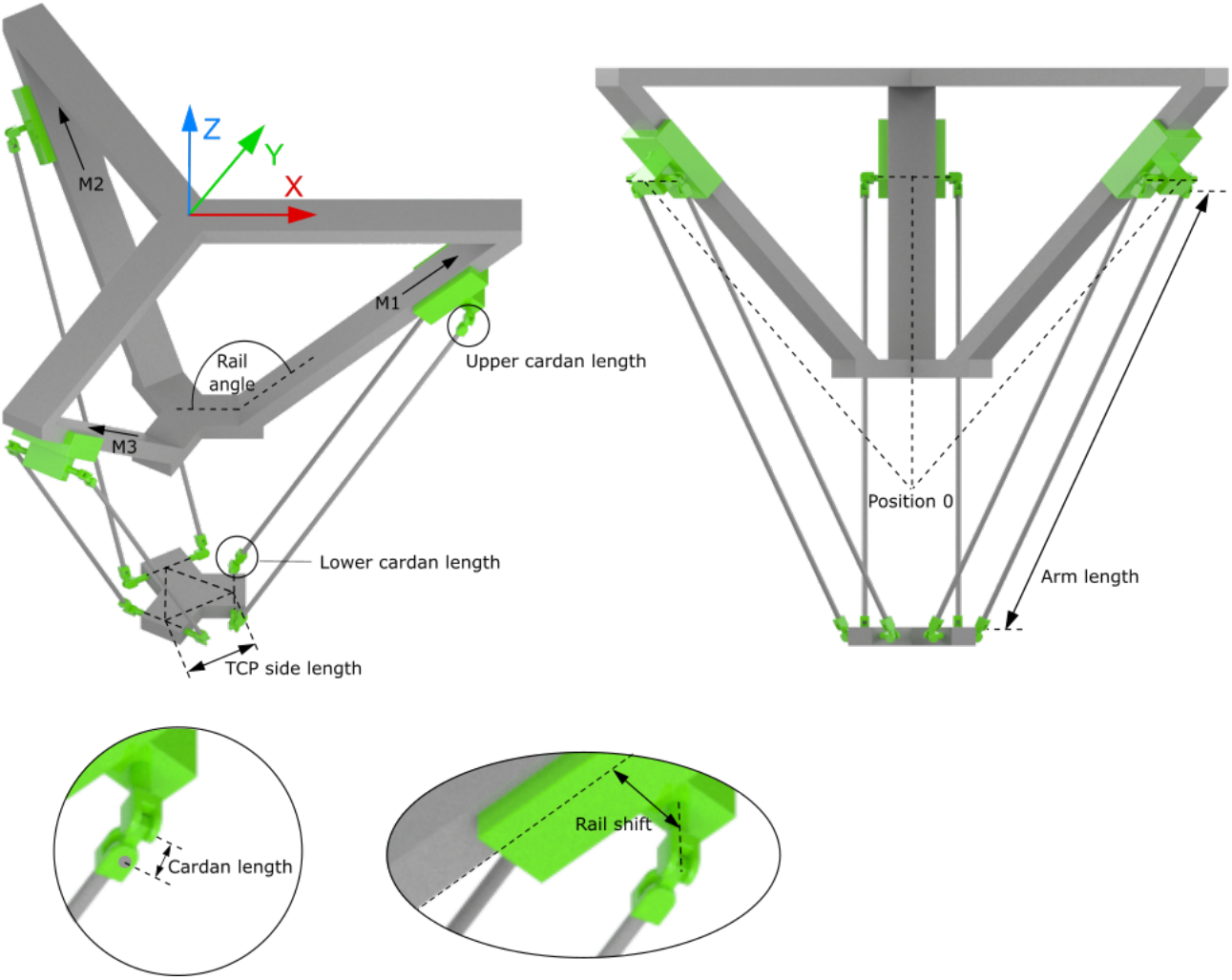
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.24 TF5400 TC3 Advanced Motion Pack V3.1.10.66	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.12 3D-Tripod Type 2 (P_3L)



Der 3D-Tripod Type 2 (P_3L) ist, wie oben im Schema gezeigt, aufgebaut.

Alle linear Achsen (ACS) sind in Millimeter (mm) skaliert. Die 0-Position der Achsen ist nur ein „virtueller“ Punkt, welcher nicht angefahren werden kann. Eine positive Geschwindigkeit der Motoren bewegt das Tool aufwärts, sodass die linearen Achsen keine negative Position erreichen können.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Arm length	Armlänge von Drehpunkt zu Drehpunkt	LREAL	mm
Rail angle	Winkel, in dem die Führungsschienen der Linearmotoren angebracht sind.	LREAL	°
Rail shift	Versatz der Armaufhängungspunkte zu den Führungsschienen der Linearmotoren.	LREAL	mm

Parameter	Beschreibung	Typ	Einheit
Upper cardan length	Wird an den oberen Armaufhängungspunkten ein Kardan-Gelenk verwendet, so kann über diesen Parameter der Versatz der beiden Gelenke innerhalb des Kardan-Gelenks angegeben werden. Bei Verwendung eines Kugelgelenkes ist die Länge 0 einzutragen.	LREAL	mm
Lower cardan length	Wird an den unteren Armaufhängungspunkten ein Kardan-Gelenk verwendet, so kann über diesen Parameter der Versatz der beiden Gelenke innerhalb des Kardan-Gelenks angegeben werden. Bei Verwendung eines Kugelgelenkes ist die Länge 0 einzutragen.	LREAL	mm
TCP side length	Seitenlänge des virtuellen Dreiecks im TCP.	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

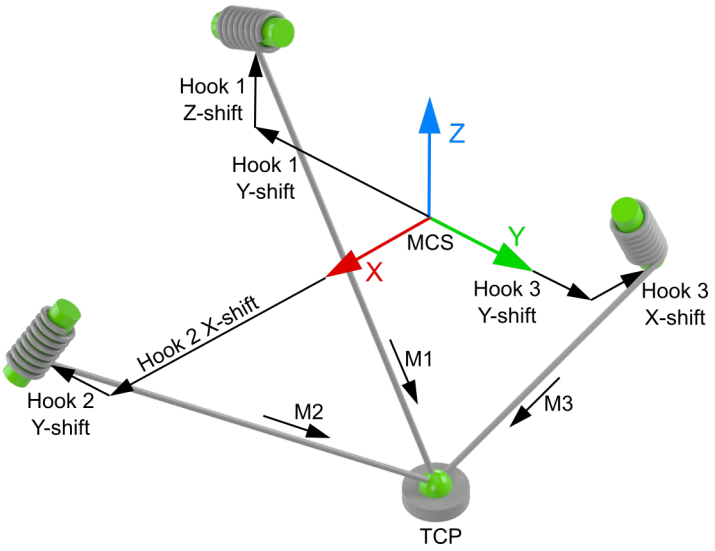
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.24 TF5400 TC3 Advanced Motion Pack V3.1.10.66	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.13 3D-Cable Kinematics Type 2 (P_3L)



Die 3D-Cable Kinematics Type 2 (P_3L) ist, wie oben im Schema gezeigt, aufgebaut.

Der Nullpunkt des Maschinenkoordinatensystems (MCS) kann dabei an einer beliebigen Stelle im Raum liegen. Von MCS-Ursprung aus werden die Aufhängungspunkte („Hooks“) der Kabel/Seile definiert.

Alle Motorachsen sind in Millimetern skaliert, wobei der Pfeil die positive Richtung anzeigt.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Hook 1	Aufhängungspunkt des ersten Kabels		
X-shift	X-Position des Hook 1 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 1 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 1 in Relation zum MCS	LREAL	mm
Hook 2	Aufhängungspunkt des zweiten Kabels		
X-shift	X-Position des Hook 2 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 2 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 2 in Relation zum MCS	LREAL	mm
Hook 3	Aufhängungspunkt des mittleren Kabels		
X-shift	X-Position des Hook 3 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 3 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 3 in Relation zum MCS	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

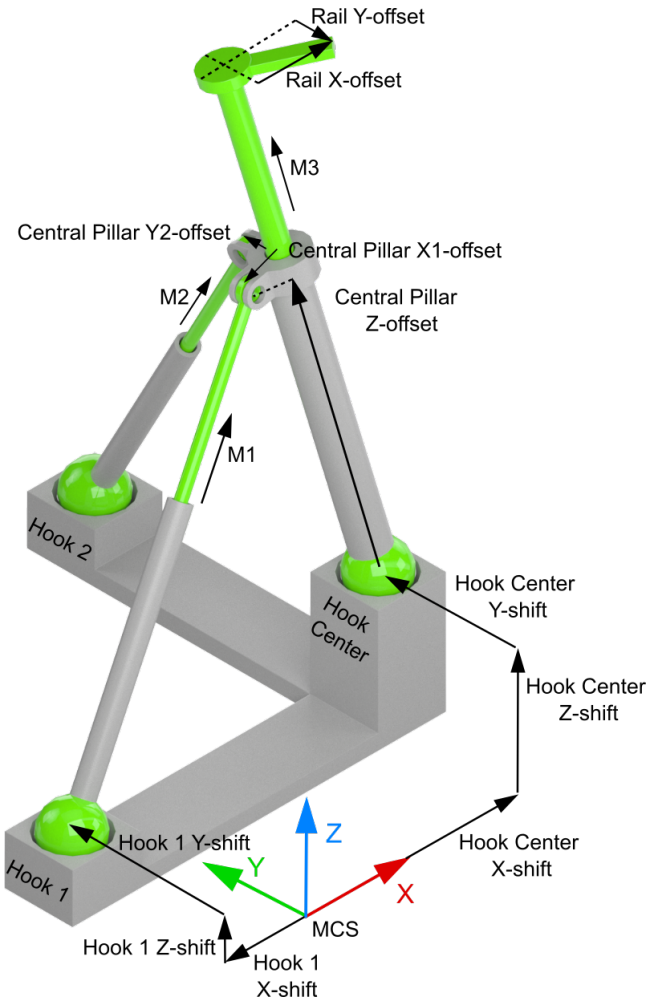
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspaket	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.7 TF5400 TC3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.14 3D-Kinematics Type 7 (PXX_SZ)



Die 3D-Kinematik Type 7 (PXX SZ) ist, wie oben im Schema gezeigt, aufgebaut.

Der Nullpunkt des Maschinenkoordinatensystems (MCS) kann dabei an einer beliebigen Stelle im Raum liegen. Vom MCS-Ursprung aus werden die Aufhängungspunkte („Hooks“) der drei Arme definiert. Zur Definition der offsets wird beim zentralen Arm („Central Pillar“) von der Position parallel zur Z-Achse des MCS ausgegangen, sodass die feste Länge des zentralen Arms bis zur Aufhängung der beiden anderen Arme das „Central Pillar Z-offset“ ist. Am Ende des zentralen Arms, kann sich noch ein Ausleger („Rail“) befinden, dessen Spitze durch das „Rail-offset“ angegeben wird.

Alle Motorachsen sind in Millimetern skaliert, wobei der Pfeil die positive Richtung anzeigt. Der Nullpunkt der Motorachsen M1 und M2 befindet sich im jeweiligen Aufhängungspunkt Hook1 bzw. Hook 2. Der Nullpunkt der Motorachse M3 befindet sich hingegen im Central Pillar Z-offset.

Parameters Joint Hooks

Parameter	Beschreibung	Typ	Einheit
Hook 1	Aufhängungspunkt des ersten Arms		
X-shift	X-Position des Hook 1 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 1 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 1 in Relation zum MCS	LREAL	mm
Hook 2	Aufhängungspunkt des zweiten Arms		
X-shift	X-Position des Hook 2 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 2 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 2 in Relation zum MCS	LREAL	mm
Hook Center	Aufhängungspunkt des zentralen Arms		

Parameter	Beschreibung	Typ	Einheit
X-shift	X-Position des Hook Center in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook Center in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook Center in Relation zum MCS	LREAL	mm

Parameters Central Pillar

Die Parameter zum zentralen Arm (central pillar) inkl. des Auslegers (rail) werden im Bezug zur Armposition parallel zur Z-Achse angegeben.

Parameter	Beschreibung	Typ	Einheit
X1-offset	Abstand des Aufhängungspunktes des ersten Seitenarms zum zentralen Arm in X-Richtung	LREAL	mm
X2-offset	Abstand des Aufhängungspunktes des zweiten Seitenarms zum zentralen Arm in X-Richtung	LREAL	mm
Y1-offset	Abstand des Aufhängungspunktes des ersten Seitenarms zum zentralen Arm in Y-Richtung	LREAL	mm
Y2-offset	Abstand des Aufhängungspunktes des zweiten Seitenarms zum zentralen Arm in Y-Richtung	LREAL	mm
Z-offset	Abstand vom Aufhängungspunkt des zentralen Arms bis zur Verbindung der Seitenarme an den zentralen Arm	LREAL	mm
Rail offset			
X-rail offset	Abstand vom Mittelpunkt der Spitze des zentralen Arms bis zur Spitze des Auslegers in X-Richtung	LREAL	mm
Y-rail offset	Abstand vom Mittelpunkt der Spitze des zentralen Arms bis zur Spitze des Auslegers in Y-Richtung	LREAL	mm
Z-rail offset	Abstand vom Mittelpunkt der Spitze des zentralen Arms bis zur Spitze des Auslegers in Z-Richtung	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

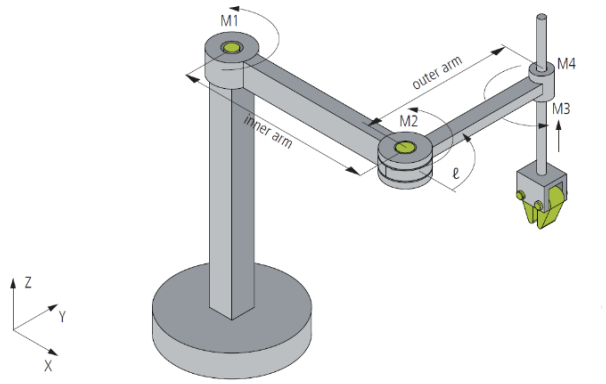
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspaket	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.7 TF5400 TC3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.15 4D-SCARA (S_CCZC)



Die 4D-SCARA (**S**elective **C**ompliance **A**ssembly **R**obot **A**rm) Kinematics (S_CCZC) ist, wie oben im Schema gezeigt, aufgebaut.

Die Motorachsen 1, 2 und 4 sind in Grad skaliert, wobei die positive Drehrichtung in Pfeilrichtung ist. Die dritte Motorachse ist in Millimetern skaliert.

Der Nullpunkt des MCS befindet sich im ersten Gelenk (M1). Die X-Achse wird vom SCARA-Arm bestimmt, wenn alle Drehmotorachsen bei 0° stehen.

> Die Strecklage des SCARA-Arms (alle Drehmotorachsen auf Position 0°) kann nicht im kartesischen Modus angefahren werden, weil sich der Roboter dort in einer singulären Position befindet. Ein Anfahren dieser Positionen ist nur im Achs-Modus (Direct Mode) möglich.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Inner arm length	Länge von Drehpunkt zu Drehpunkt des inneren Arms; dieses ist der Arm auf der Nullpunktseite.	LREAL	mm
Outer arm length	Länge von Drehpunkt zu Drehpunkt des äußeren Arms; dieses ist der Arm auf der TCP-Seite.	LREAL	mm
Gear coupling	Kopplungsfaktor zwischen den Achsen M4 und M3.	LREAL	mm/°
Flange rotation A	Drehwinkel um die lokale X-Achse herum.	LREAL	°
Tool offset OID	Objekt ID eines Werkzeugs, das am Kinematik-Flansch befestigt ist. Das Flansch-Koordinatensystem ist 180° um die X-Achse gedreht, so dass seine Z-Achse nach unten zeigt.	OTCID	

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

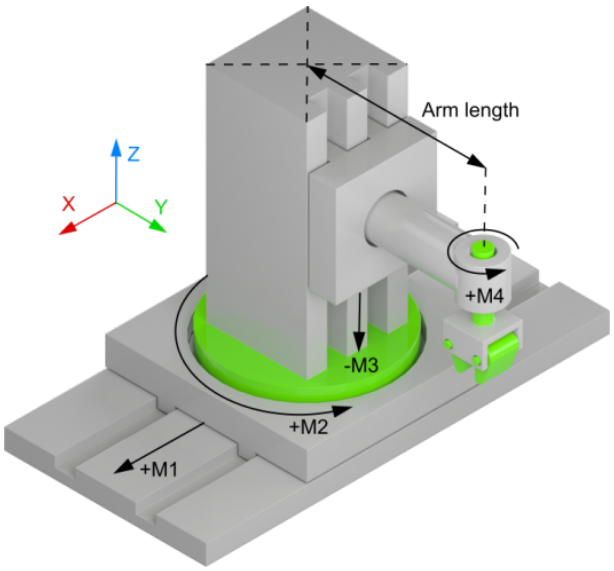
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.16 4D-Kinematics Type 6 (S_XCZC)



Die 4D-Kinematics Type 6 (S_XCZC) beschreibt eine serielle kinematische Transformation, die, wie oben im Schema gezeigt, aufgebaut ist.

Die Motorenachsen M2 und M4 sind in Grad skaliert, wobei die positive Drehrichtung in Pfeilrichtung ist.

Die Motorachse M1 und M3 sind in Millimetern skaliert. In Bezug zum MCS gibt M1 eine Bewegung auf der X-Achse und M3 eine Bewegung auf der Z-Achse an. Der Ursprung des MCS-Koordinatensystems befindet sich auf der Linearachse M1 im Gelenk M2.

Singuläre Positionen anfahren

Singuläre Positionen, wie bei diesem Robotertyp z. B. M2=+-90°, können im kartesischen Modus nicht angefahren werden. Ein Anfahren dieser Positionen ist nur im Achs-Modus (Direct Mode) möglich.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Arm length	Abstand von der Drehachse M2 zur Drehachse M4 Arm length > 0	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

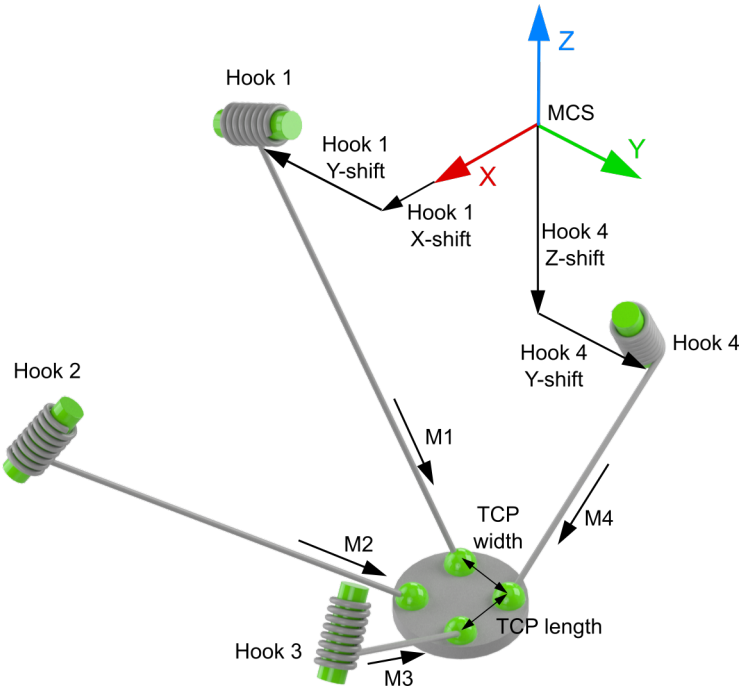
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspaket	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.7 TF5400 TC3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.17 4D-Cable Kinematics (P_4L)



Die 4D-Cable Kinematics (P_4L) ist, wie oben im Schema gezeigt, aufgebaut.

Der Nullpunkt des Maschinenkoordinatensystems (MCS) kann dabei an einer beliebigen Stelle im Raum liegen. Von MCS-Ursprung aus werden die Aufhängungspunkte („Hooks“) der Kabel/Seile definiert.

Alle Motorachsen sind in Millimetern skaliert, wobei der Pfeil die positive Richtung anzeigt.

Parameter für Joint Hooks

Parameter	Beschreibung	Typ	Einheit
Hook 1	Aufhängungspunkt des ersten Kabels		
X-shift	X-Position des Hook 1 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 1 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 1 in Relation zum MCS	LREAL	mm
Hook 2	Aufhängungspunkt des zweiten Kabels		
X-shift	X-Position des Hook 2 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 2 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 2 in Relation zum MCS	LREAL	mm
Hook 3	Aufhängungspunkt des dritten Kabels		
X-shift	X-Position des Hook 3 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 3 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 3 in Relation zum MCS	LREAL	mm
Hook 4	Aufhängungspunkt des vierten Kabels		
X-shift	X-Position des Hook 4 in Relation zum MCS	LREAL	mm
Y-shift	Y-Position des Hook 4 in Relation zum MCS	LREAL	mm
Z-shift	Z-Position des Hook 4 in Relation zum MCS	LREAL	mm
TCP	Tool Center Point		
TCP length	Abstand der Aufhängungspunkte am TCP entlang der X-Achse	LREAL	mm
TCP width	Abstand der Aufhängungspunkte am TCP entlang der Y-Achse	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

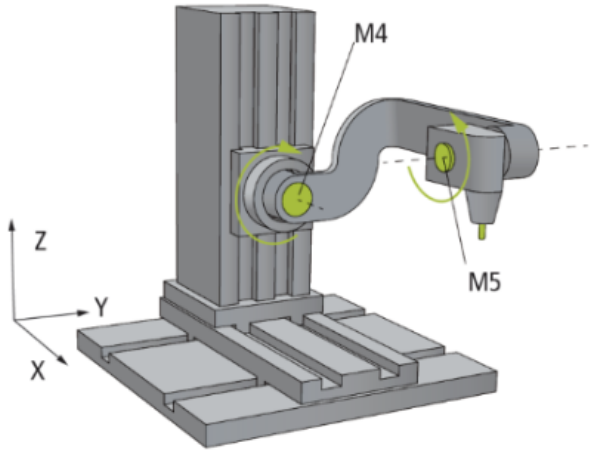
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspaket	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.7 TF5400 TC3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

6.18 5D-Kinematics Type 2 (XYZab)



Die 5D-Kinematics Type 2 (XYZab) ist, wie oben im Schema gezeigt, aufgebaut.

Die Motoren 1 bis 3 (X, Y, Z) sind in Millimetern skaliert. Die Motoren 4 und 5 sind in Grad skaliert. Die 0° Position ist die in der Zeichnung dargestellte Achsenposition; die Pfeile zeigen die positive Drehrichtung an.

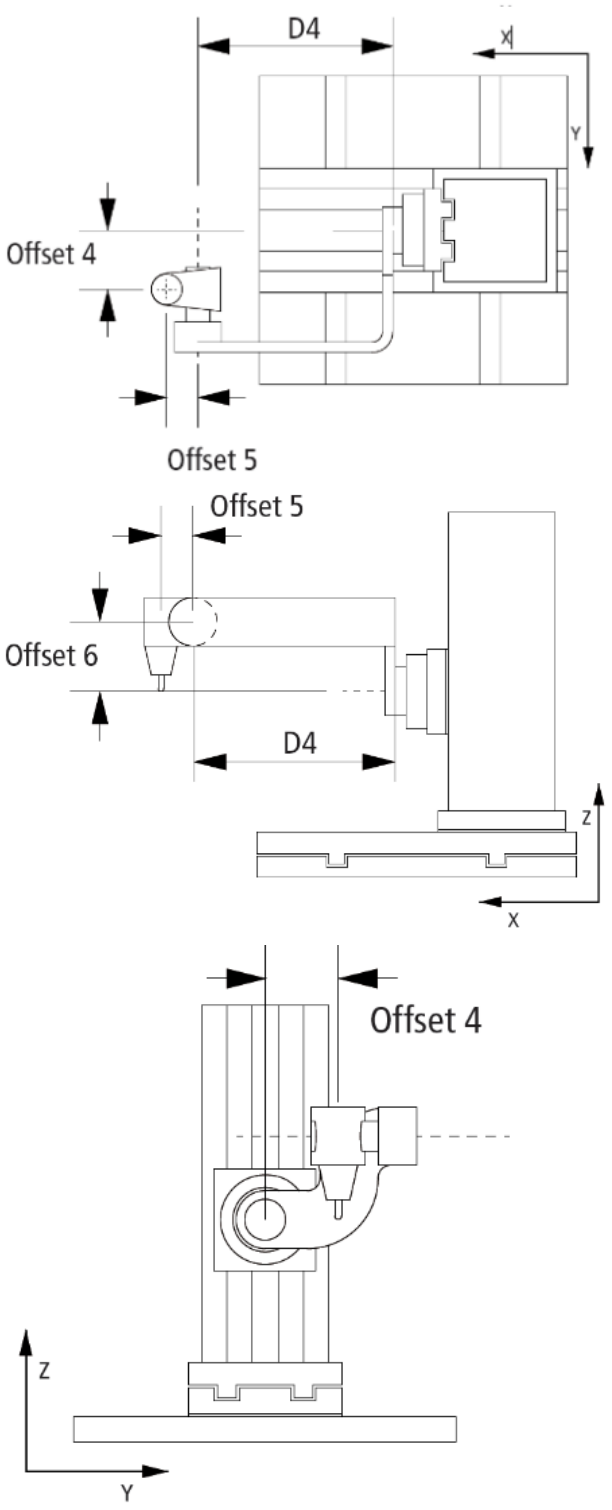
Unterschied von Typ 2

Die 5D-Kinematics Typ 2 unterscheiden sich von den 5D-Kinematics Typ 3 in der Orientierung der positiven Richtung der Achsrotation um die Motorachsen M4 und M5.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Handle D4	Armlänge in X-Richtung von Motorachse 4 zur Motorachse 5, wie in der Zeichnung dargestellt.	LREAL	mm
Offset 4	Versatz in Y-Richtung zwischen Motorachse 4 und TCP.	LREAL	mm
Offset 5	Versatz in X-Richtung zwischen Motorachse 5 und TCP.	LREAL	mm
Offset 6	Versatz in Z-Richtung zwischen Motorachse 4 und Motorachse 5.	LREAL	mm

Parameter	Beschreibung	Typ	Einheit
Tool offset OID	Objekt-ID eines am Kinematikflansch befestigten Werkzeugs. Das Flanschkoordinatensystem ist um 180° um die X-Achse gedreht, so dass die Z-Achse des Flanschkoordinatensystems nach unten zeigt.	OTCID	



Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

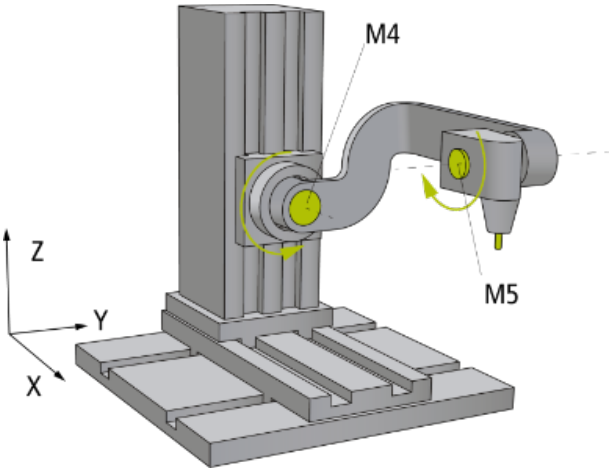
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5113 TC3 Kinematic Transformation (Level 4)

6.19 5D-Kinematics Type 3 (XYZAB)



Die 5D-Kinematics Type 3 ist, wie oben im Schema gezeigt, aufgebaut.

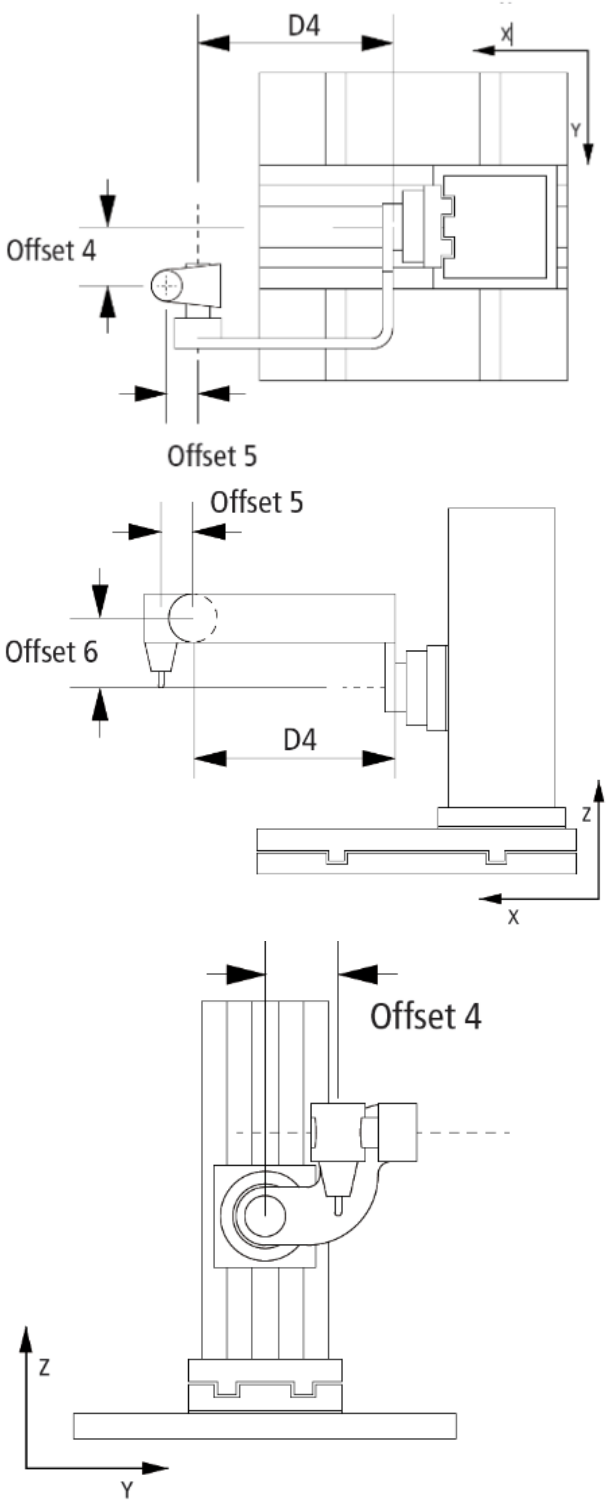
Die Motoren 1 bis 3 (X, Y, Z) sind in Millimetern skaliert. Die Motoren 4 und 5 sind in Grad skaliert. Die 0° Position ist die in der Zeichnung dargestellte Achsenposition; die Pfeile zeigen die positive Drehrichtung an.

Unterschied von Typ 3

Die 5D-Kinematics Typ 3 unterscheiden sich von den 5D-Kinematics Typ 2 in der Orientierung der positiven Richtung der Achsrotation um die Motorachsen M4 und M5.

Parameter für die Kinematik

Parameter	Beschreibung	Typ	Einheit
Handle D4	Armlänge in X-Richtung von Motorachse 4 zur Motorachse 5, wie in der Zeichnung dargestellt.	LREAL	mm
Offset 4	Versatz in Y-Richtung zwischen Motorachse 4 und TCP.	LREAL	mm
Offset 5	Versatz in X-Richtung zwischen Motorachse 5 und TCP.	LREAL	mm
Offset 6	Versatz in Z-Richtung zwischen Motorachse 4 und Motorachse 5.	LREAL	mm
Tool offset OID	Objekt-ID eines am Kinematikflansch befestigten Werkzeugs. Das Flanschkoordinatensystem ist um 180° um die X-Achse gedreht, so dass die Z-Achse des Flanschkoordinatensystems nach unten zeigt.	OTCID	



Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

Für alle Kinematiken mit Tool gilt zudem:

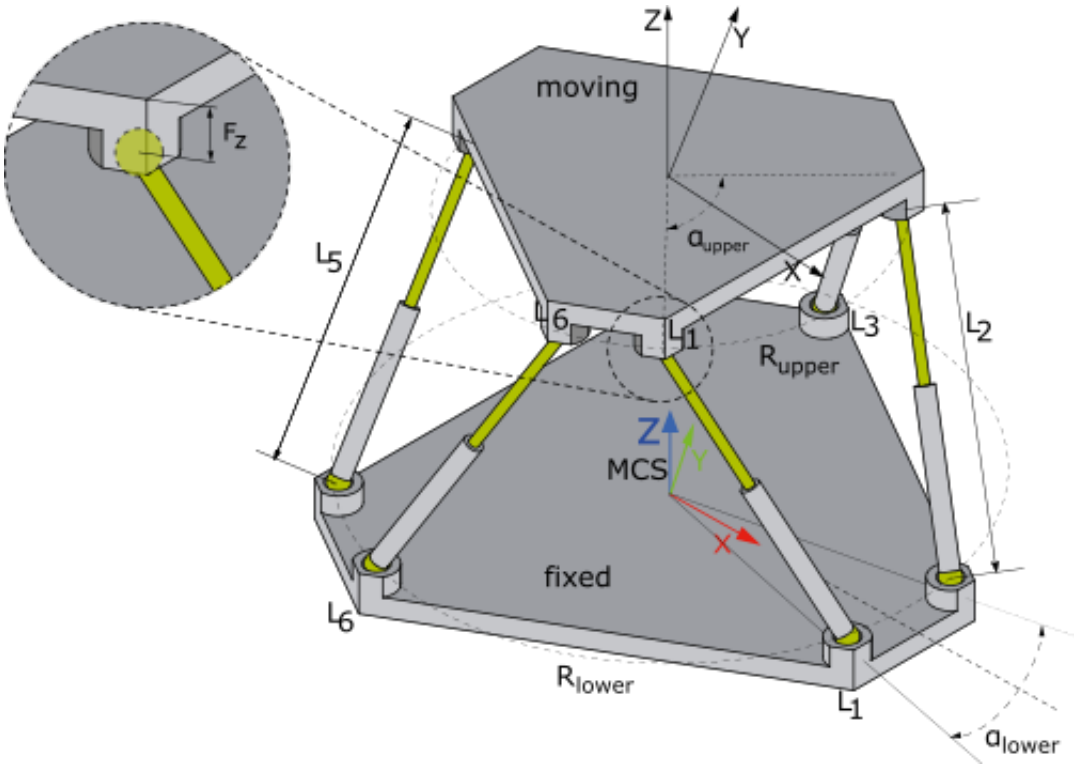
- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4018.26 TF5400 TC3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5113 TC3 Kinematic Transformation (Level 4)

6.20 6D-Stewart Platform (P_6L)

Für die kinematische Transformation 6D-Stewart Platform (P_6L) wird eine sich bewegende Plattform von sechs Zylindern getragen. Die Stewart Plattform ist eine parallele Kinematik mit sechs Freiheitsgraden.



Die X-Achse des Maschinenkoordinatensystems (MCS) zeigt in Richtung des Punktes, der sich in der Mitte zwischen Gelenk 1 und 2 befindet. Der Nullpunkt in Z-Richtung liegt meist etwas über der unteren (Basis-)Plattform und in der Ebene, die durch die Gelenkmittelpunkte der Ankerpunkte aufgespannt wird.

Alle Gelenke auf einer Plattform befinden sich auf einer Kreisbahn und haben so den gleichen Abstand zum Plattformmittelpunkt. Dieser Abstand ist mit R_{lower} (untere Plattform) und R_{upper} (obere Plattform) anzugeben.

Der Winkel zwischen den Gelenken 1 und 2, welcher gleichermaßen auch zwischen den Gelenken 3 und 4 sowie zwischen den Gelenken 5 und 6 ist, ist mit α_{lower} (untere Plattform) und α_{upper} (obere Plattform) anzugeben.

Die ACS-Achspositionen beziehen sich immer auf die gesamte Zylinderlänge L. Ein Aufstarten mit einer ACS-Achsposition gleich 0 ist daher nicht möglich.

Parameter für die Kinematik

Für die Stewart Kinematik gibt es die folgenden Parameter:

Parameter	Beschreibung	Typ	Einheit
Flange translation Z	Verschiebt die obere Ebene auf die Oberfläche der oberen Plattform, sodass die Stärke der Plattform berücksichtigt wird.	LREAL	mm
Tool offset OID	Die Werkzeug-Ausdehnung setzen.	OTCID	

Parameter	Beschreibung	Typ	Einheit
Lower radius R_{lower}	Radius der unteren Plattform. Beschreibt den Abstand vom Ursprung im Zentrum der unteren Plattform zu den Armgelenken der unteren Plattform.	LREAL	mm
Upper radius R_{upper}	Radius der oberen Plattform. Beschreibt den Abstand vom Ursprung im Zentrum der oberen Plattform zu den Armgelenken der oberen Plattform.	LREAL	mm
Lower angle α_{lower}	Der Winkel α_{lower} beschreibt den halben Winkel zwischen den Ankerpunkten von L_1 und L_2, L_3 und L_4, L_5 und L_6 auf der unteren Plattform.	LREAL	°
Upper angle α_{upper}	Der Winkel α_{upper} beschreibt den halben Winkel zwischen den Ankerpunkten von L_1 und L_2, L_3 und L_4, L_5 und L_6 auf der oberen Plattform.	LREAL	°

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

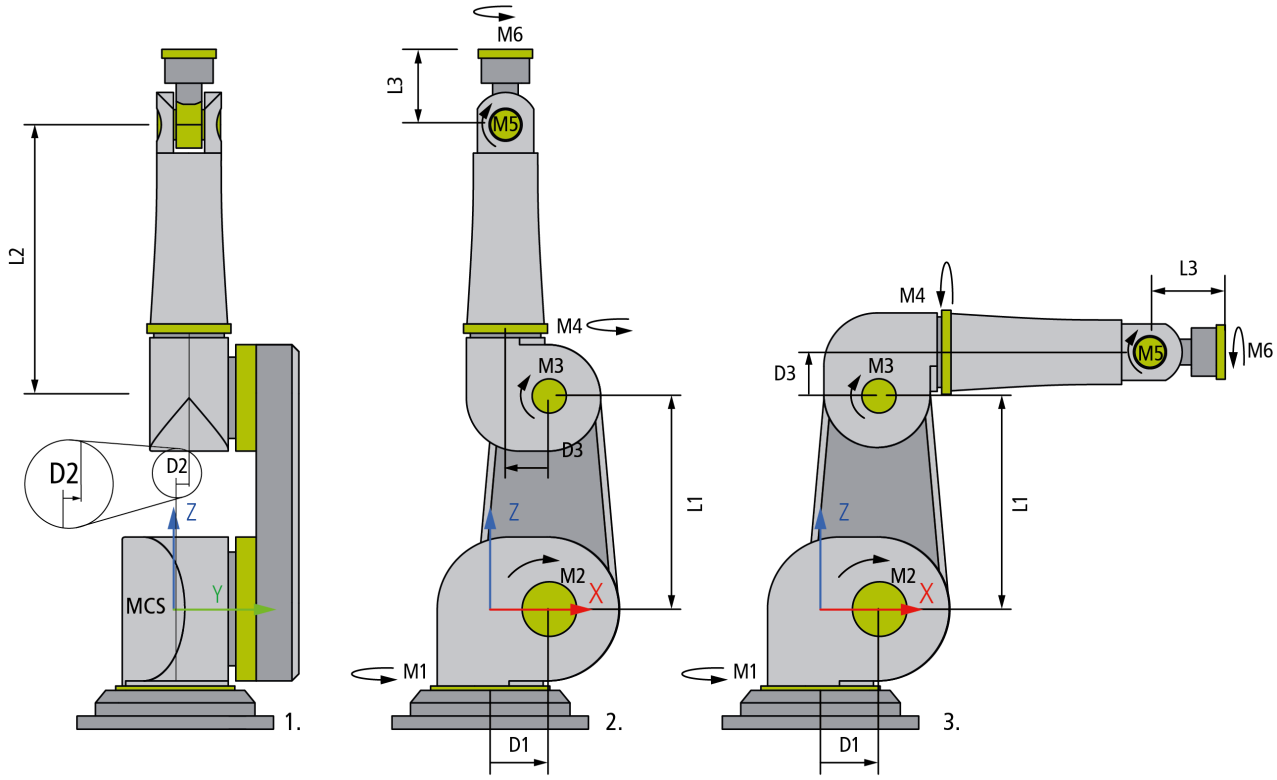
Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

Voraussetzungen

Entwicklungsumgebung Installationspackage	Zielplattform	TwinCAT Funktion
TwinCAT V3.1.4024.11 TF5400 TC3 Advanced Motion Pack V3.1.6.67	PC or CX (x86 or x64)	TF5113 TC3 Kinematic Transformation (Level 4)

6.21 Six Axis Articulated (S_CBBCBC)



Auf die Motorachsen der Six Axis Articulated (S_CBBCBC) Kinematics wird in der Einheit Grad Bezug genommen. Die Zeichnungen 1. und 2. oben zeigen die Kinematik mit allen Achsen in der Nullposition. Die Nullpositionen der Achsen M4 und M6 sind so definiert, dass das Maschinen-Koordinatensystem und das Flansch-Koordinatensystem die gleiche Orientierung haben. Die Zeichnung 3. zeigt die Achse M3 in 90° Position.

Der MCS Ursprung liegt im Schnittpunkt des ersten kinematischen Gelenks M1 mit dem zweiten kinematischen Gelenk M2. Er ist so orientiert, dass das Gelenk M2 eine Rotation um die Y-Achse beschreibt. Die Mitte von M1 beschreibt die X-Nullkoordinate. Der Schnittpunkt von M1 und M2 beschreibt die Y-Nullkoordinate. Die Mitte von M2 beschreibt die Z-Nullkoordinate.

Singuläre Positionen

Die in den Bildern 1., 2. und 3. dargestellten Positionen können nicht im kartesischen Modus angefahren werden, weil sich der Roboter jeweils in einer singulären Position befindet. Ein Anfahren dieser Positionen ist nur im Achs-Modus (Direct Mode) möglich.

Parameter für die Kinematik

Für die “Six Axis Articulated” Kinematics, eine serielle Sechs Achsen Kinematik, gibt es die folgenden Gelenk-Parameter.

Parameter	Beschreibung	Typ	Einheit
Arm length L1	Abstand zwischen den Motorachsen M2 und M3.	LREAL	mm
Arm length L2	Abstand zwischen den Motorachsen M3 und M5.	LREAL	mm
Arm length L3	Abstand zwischen der Motorachse M5 und dem Flansch.	LREAL	mm
Arm offset D1	Abstand zwischen den Motorachsen M1 und M2 in X-Richtung.	LREAL	mm
Arm offset D2	Abstand in Y-Richtung zwischen den Motorachsen M1 and M4.	LREAL	mm

Parameter	Beschreibung	Typ	Einheit
Arm offset D3	Abstand in X-Richtung zwischen den Motorachsen M3 und M5. Das Vorzeichen im Beispielbild ist positiv.	LREAL	mm

Allgemeine Parameter für die Kinematik

Allgemeine Parameter, die für jede Kinematik gelten, sind in den Abschnitten beschrieben:

- MCS Offset [▶ 19],
- Spatial reference definition [▶ 19].

Für alle Kinematiken mit Tool gilt zudem:

- Tool Offset OID [▶ 21].

6.22 Antriebsdrehmoment (Drive Torque)

Das Antriebsdrehmoment stellt die Trägheit und die Effizienz von Motor und Getriebe dar. Dieses wird für die genaue Berechnung des dynamischen Modells verwendet.

Durch einen Parameter in der Kinematik kann ein Objekt Drive Torque einer Kinematik zugewiesen werden.

Parameter für den Antrieb

Parameter	Beschreibung	Typ	Einheit
Drive moment of inertia	Trägheitsmoment des Motorrotors	LREAL	kg mm^2

Parameter für Getriebe

Parameter	Beschreibung	Typ	Einheit
Ratio	Getriebeübersetzung	LREAL	
Gearbox moment of inertia	Trägheitsmoment des Getriebes im Bezug auf den Antrieb	LREAL	kg mm^2
Coulomb friction	Stellt den Bewegungsreibungskoeffizienten dar	LREAL	Nm
Stokes friction	Stellt den Anteil der proportional zur Geschwindigkeit ansteigenden Reibung dar	LREAL	Nms

Erforderliches Produktlevel:
Level 1

6.23 Werkzeugversatz (Tool Offset)

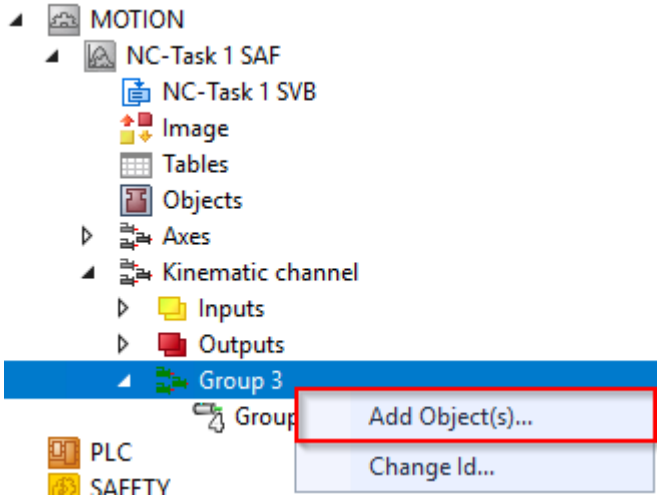
Der Werkzeugversatz bietet dem Benutzer die Möglichkeit, ein Werkzeug an den Flansch der Kinematik zu verbinden. Wenn in der Kinematik nicht anders angegeben, ist das Flanschkoordinatensystem so definiert, dass wenn alle Achsen auf 0 stehen, die Orientierung des Flanschkoordinatensystems der des Maschinenkoordinatensystems MCS entspricht.

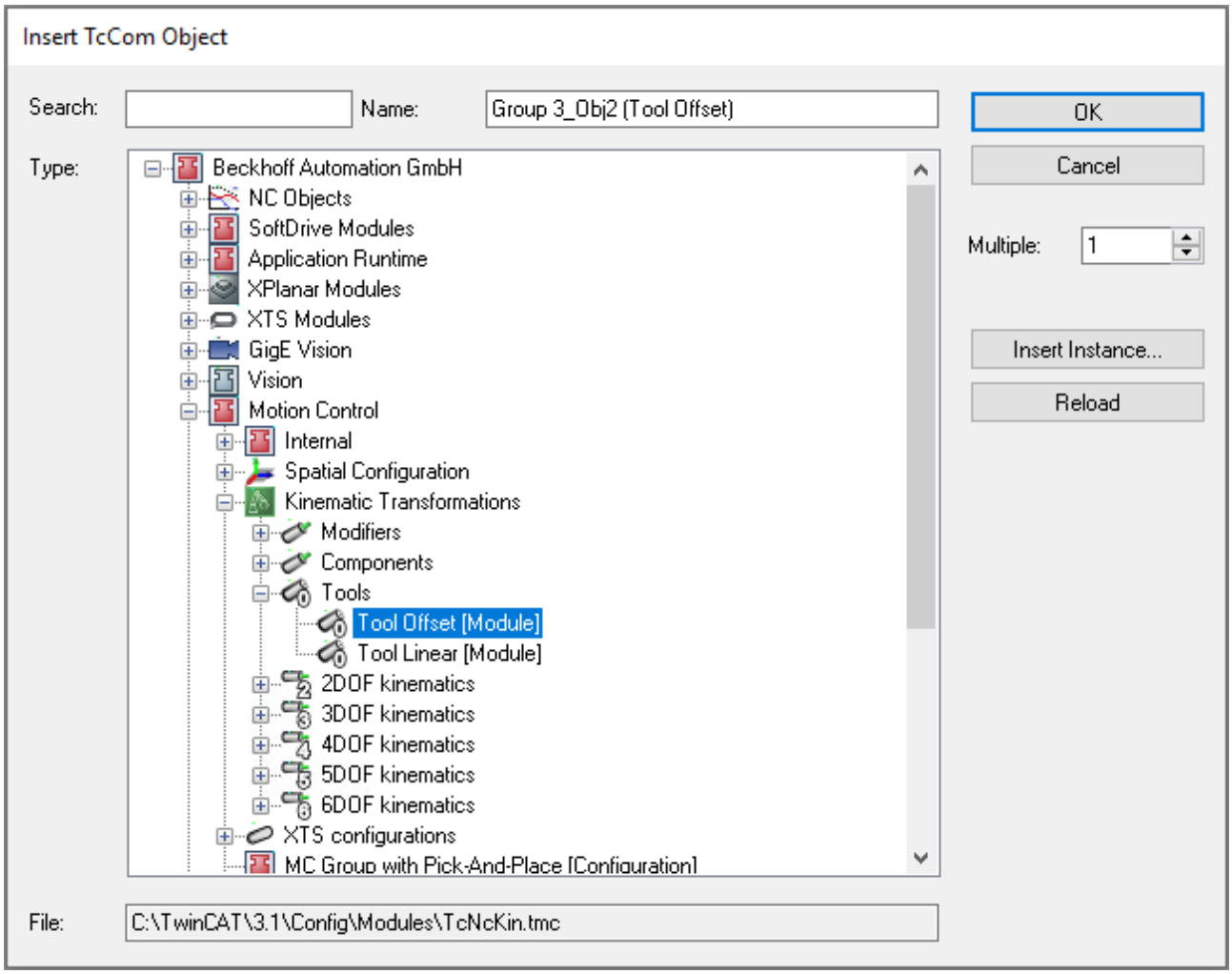
Parameter	Beschreibung	Typ	Einheit
Extension X	X-Versatz des statischen Werkzeugs, das am Koordinatensystem des Flansches der übergeordneten Transformation befestigt ist	LREAL	mm
Extension Y	Y-Versatz des statischen Werkzeugs, das am Koordinatensystem des Flansches der übergeordneten Transformation befestigt ist	LREAL	mm

Parameter	Beschreibung	Typ	Einheit
Extension Z	Z-Versatz des statischen Werkzeugs, das am Koordinatensystem des Flansches der übergeordneten Transformation befestigt ist	LREAL	mm

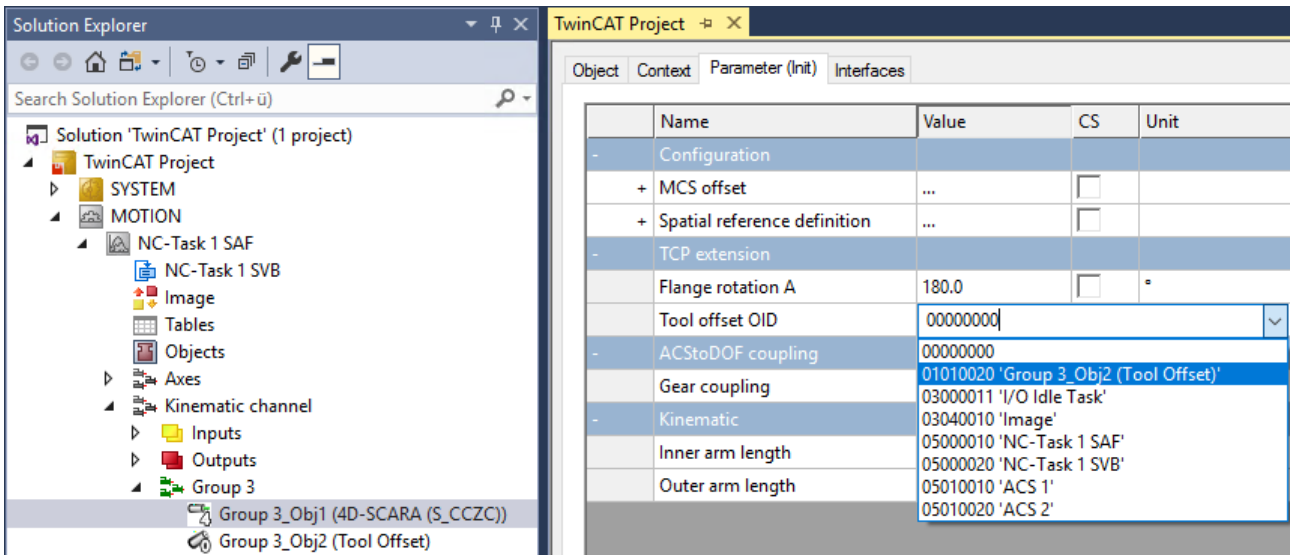
Anlegen eines Werkzeugs

1. Zunächst muss unter der Gruppe der Kinematik das Werkzeug angelegt werden.





2. Das angelegte Tool-Objekt kann über seine Tool OID in den Parametern der Kinematik zugewiesen werden.



3. Das Werkzeug kann nun über seine Objektparameter konfiguriert werden.

6.24 Werkzeug Linear (Tool Linear)

Das Tool Linear beschreibt ein 1D-Werkzeug, das am Flansch der Kinematik befestigt ist. Mittels Verwendung einer zusätzlichen Simulationsachse besteht die Möglichkeit der Bewegung in Werkzeugrichtung. Das 1D-Werkzeug kann dazu verwendet werden, den TCP in einen bestimmten Abstand von einem Werkstück zu bewegen.

Wenn in der Kinematik nicht anders angegeben, ist das Flanschkoordinatensystem so definiert, dass wenn alle Achsen auf 0 stehen, die Orientierung des Flanschkoordinatensystems der des Maschinenkoordinatensystems entspricht.

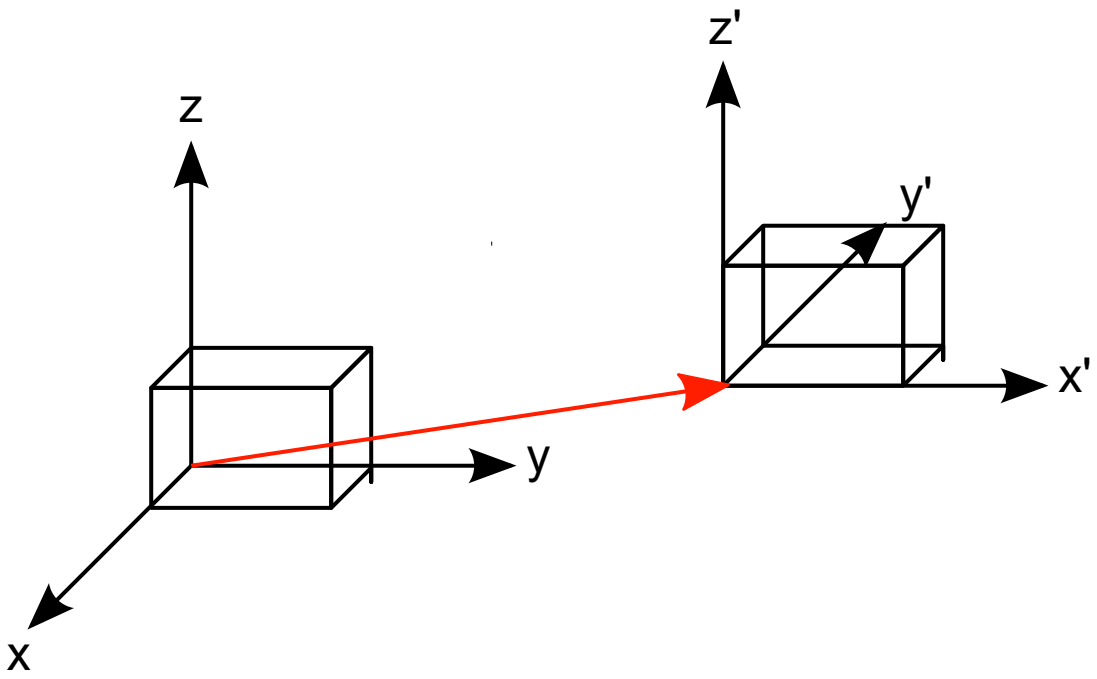
Wenn die Achsposition der zusätzlichen Simulationsache gleich 0 ist, dann befindet sich der TCP an der Position des Werkzeugversatzes (Parameter L_init).

Parameter	Beschreibung	Typ	Einheit
Length offset	Länge des Werkzeugs	LREAL	mm
Length Axis ID	Achs-ID der Simulationsachse - beim Bewegen dieser Achse bewegt sich der TCP in Richtung des linearen Werkzeugs.	UDINT	

Zum Anlegen eines Werkzeugs siehe Werkzeugversatz (Tool Offset) [▶ 51].

6.25 Koordinatensystem (Coordinate Frame)

Das Koordinatensystem unterstützt eine Translation und eine Rotation. Durch Verwendung dieser Transformation besteht die Möglichkeit, ein Benutzerkoordinatensystem (UCS „User Coordinate System“) zu definieren. Eine allgemeine Einführung über Koordinatensysteme befindet sich hier [▶ 8].

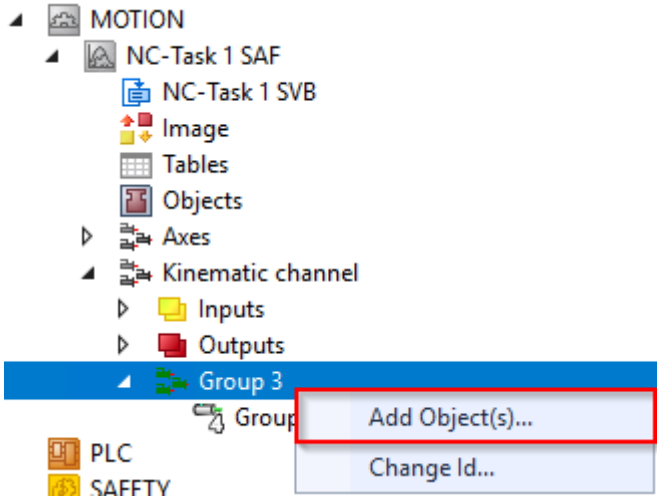


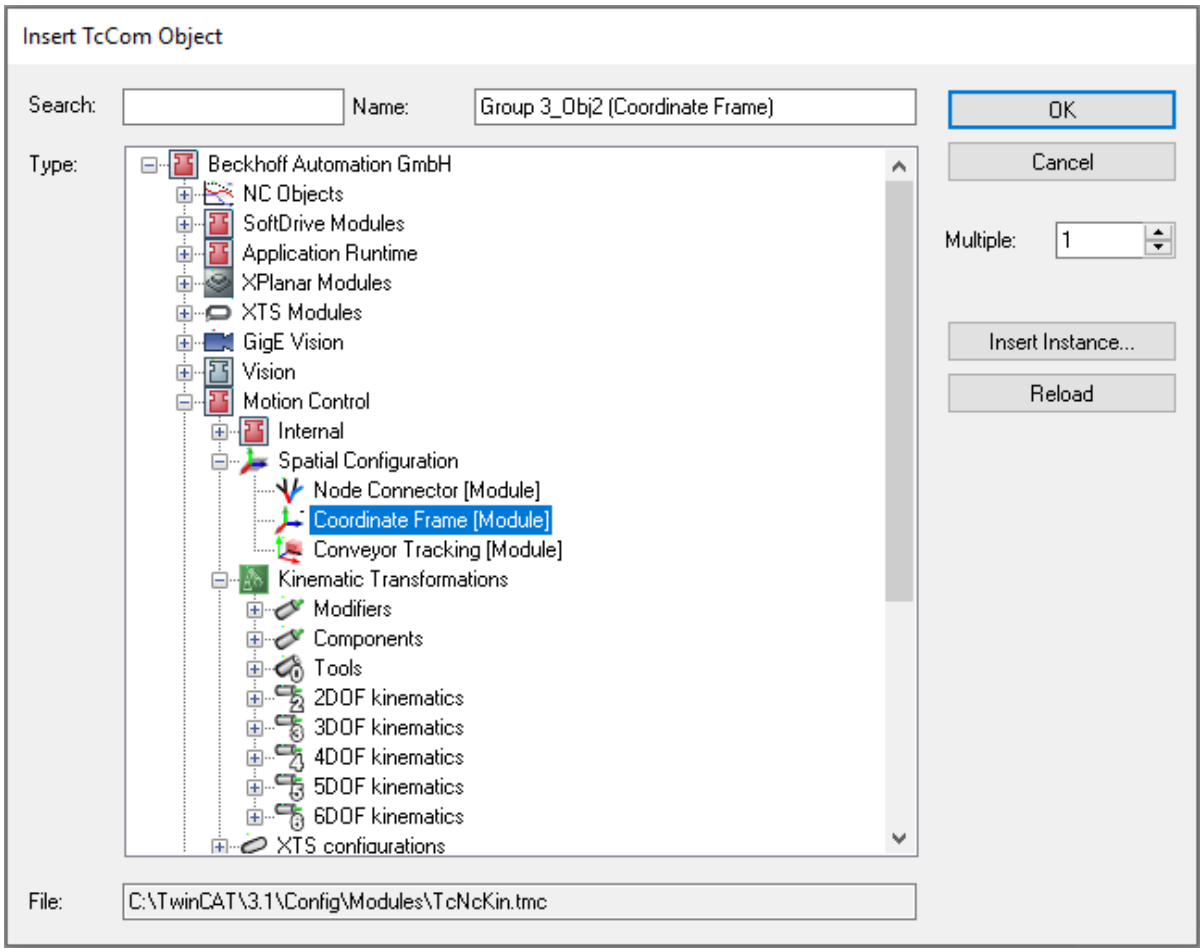
Zuerst wird die Translation berechnet, anschließend die Rotation. Die Reihenfolge der Rotationen beeinflusst die Orientierung des Koordinatensystems. Als Default für die Rotationsreihenfolge wird die in DIN 9300 beschriebene Roll-Pitch-Gier-Regel verwendet. Die Berechnungssequenz für die Vorwärtstransformation ist Z, Y', X".

Parameter	Beschreibung	Einheit
Translation X	Verschiebung in der X-Richtung	mm
Translation Y	Verschiebung in der Y-Richtung	mm
Translation Z	Verschiebung in der Z-Richtung	mm
Rotation 1	Winkel, um den als erstes rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	°
Rotation 2	Winkel, um den als zweites rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	°
Rotation 3	Winkel, um den als drittes rotiert wird. Die Interpretation wird vom Parameter Rotation Convention definiert.	°
Rotation convention	Die Rotationskonvention gibt an, in welcher Reihenfolge um die Achsen rotiert werden soll (Parameter Rotation 1-3). Dabei geben die Buchstaben (X, Y, Z) von links nach rechts die Reihenfolge an, in der um die entsprechenden Achsen rotiert wird. Die nachfolgende Zahl gibt an auf welchen Parameter (Rotation 1-3) der Wert zu parametrieren ist. Die translatorische Verschiebung wird immer vor der Rotation ausgeführt.	
Spatial reference	Der Parameter Spatial reference, gibt an, auf welches Koordinatensystem als Basis sich dieses Koordinatensystem bezieht. Ist hier der Wert 0 eingestellt wird das WCS als Basis verwendet. Soll ein anderes Koordinatensystem als Ausgangspunkt für die Verschiebung genutzt werden, so kann ein weiteres Objekt Koordinatensystem angelegt werden. Die Objekt-ID dieses Koordinatensystems kann über das Dropdown-Menü ausgewählt werden	
Definition direction	Gibt die Richtung an, in der die Verschiebung programmiert wird (aus Sicht des Bezugssystems oder aus Sicht dieses Koordinatensystems).	

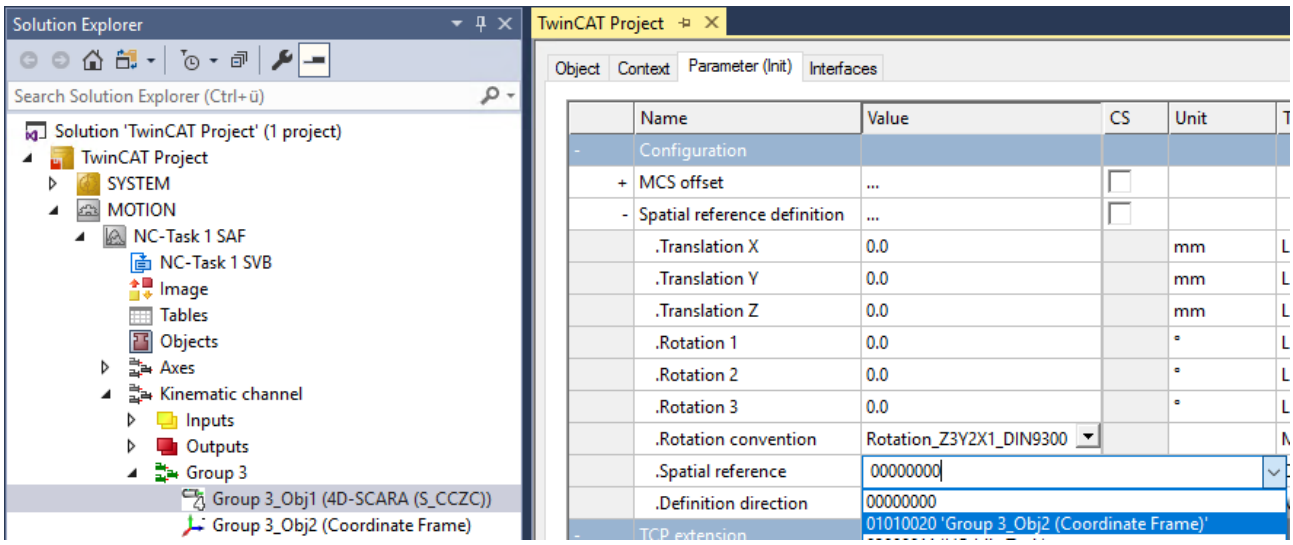
Anlegen eines Koordinatensystems

1. Zunächst muss unter der Kinematikgruppe das Koordinatensystem angelegt werden.





2. Das angelegte Koordinatensystem-Objekt kann in der Kinematik über den Parameter Spatial reference als Ursprung des MCS der Kinematik definiert werden.



3. Das Koordinatensystem kann nun über seine Objektparameter konfiguriert werden.

7 Benutzerspezifische Transformationen - Wie es geht …

⚠ WARNUNG

Online-Change vermeiden

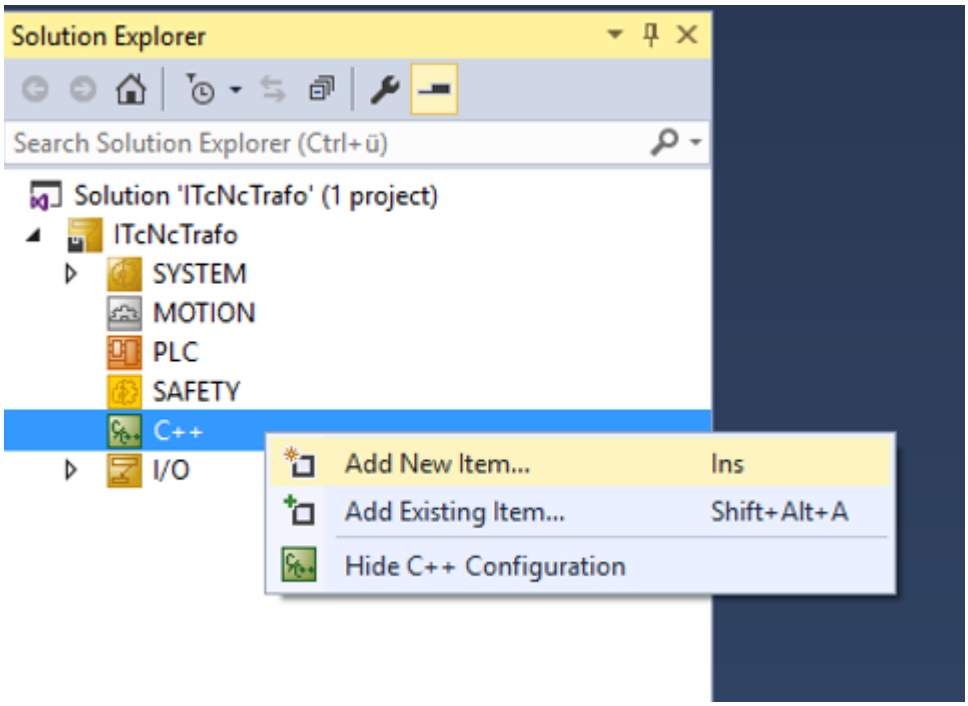
Bei einem Online-Change in Verbindung mit benutzerspezifischen Kinematiken kann es auf der C++ Seite zu Unstetigkeiten kommen. Deshalb sollte ein Online-Change in der Regel vermieden werden.
Beachten Sie das Kapitel „Schnellstart mit Online-Change“.

ITcNcTrafo

Schritt-für-Schritt Anleitung zum Integrieren eigener Kinematik mit TF511x

✓ Starten Sie mit einem leeren TwinCAT Projekt.

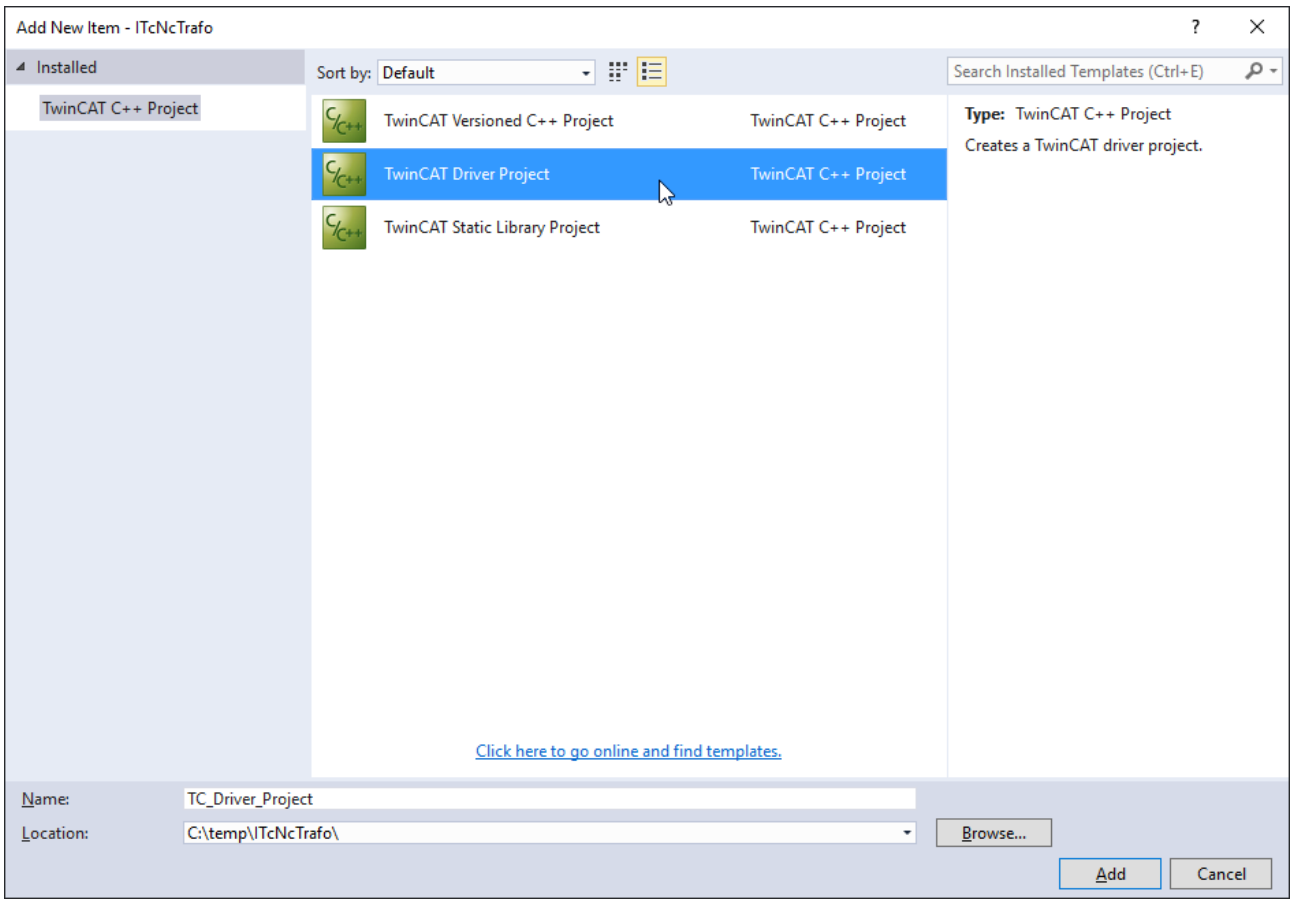
1. Fügen Sie ein C++ Objekt hinzu.



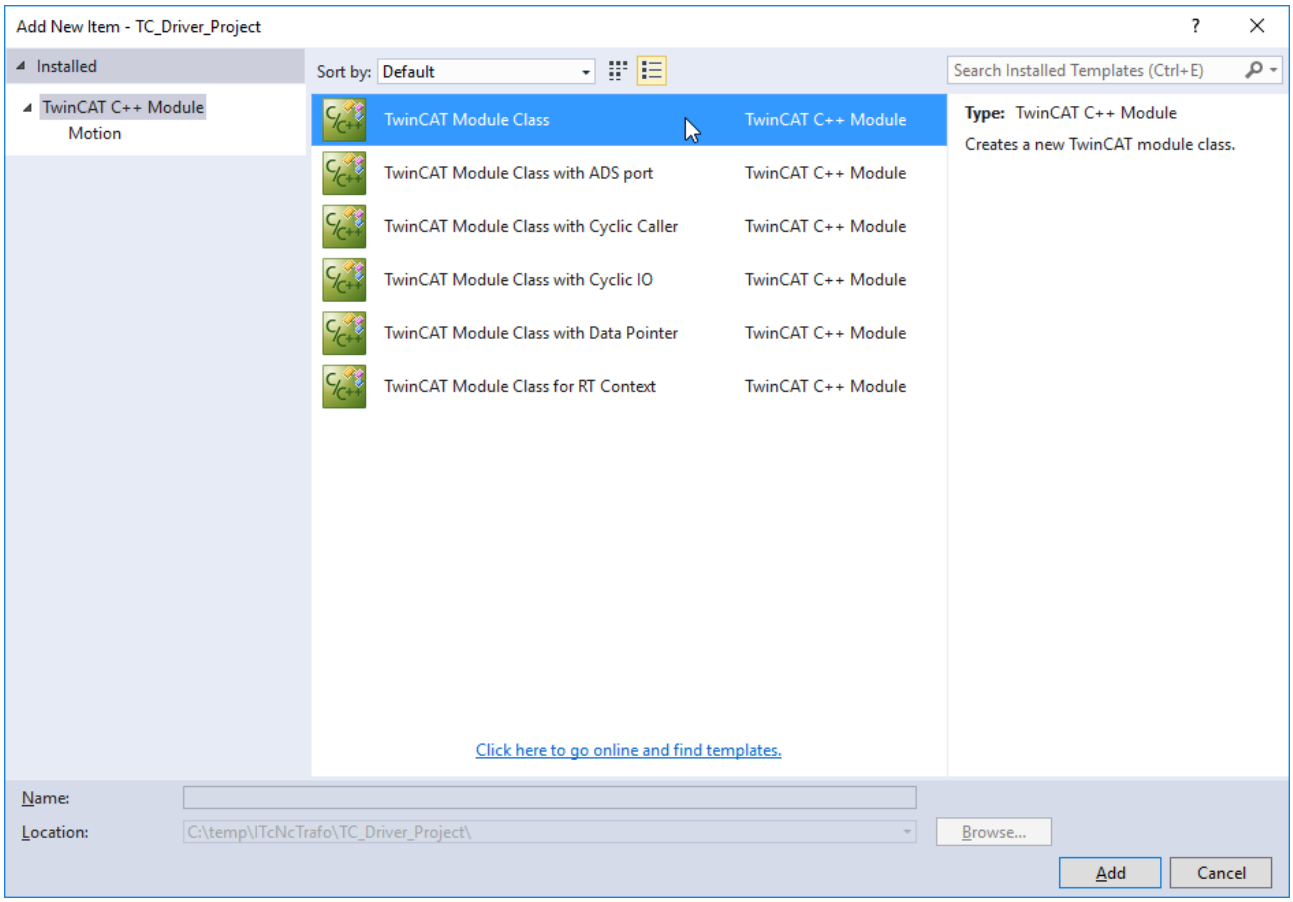
2. Wählen Sie ein TwinCAT Driver Projekt oder ein TwinCAT Versioned C++ Projekt.

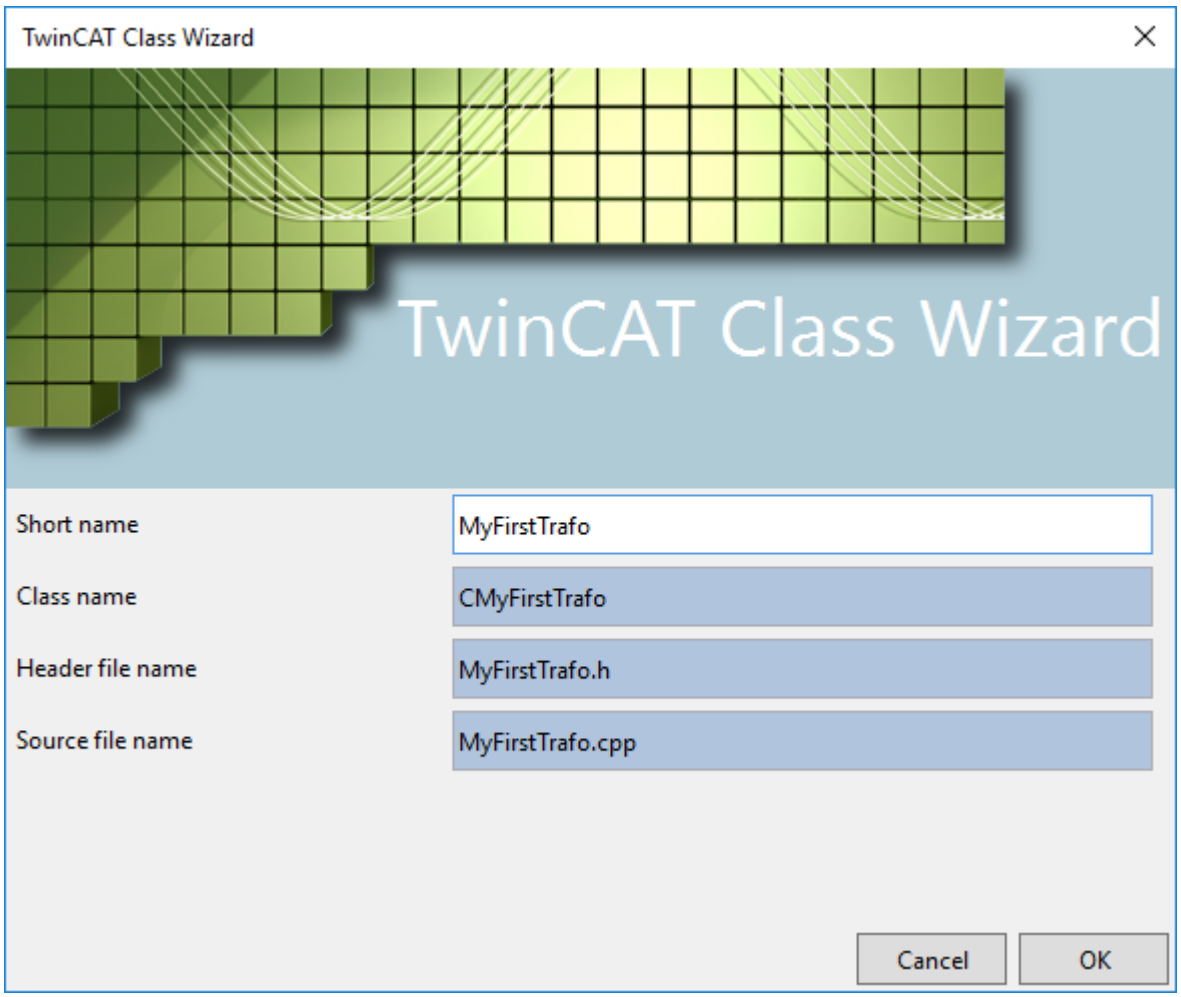
Versioned C++ Project

Versioned C++ Project kann ab TC3.1.4024.10 für Benutzerspezifische Kinematiken genutzt werden.

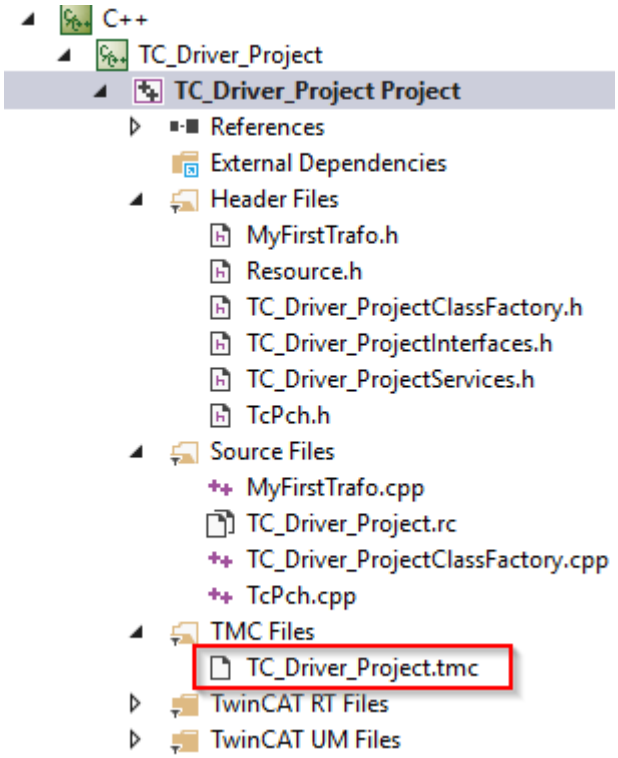


3. Wählen Sie eine TwinCAT Modul-Klasse.

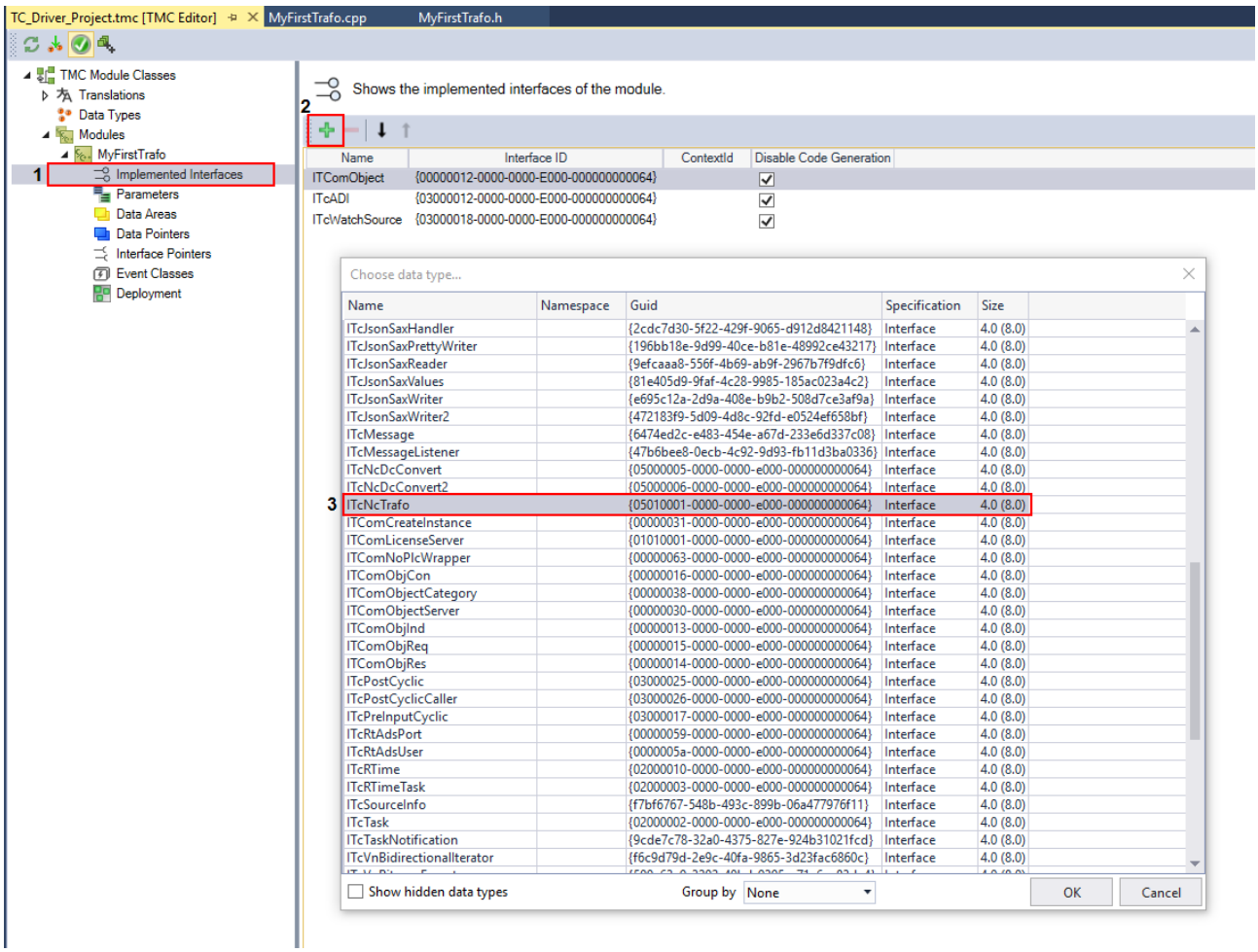




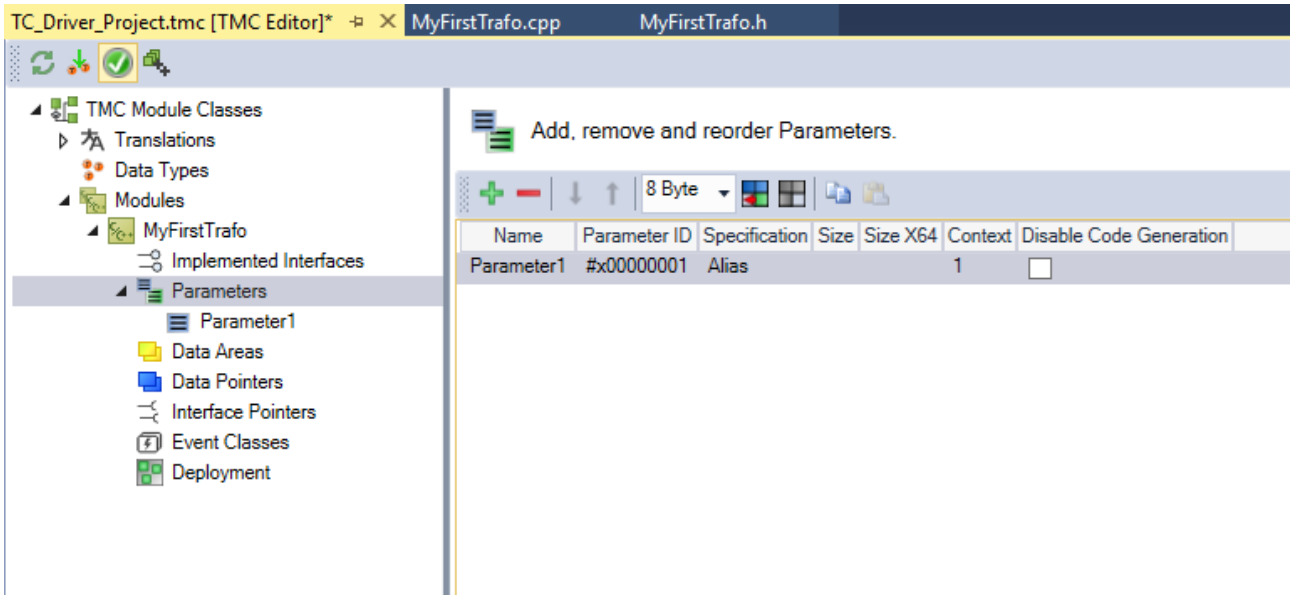
4. Wählen Sie den `TMC` Editor an.

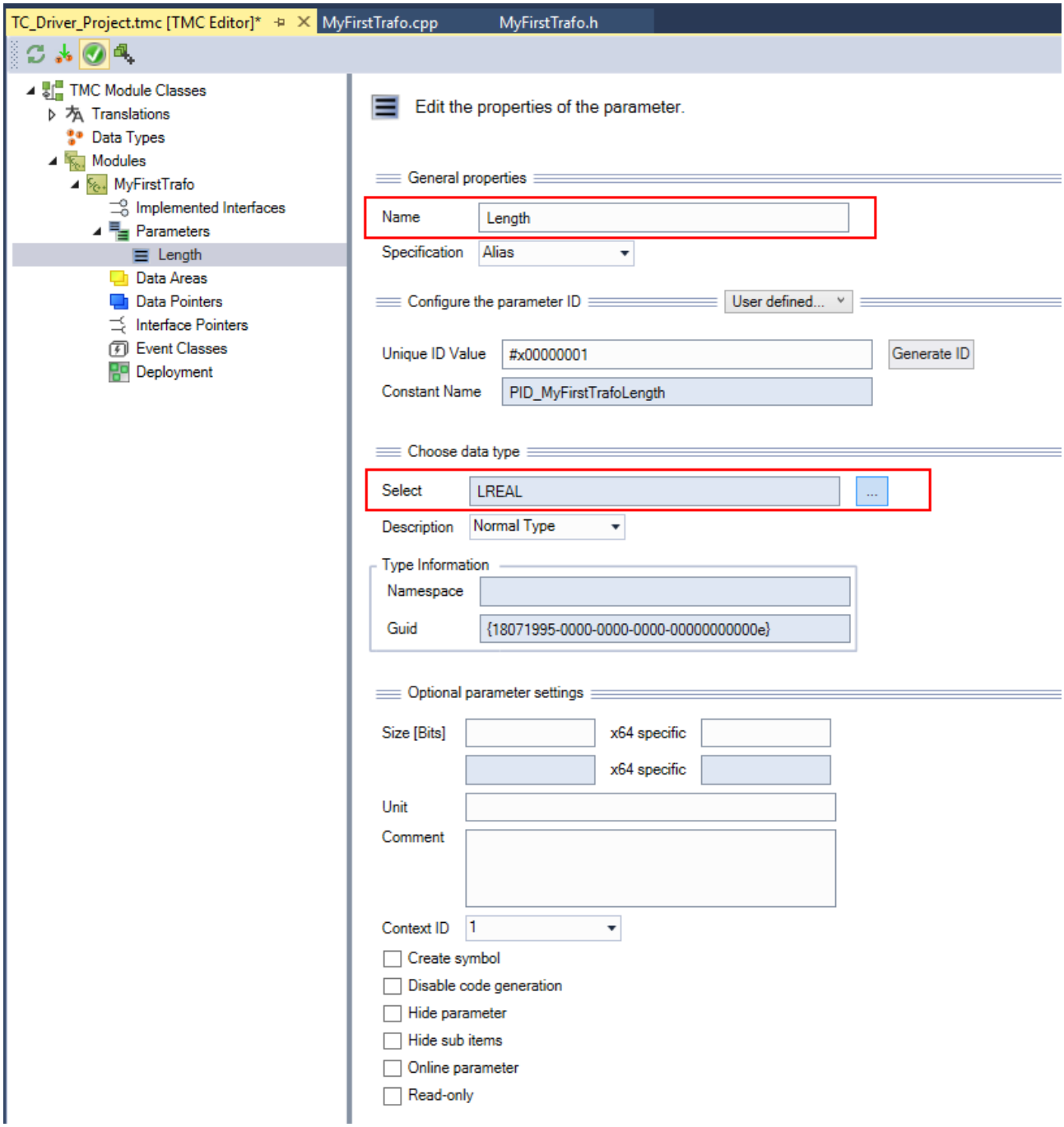


5. Fügen Sie das `ITcNcTrafo` Interface hinzu.

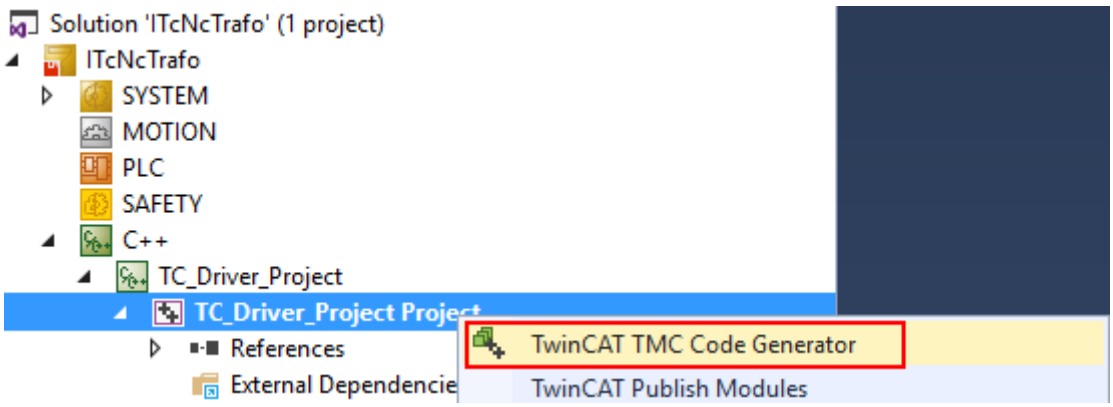


6. Fügen Sie kinematikspezifische Parameter hinzu (Armlänge, Verschiebungen, usw.).

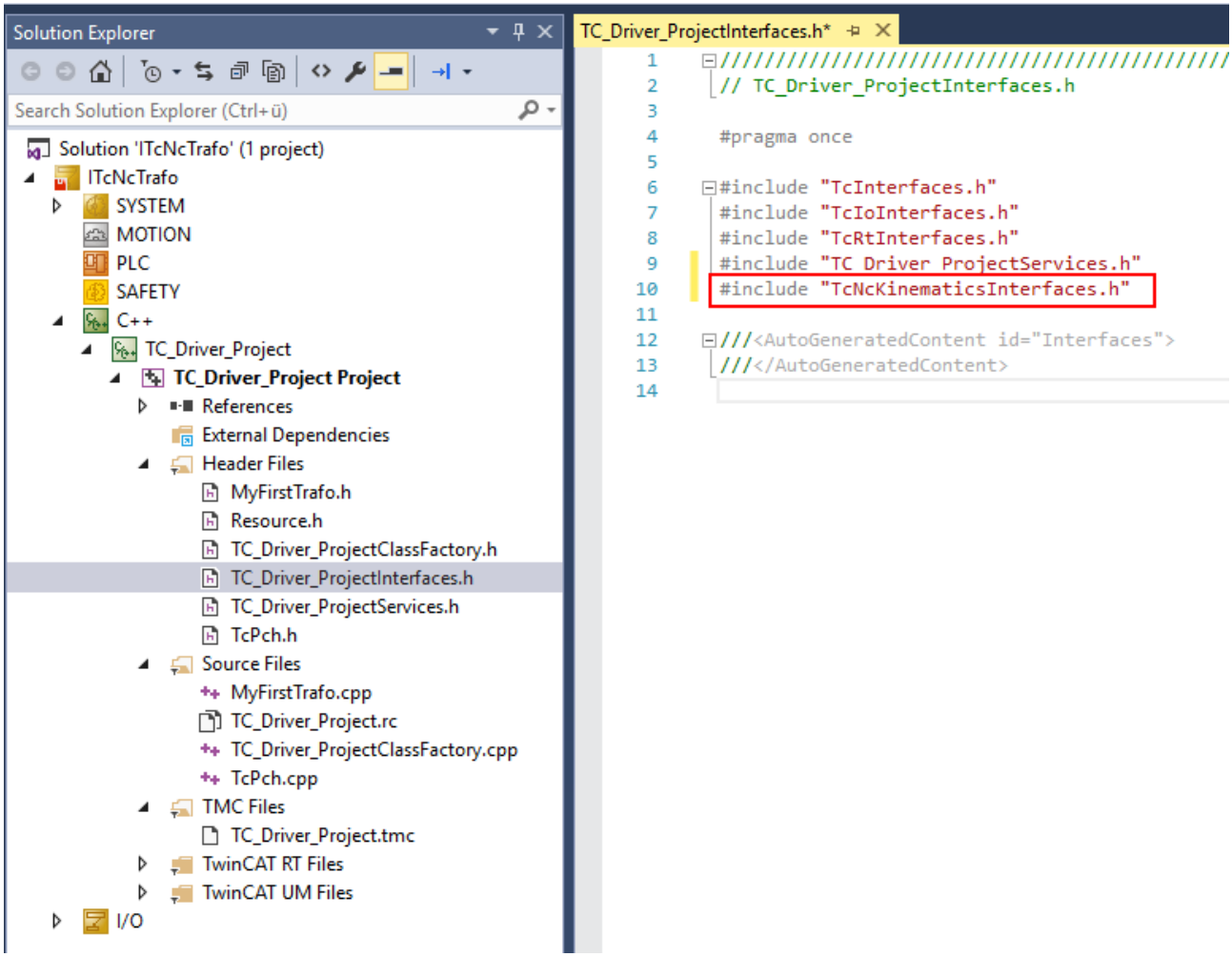




7. Führen Sie den `TMC` Code Generator aus.



8. Fügen Sie einen Header hinzu. `<ProjectName>Interfaces.h` benötigt zusätzlich eine Include-Anweisung. Fügen Sie auch `TcNcKinematicsInterfaces.h` hinzu.

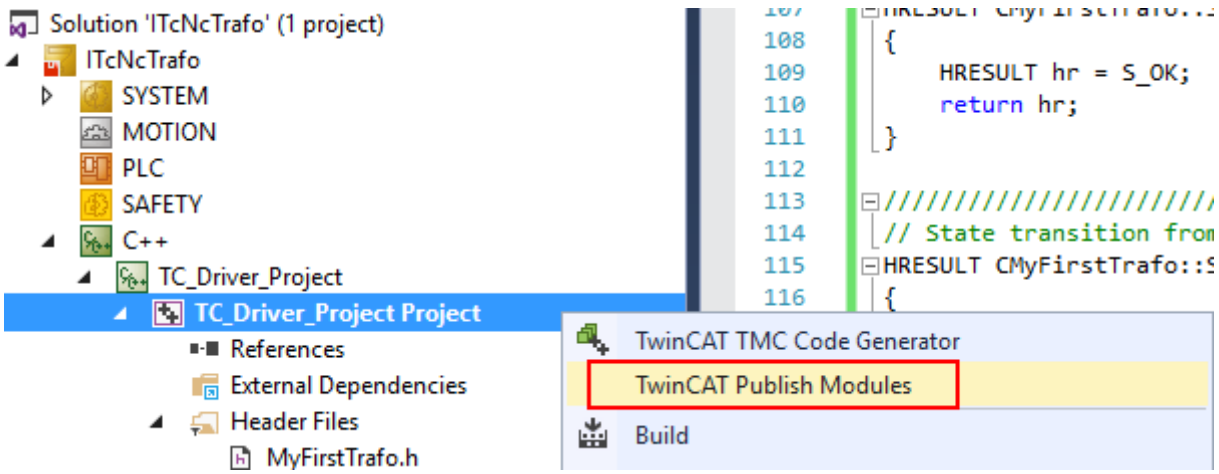


9. Implementieren Sie die Methoden `Forward` (Vorwaerts), `Backward` (Rueckwaerts), `TrafoSupported` (UnterstuetzteTransformationen), `GetDimensions` (HoleDimensionen). Diese Methoden müssen eine valide Implementierung haben und werden automatisch an der Stelle `Source Files\<TrafoName>.cpp` abgelegt.

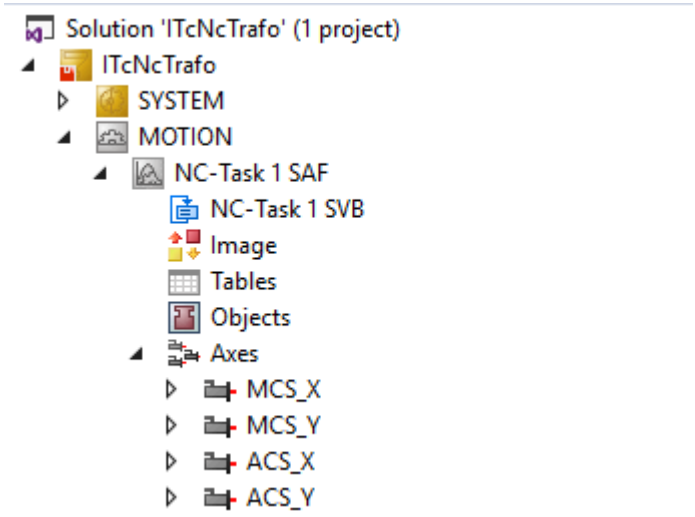
E_NOTIMPL

Der Default-Rückgabewert `E_NOTIMPL` der Methoden `Forward`, `Backward`, `TrafoSupported` und `GetDimensions` führt bei der Nutzung der Kinematik zu Fehlermeldungen und muss unbedingt überschrieben werden.

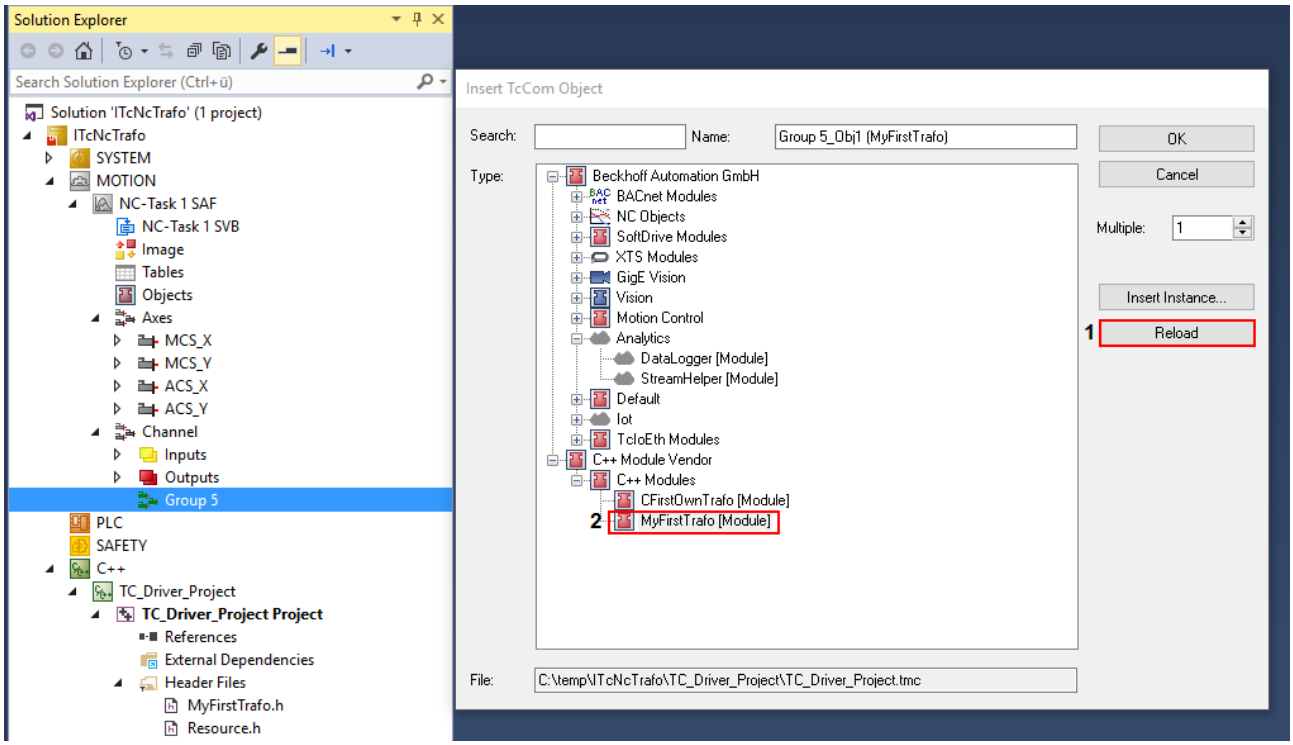
10. Erstellen Sie und veröffentlichen Sie die Module.



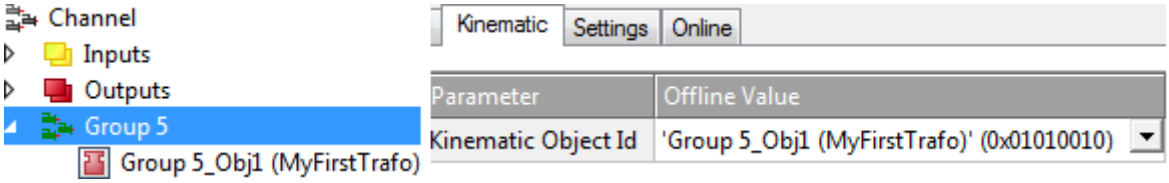
11. Konfigurieren Sie Achsen im `MOTION` Teilbaum und fügen Sie PTP Achsen hinzu.



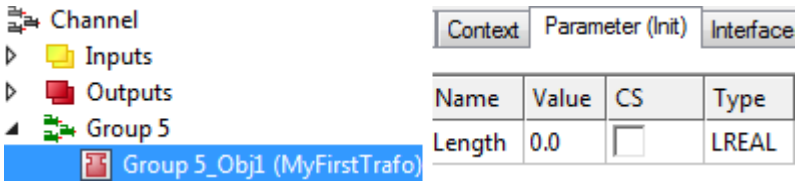
12. Fügen Sie benutzerspezifische Module (Channel) hinzu.
13. Laden Sie die `TcCom` Objekte neu, indem Sie die Schaltfläche `Reload` betätigen.
14. Wählen Sie Ihr Transformations-Objekt, und bestätigen Sie Ihre Auswahl, indem Sie die Schaltfläche `OK` betätigen.



15. Die Transformationsgruppe muss wissen, welches Root-Modul aufzurufen ist. Deshalb muss die Objekt-ID der Kinematik (in diesem Fall `MyFirstTrafo`) ausgewählt werden. Das Kinematik-Objekt definiert die Anzahl der in der SPS zu verwendenden ACS-Achsen und MCS-Achsen (siehe ST_KinAxes [▶ 81]).



16. Die Objektparameter entsprechend der verwendeten Kinematik parametrieren. Wenn das erledigt ist, dann ist die XAE Konfiguration abgeschlossen.



⇨ Die Transformation kann jetzt von der SPS aus aktiviert werden (siehe SPS-Bibliothek). Zum Ansprechen der Transformation ein zyklisches Kanalinterface in der SPS definieren und dieses mit den IO des Kinematikkanals verknüpfen.

```
in_stKinToPlc       AT %I*      : NCTOPLC_NCICHANNEL_REF;
out_stPlcToKin      AT %Q*      : PLCTONC_NCICHANNEL_REF;
```

Kundenspezifische Kinematik

Der Funktionsbaustein `FB_KinCalcTrafo` kann nicht für Transformationen verwendet werden, die selbst erstellt worden sind.

8 SPS-Bibliothek

Funktionsbaustein	Beschreibung
Kinematic Tranformation	
FB_KinConfigGroup [▶ 66]	Konfiguriert ACS- und MCS-Achsen entsprechend der kinematischen Transformationsgruppe und aktiviert den kartesischen Modus oder Gelenkmodus(ACS).
FB_KinResetGroup [▶ 68]	Setzt die kinematische Transformationsgruppe zurück.
F_KinGetChnOperationState [▶ 79]	Liest zyklisch den Status der kinematischen Transformationsgruppe.
F_KinGetAcsMcsAxisIds [▶ 80]	Liest die aktiven ACS- und MCS-Achsen der Kinematikgruppe.
Transformationsberechnung	
FB_KinCalcTrafo [▶ 69]	Berechnet die kinematische Transformation ohne Verbindung zu den Achsen.
FB_KinCalcMultiTrafo [▶ 71]	Berechnet die kinematische Transformation für mehrere Haltungen.
Parameter und Koordinatensysteme online bearbeiten	
FB_KinLockTrafoParam [▶ 75]	Sperrt die Parameter der kinematischen Transformationsgruppe, verweigert Schreibzugriff.
FB_KinUnlockTrafoParam [▶ 73]	Entsperrt die Parameter der kinematischen Transformationsgruppe, erlaubt Schreibzugriff.
Erweiterter Drehbereich	
FB_KinExtendedRotationRange [▶ 77]	Speichert und stellt den Rotationszustand der kinematischen Gruppe wieder her.
FB_KinPresetRotation [▶ 78]	Setzt den Rotationszustand.

Strukturen und Aufzählungen

Name	Beschreibung
ST_KinAxes [▶ 81]	Struktur der ACS- und MCS-Achsen, die die Kinematik bilden
E_KinStatus [▶ 81]	Status der Kinematikgruppe (Enum)

Entwicklungsumgebung	Zielsystem	Einzubindende SPS-Bibliotheken
TwinCAT 3	PC oder CX (x86, x64)	Tc2_NcKinematicTransformation

Funktionsbausteine zur Kompatibilität mit bestehenden Programmen

Funktionsbausteine zur Kompatibilität

Die unten aufgeführten Funktionsbausteine existieren zur Kompatibilität mit bestehenden Projekten. Für neue Projekte wird empfohlen, diese Bausteine nicht zu verwenden und stattdessen die äquivalenten Bausteine in der oberen Tabelle zu benutzen.

Funktionsbaustein	Beschreibung
FB_KinCheckActualStatus [▶ 83]	Liest azyklisch den Status der kinematischen Transformationsgruppe

8.1 Funktionsbausteine

8.1.1 FB_KinConfigGroup

Der Funktionsbaustein FB_KinConfigGroup konfiguriert Achsen entsprechend der kinematischen Transformation. Dies sind Achsen für das ACS (Gelenk) und das MCS (kartesisch). Der Funktionsbaustein nimmt die in der **stAxesList** definierten ACS- und MCS-Achsen und konfiguriert diese in der Kinematikgruppe von **stKinRefIn**.

VAR_INPUT

```
VAR_INPUT
    bExecute            : BOOL;
    bCartesianMode      : BOOL;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

bCartesianMode: Wenn FALSE, dann können die ACS-Achsen (Gelenk) direkt bewegt werden. Wenn TRUE, dann wird die in den MCS-Achsen (kartesisch) beschriebene Bewegung in eine Bewegung der ACS-Achsen (Gelenk) transformiert. Die ACS-Achsen können nicht direkt bewegt werden.

VAR_IN_OUT

```
VAR_IN_OUT
    stAxesList          : ST_KinAxes;
    stKinRefIn          : NCTOPLC_NCICHANNEL_REF;
END_VAR
```

stAxesList: Bestimmt die ACS- und MCS-Achsen, die in der Konfiguration enthalten sind. Siehe ST_KinAxes.

stKinRefIn: Bestimmt die Kinematikgruppe der Konfiguration.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy               : BOOL;
    bDone               : BOOL;
    bError              : BOOL;
    nErrorId            : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

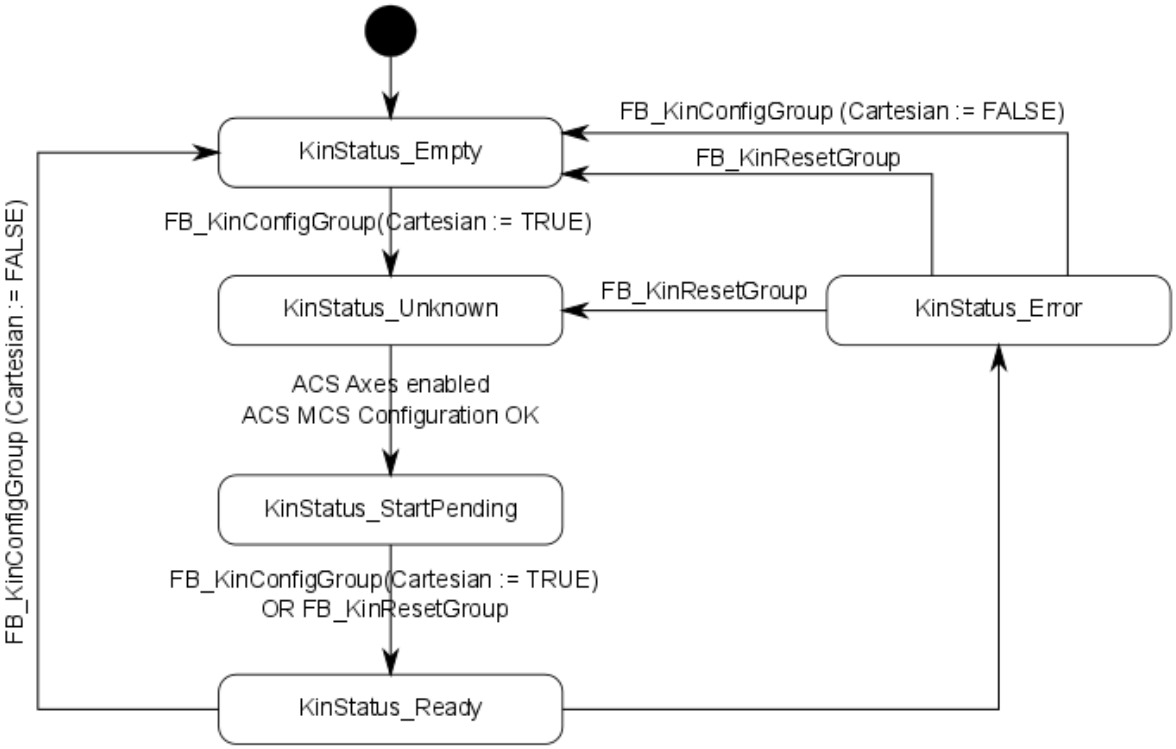
```
VAR
    io_X                  : AXIS_REF;
    io_Y                  : AXIS_REF;
    io_Z                  : AXIS_REF;
    io_M1                 : AXIS_REF;
    io_M2                 : AXIS_REF;
    io_M3                 : AXIS_REF;
    in_stKinToPlc AT %I*  : NCTOPLC_NCICHANNEL_REF;
    fbConfigKinGroup      : FB_KinConfigGroup;
    stAxesConfig          : ST_KinAxes;
    bAllAxesReady         : BOOL;
    bExecuteConfigKinGroup: BOOL;
    bUserConfigKinGroup   : BOOL;
    bUserCartesianMode    : BOOL := TRUE;
    (*true: cartesian mode - false: direct mode (without transformation) *)
END_VAR
```

```
(* read the IDs from the cyclic axis interface so the axes can mapped later to the kinematic group
*)
stAxesConfig.nAxisIdsAcs[1] := io_M1.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[2] := io_M2.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[3] := io_M3.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[1] := io_X.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[2] := io_Y.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[3] := io_Z.NcToPlc.AxisId;

IF bAllAxesReady AND bUserConfigKinGroup THEN
    bExecuteConfigKinGroup := TRUE;
ELSE
    bExecuteConfigKinGroup := FALSE;
END_IF

fbConfigKinGroup(
    bExecute       := bExecuteConfigKinGroup ,
    bCartesianMode := bUserCartesianMode ,
    stAxesList     := stAxesConfig,
    stKinRefIn     := in_stKinToPlc );
```

Zustand der Kinematikgruppe



Konfiguration freigeben

Die ACS-Achsen müssen durch MC_Power freigegeben sein, damit der Zustand den Wert **KinStatus_Ready** erreichen kann. Wenn die ACS-Achsen nicht freigegeben sind, geben Sie die Achsen frei und rufen dann FB_KinConfigGroup oder FB_KinResetGroup auf.

8.1.2 FB_KinResetGroup

Der Funktionsbaustein FB_KinResetGroup setzt die Kinematikgruppe zurück. Alle ACS- und MCS-Achsen werden zurückgesetzt. Des Weiteren kann der Eingang *nItpChannelId* für die Festlegung des zugehörigen Interpolationskanals verwendet werden. Der Kanal wird zurückgesetzt, wenn die nItpChannelId ungleich 0 ist.

Wenn alle Achsen freigegeben sind und die Gruppe sich im kartesischen Modus befand, dann kehrt die Gruppe zurück zum Zustand KinStatus_Ready. Befand sich die Gruppe nicht im kartesischen Modus, dann kehrt die Gruppe zum Zustand KinStatus_Empty zurück. Wenn die Achsen nicht freigegeben sind, dann verbleibt die Gruppe im Zustand KinStatus_Empty.

VAR_INPUT

```
VAR_INPUT
    bExecute      : BOOL;
    nItpChannelId : UDINT;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

nItpChannelId: ID des zugehörigen Interpolationskanals. Wenn der Eingang ungleich 0 ist, dann wird der zugehörige Interpolationskanal zurückgesetzt.

VAR_IN_OUT

```
VAR_IN_OUT
    stAxesList            : ST_KinAxes;
    stKinRefIn            : NCTOPLC_NCICHANNEL_REF;
END_VAR
```

stAxesList: Bestimmt die ACS- und MCS-Achsen, die in der Konfiguration enthalten sind. Siehe ST_KinAxes.

stKinRefIn: Bestimmt die Kinematikgruppe der Konfiguration.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy                 : BOOL;
    bDone                 : BOOL;
    bError                : BOOL;
    nErrorId              : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

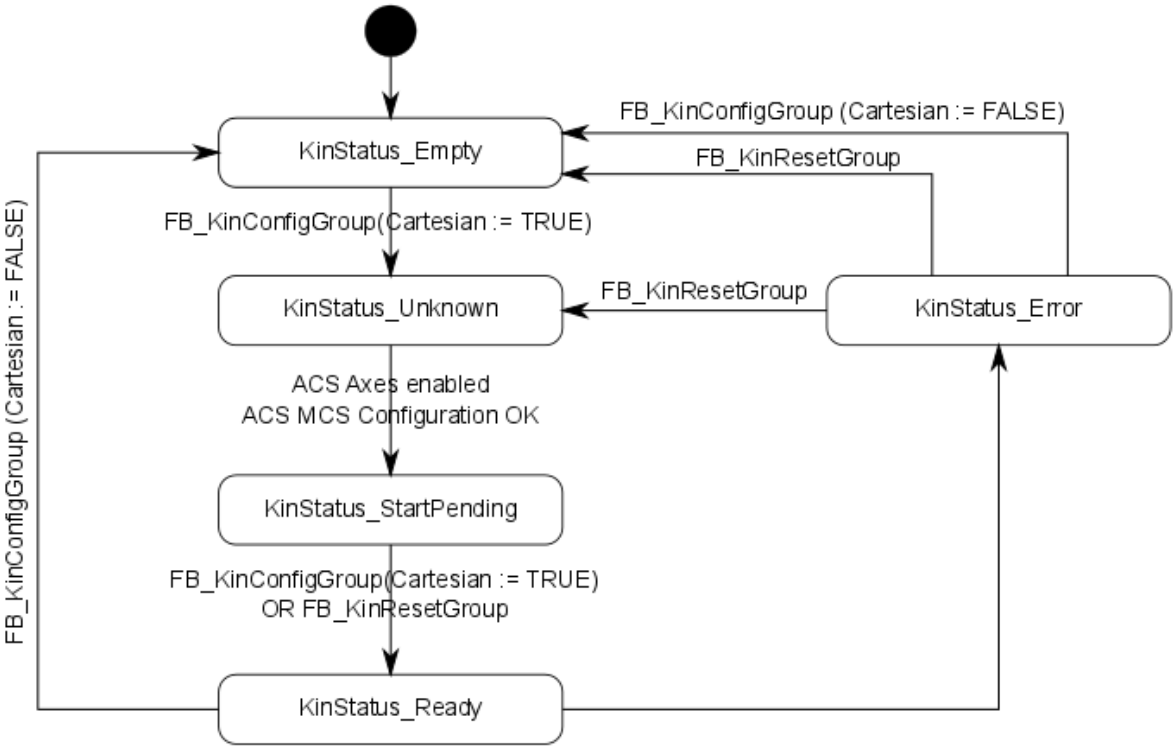
nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

```
VAR
    fbFB_ResetKinGroup   : FB_KinResetGroup;
    stAxesConfig         : stAxesConfig;
    in_stKinToPlc AT %I* : NCTOPLC_NCICHANNEL_REF;
END_VAR
```

```
fbFB_ResetKinGroup(
    bExecute := TRUE,
    nItpChannelId := 3,
    stKinRefIn := in_stKinToPlc,
    stAxesList := stAxesConfig,
    bBusy=> ,
    bDone=> ,
    bError=> ,
    nErrorId=> );
```

Zustand der Kinematikgruppe



8.1.3 FB_KinCalcTrafo

Der Funktionsbaustein FB_KinCalcTrafo berechnet die Vorwärts- oder Rückwärtstransformation, selbst dann, wenn keine Kinematikgruppe mit FB_KinConfigGroup [▶ 66] erstellt wurde.

VAR_INPUT

```
VAR_INPUT
    bExecute            : BOOL;
    bForward            : BOOL;
    oidTrafo            : UDINT;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

bForward: Bestimmt, ob die Vorwärts- oder Rückwärtstransformation berechnet wird.

oidTrafo: Objekt-ID des zu berechnenden kinematischen Transformationsobjekts. Siehe Beispiel [▶ 71] unten.

VAR_IN_OUT

```
VAR_IN_OUT
    stAxesPosIn         : ARRAY[1..8] OF LREAL;
    stAxesPosOut        : ARRAY[1..8] OF LREAL;
    uMetaInfoIn         : U_KinMetaInfo;
    uMetaInfoOut        : U_KinMetaInfo;
END_VAR
```

stAxesPosIn: Array, das die Eingangspositionen der Transformation enthält. Bei der Berechnung einer Vorwärtstransformation stellen diese die Gelenkpositionen dar. Bei der Berechnung einer Rückwärtstransformation stellen diese die kartesischen Achspositionen dar.

stAxesPosOut: Array, das die Ergebnispositionen der Transformation enthält. Im Falle der Berechnung einer Vorwärtstransformation stellen diese die kartesischen Achspositionen dar. Im Falle der Berechnung einer Rückwärtstransformation stellen diese die Gelenkpositionen dar.

uMetaInfoIn: Wenn verschiedene Roboterkonfigurationen zu einer Lösung führen, kann die Lösung ausgewählt werden, die verwendet werden soll (siehe Beispiel [▶ 71]). Für Kinematiken, bei denen dieser Parameter nicht benötigt wird, kann dieser Eingang mit einer Dummy-Variablen belegt werden.

uMetaInfoOut: Wenn verschiedene Lösungen für eine Transformation möglich sind, wird die gefundene Lösung spezifiziert. Für Kinematiken, bei denen dieser Parameter nicht benötigt wird, kann dieser Eingang mit einer Dummy-Variablen belegt werden.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy               : BOOL;
    bDone               : BOOL;
    bError              : BOOL;
    nErrorId            : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

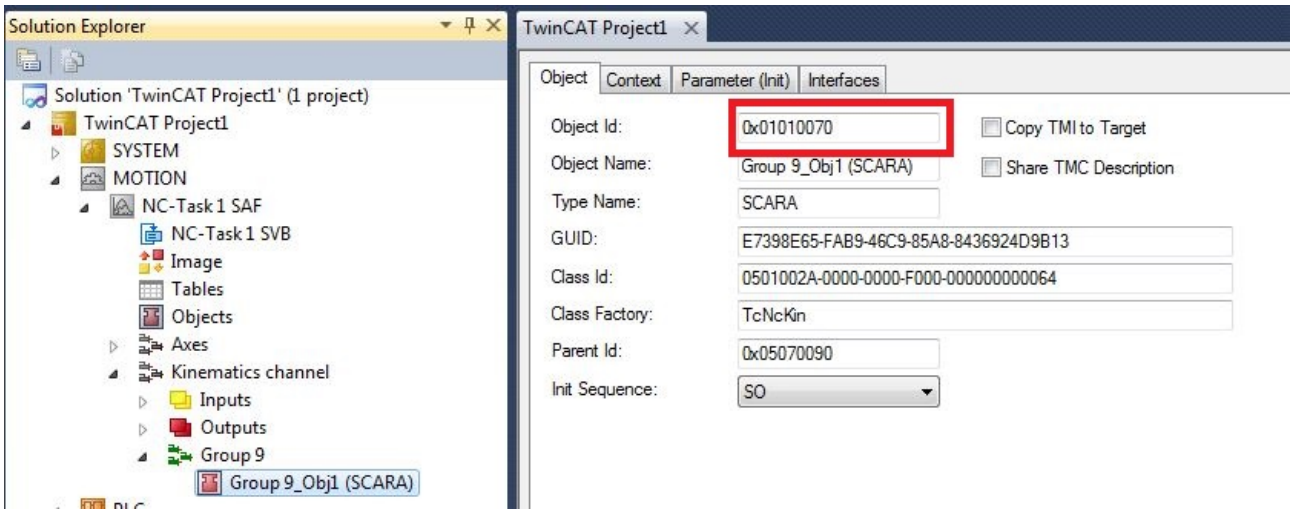
bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

Die Objekt-Id der Transformation kann aus dem Transformationsobjekt unter dem Kinematikkanal eingesehen werden.

SCARA-Transformation [▶ 41] - Beispiel-Objekt-ID




```
VAR
    fbKinCalcTrafo     : FB_KinCalcTrafo;
    stAxesPosIn        : ARRAY[1..8] OF LREAL;
    stAxesPosOut       : ARRAY[1..8] OF LREAL;
    bUserExecute       : BOOL;
    bUserCalcFwdTrafo  : BOOL;
    uScaraMetaInfoIn : U_KinMetaInfo;
    uScaraMetaInfoOut : U_KinMetaInfo;
END_VAR
```

```
uScaraMetaInfoIn.eScara := E_KinMetaInfoScara.scaraLeftArm;

fbKinCalcTrafo(
    bExecute := bUserExecute,
    bForward := bUserCalcFwdTrafo,
    oidTrafo := 16#01010070,
    stAxesPosIn := stAxesPosIn,
    stAxesPosOut := stAxesPosOut,
    uMetaInfoIn:= uScaraMetaInfoIn ,
    uMetaInfoOut:= uScaraMetaInfoOut,
    bBusy=> ,
    bDone=> ,
    bError=> ,
    nErrorId=> );
```

8.1.4 FB_KinCalcMultiTrafo

Der Funktionsbaustein FB_KinCalcMultiTrafo berechnet für mehrere Positionen die Vorwärts- oder Rückwärtstransformation, selbst dann, wenn keine Kinematikgruppe mit FB_KinConfigGroup [▶ 66] erstellt wurde.

Alternativ kann der Funktionsbaustein FB_KinCalcTrafo [▶ 69] verwendet werden, um die kinematischen Transformationen einzeln zu berechnen.

Eingänge

Name	Typ	Beschreibung
bExecute	BOOL	Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.
bForward	BOOL	Bestimmt, ob die Vorwärts- oder Rückwärtstransformation berechnet wird.
oidTrafo	UDINT	Objekt-ID des zu berechnenden kinematischen Transformationsobjekts.
pDataIn	POINTER TO BYTE	Zeiger auf die Eingangsdaten, bestehend aus einer Instanz von ST_KinMultiTrafoHeader [▶ 83] und einem Array mit Eingangspositionen. Bei der Berechnung einer Vorwärtstransformation stellen diese die Gelenkpositionen dar. Bei der Berechnung einer Rückwärtstransformation stellen diese die kartesischen Achspositionen dar.
nSizeIn	UDINT	Größe der Eingangsdaten auf die pDataIn zeigt
pDataOut	POINTER TO BYTE	Zeiger auf die Ausgangsdaten.
nSizeOut	UDINT	Größe der Ausgangsdaten auf die pDataOut zeigt.

Ausgänge

Name	Typ	Beschreibung
bBusy	BOOL	Der Ausgang wird TRUE, wenn der Befehl mit bExecute gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während bBusy gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn bBusy wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge bDone oder bError gesetzt.
bDone	BOOL	Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.
bError	BOOL	Der Ausgang bError wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.
nErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

ST_KinCalcMultiTrafoIn

```
TYPE ST_KinCalcMultiTrafoIn :
STRUCT
hdr  : ST_KinMultiTrafoHeader;
fPos : ARRAY[1..2] OF ARRAY[1..4] OF LREAL;
END_STRUCT
END_TYPE
```

ST_KinCalcMultiTrafoOut

```
TYPE ST_KinCalcMultiTrafoOut :
STRUCT
    fPos      : ARRAY[1..2] OF ARRAY[1..4] OF LREAL;
    fMetaInfo : ARRAY[1..2] OF U_KinMetaInfo;
END_STRUCT
END_TYPE
```

MAIN

```
PROGRAM MAIN
VAR
    {attribute 'TcInitSymbol'} oidKinematic: OTCID;
    nState: UDINT := 0;
```

```
    fbKinCalcMultiTrafo   : FB_KinCalcMultiTrafo;
    stKinCalcMultiIn      : ST_KinCalcMultiTrafoIn;
    stKinCalcMultiOut     : ST_KinCalcMultiTrafoOut;
END_VAR
```

```
CASE nState OF
0:
    // Header for Multi Trafo
    stKinCalcMultiIn.hdr.nColumnsIn := 4;
    stKinCalcMultiIn.hdr.nColumnsOut := 4;
    stKinCalcMultiIn.hdr.nLines := 2;
    stKinCalcMultiIn.hdr.uMetaInfo.eScara := E_KinMetaInfoScara.scaraLeftArm;
    stKinCalcMultiIn.hdr.bGetMetaInfo := TRUE;

    // Positions
    stKinCalcMultiIn[1][1]:=0;
    stKinCalcMultiIn[1][2]:=90;
    stKinCalcMultiIn[1][3]:=0;
    stKinCalcMultiIn[1][4]:=0;

    stKinCalcMultiIn[2][1]:=0;
    stKinCalcMultiIn[2][2]:=-90;
    stKinCalcMultiIn[2][3]:=0;
    stKinCalcMultiIn[2][4]:=0;

    nState := nState + 10;

10:
    fbKinCalcMultiTrafo( bExecute := TRUE,
                         bForward := TRUE,
                         oidTrafo := oidKinematic,
                         pDataIn  := ADR(stKinCalcMultiIn),
                         nSizeIn  := SIZEOF(stKinCalcMultiIn),
                         pDataOut := ADR(stKinCalcMultiOut),
                         nSizeOut := SIZEOF(stKinCalcMultiOut) );

    IF NOT fbKinCalcMultiTrafo.bBusy THEN
        fbKinCalcMultiTrafo(bExecute:= FALSE, bForward:= TRUE, oidTrafo:= oidKinematic,
                            pDataIn:=ADR(stKinCalcMultiIn), nSizeIn:= SIZEOF(stKinCalcMultiIn),
                            pDataOut:=ADR(stKinCalcMultiOut), nSizeOut:=
SIZEOF(stKinCalcMultiOut) );

        nState := nState + 10;
    END_IF
END_CASE
```

Systemvoraussetzungen

Entwicklungsumgebung	Zielsystem	Einzubindende SPS-Bibliotheken
Advanced Motion Pack V3.1.10.51	PC or CX (x64)	Tc2_NcKinematicTransformation

8.1.5 FB_KinUnlockTrafoParam

Der Funktionsbaustein FB_KinUnlockTrafoParam entsperrt Transformationsparameter, die einen Einfluss auf die Position haben, so dass diese geschrieben werden können.

Nach der Freigabe können die Kinematikparameter von der SPS mit ADSWRITE geschrieben werden. Die erforderliche Indexgruppe ist die Objekt-ID und der Indexoffset ist die Parameter-ID. Die geschriebenen Parameter sind nicht persistent. Parameter die keinen Einfluss auf die Position haben (z. B. Drehmomente und Massen), können ohne Aufruf von FB_KinUnlockTrafoParam geschrieben werden.

⚠ VORSICHT

Das Verändern der Parameter kann zu Unstetigkeiten führen.

Beachten Sie, dass dies mit größter Vorsicht zu benutzen ist. Die Neufestlegung von Kinematikparametern kann zu Positionssollwertsprüngen in der Kinematikkette führen.

Nachdem Kinematikparameter geschrieben wurden, kann das Beschreiben mit FB_LockTrafoParam wieder gegen Beschreiben gesperrt werden.

VAR_INPUT

```
VAR_INPUT
    bExecute            : BOOL;
    oidTrafo            : UDINT;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

oidTrafo: Objekt-ID des kinematischen Transformationsobjekts. Siehe Beispiel [▶ 74] unten.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy               : BOOL;
    bDone               : BOOL;
    bError              : BOOL;
    nErrorId            : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

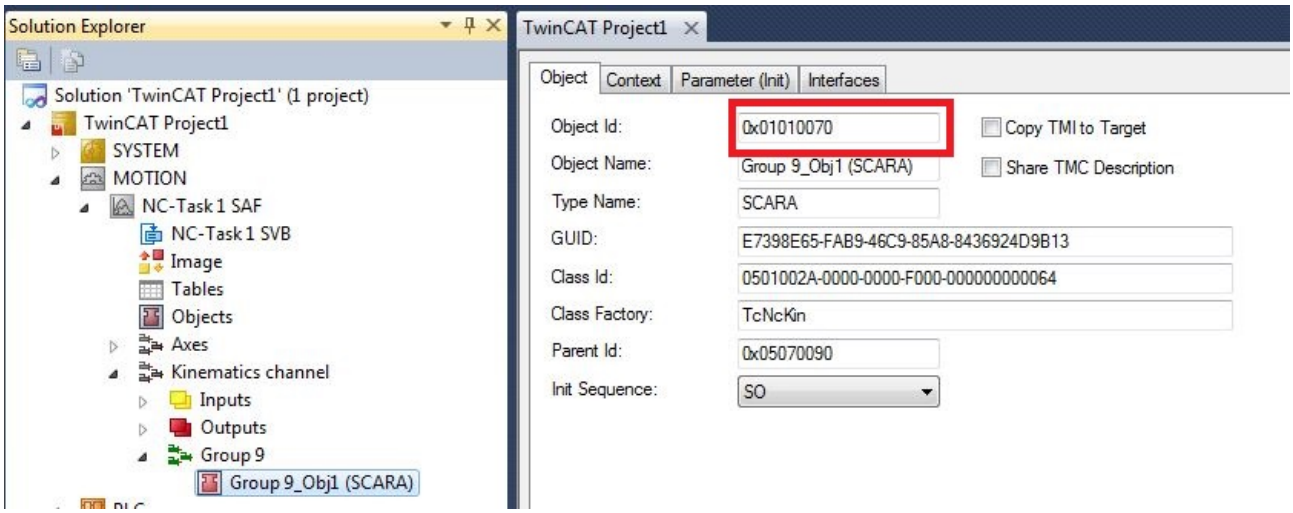
bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

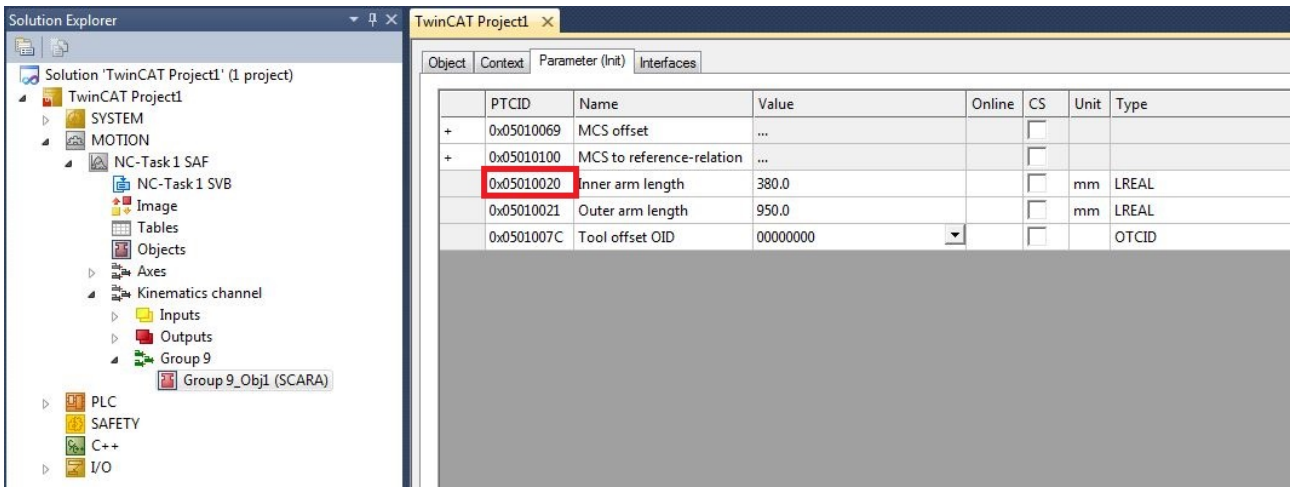
bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

Die für die Freigabe eines Transformationsparameters und für das Schreiben eines entsprechenden neuen Wertes benötigten Objekt-ID und Parameter-ID können aus dem Transformationsobjekt in der XAE gelesen werden.






```
VAR
    bUserExecuteUnlock     : BOOL;
    fbFB_UnlockTrafoParam  : FB_KinUnlockTrafoParam;
    bUserExecuteWriteParam : BOOL;
    fbADSWRITE             : ADSWRITE;
    oidTrafo               : UDINT := 16#01010170; (*Trafo object id*)
    pidTrafo               : UDINT := 16#05010020; (*parameter id*)
    fParamValue            : LREAL;
END_VAR
```

```
fbFB_UnlockTrafoParam(
    bExecute := bUserExecuteUnlock,
    oidTrafo := oidTrafo,
    bBusy=>,
    bDone=>,
    bError=>,
    nErrorId=> );

(*After unlocking new parameter value can be written*)
fbADSWRITE(
    NETID:='' ,
    PORT:= AMSPORT_R0_NCSAF,
    IDXGRP:=oidTrafo ,
    IDXOFFS:= pidTrafo,
    LEN:=SIZEOF(fParamValue) ,
    SRCADDR:= ADR(fParamValue),
    WRITE:=bUserExecuteWriteParam ,
    TMOUT:= ,
    BUSY=> ,
    ERR=> ,
    ERRID=> );
```

8.1.6 FB_KinLockTrafoParam

Nachdem die Transformationsparameter mit Hilfe von FB_KinUnlockTrafoParam [▶ 73] verändert wurden, sperrt der Funktionsbaustein FB_KinLockTrafoParam erneut die Transformationsparameter, so dass diese nicht mehr beschrieben werden können.

VAR_INPUT

```
VAR_INPUT
    bExecute              : BOOL;
    oidTrafo              : UDINT;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

oidTrafo: Objekt-ID des kinematischen Transformationsobjekts. Siehe Beispiel [▶ 76] unten.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy                : BOOL;
    bDone                : BOOL;
    bError               : BOOL;
    nErrorId             : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

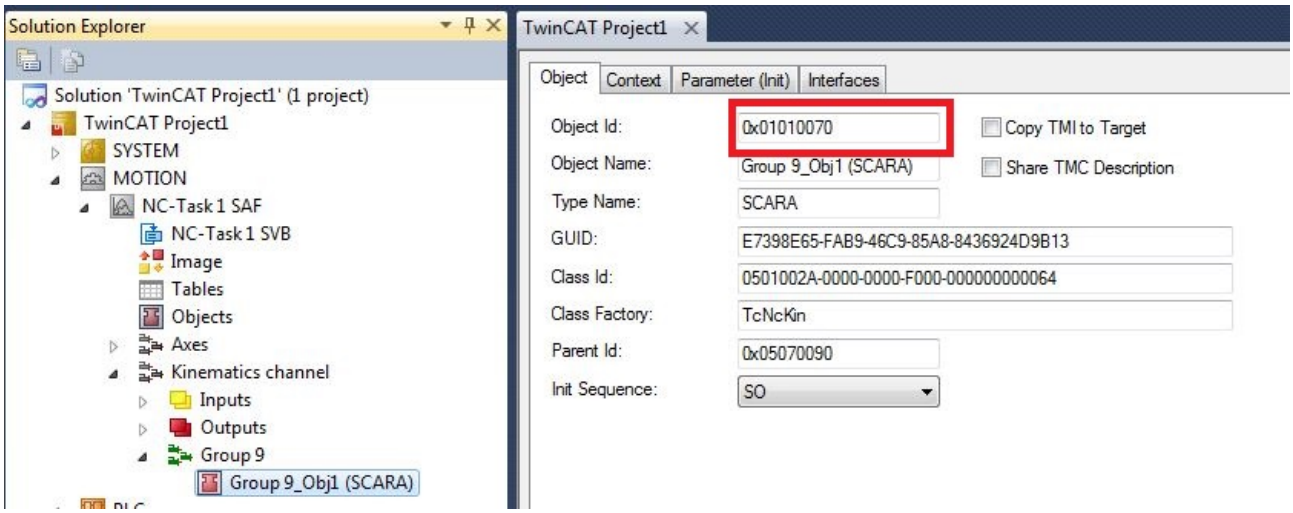
bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

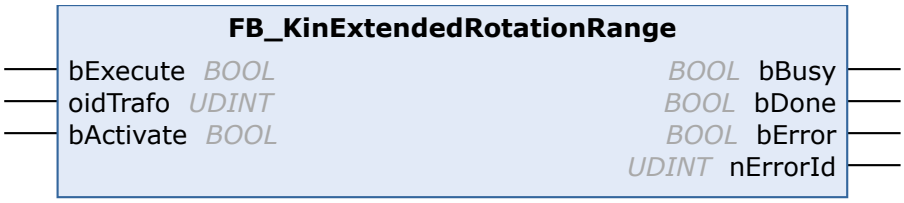
SCARA-Transformation - Beispiel-Objekt-ID




```
VAR
    bUserExecute         : BOOL;
    fbFB_LockTrafoParam  : FB_KinLockTrafoParam;
    oidTrafo             : UDINT := 16#01010070; (*Trafo object id*)
END_VAR
```

```
fbFB_LockTrafoParam(
    bExecute := bUserExecute,
    oidTrafo := oidTrafo,
    bBusy=>,
    bDone=>,
    bError=>,
    nErrorId=> );
```

8.1.7 FB_KinExtendedRotationRange



Erweiterter Drehbereich

✓ Für eine eindeutige Lösung ist der Standard-Drehbereich begrenzt auf:

a) Rotation1: -180 bis 180 Grad,

b) Rotation2: -90 bis 90 Grad,

c) Rotation3: -180 bis 180 Grad.

⇨ In manchen 6-Achs-Anwendungen ist es wünschenswert, über diesen Drehbereich hinaus rotieren zu können.
Die Funktionsbausteine FB_KinExtendedRotationRange und FB_KinPresetRotation ermöglichen es, den Rotationszustand über die Default-Werte hinaus auszuweiten, zu speichern und wiederherzustellen.

Der Funktionsbaustein FB_KinExtendedRotationRange speichert und stellt den Rotationszustand der kinematischen Gruppe wieder her.

Wenn der Bustein mit bActivate:=TRUE ausgeführt wird, wird der Rotationszustand gespeichert, bis die kinematische Gruppe aufgelöst wird. Wenn die kinematische Gruppe anschließend gebaut oder zurückgesetzt wird, wird der gespeicherte Rotationszustand wiederhergestellt. Für den Fall, dass die Rotation deutlich abweicht (>10.0 Grad pro Achse), wird der gespeicherte Rotationszustand nicht wiederhergestellt und FB_KinConfigGroup oder FB_KinResetGroup schlagen mit dem Fehler 0x815D fehl.

Wenn der Baustein mit bActivate:=FALSE ausgeführt wird, wird der Rotationszustand nicht gespeichert oder wiederhergestellt (Default-Verhalten).

VAR_INPUT

```
VAR_INPUT
    bExecute        : BOOL;
    oidTrafo        : UDINT;
    bActivate       : Bool;
END_VAR
```

bExecute: Mit einer steigenden Flanke wird das Kommando ausgeführt.

oidTrafo: Objekt ID (OTCID) des kinematischen Transformationsobjekts.

bActivate: Wenn auf TRUE gesetzt, wird der erweiterte Drehbereich aktiviert.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy                   : BOOL;
    bDone                   : BOOL;
    bError                  : BOOL;
    nErrorId                : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC oder CX (x86 oder x64)	Tc2_KinematicTransformation (V3.2.7.3 oder neuer)

8.1.8 FB_KinPresetRotation

Der Funktionsblock FB_KinPresetRotation setzt den Rotationszustand.

Der Rotationszustand ist nicht persistent und muss gegebenenfalls nach einem TwinCAT Neustart neu gesetzt werden oder wenn ein Pfad nach einer ACS-Achsbewegung (Direct Mode) begonnen wird.

Erweiterter Drehbereich

✓ Für eine eindeutige Lösung ist der Standard-Drehbereich begrenzt auf:

a) Rotation1: -180 bis 180 Grad,

b) Rotation2: -90 bis 90 Grad,

c) Rotation3: -180 bis 180 Grad.

⇨ In manchen 6-Achs-Anwendungen ist es wünschenswert, über diesen Drehbereich hinaus rotieren zu können.
Die Funktionsbausteine FB_KinExtendedRotationRange und FB_KinPresetRotation ermöglichen es, den Rotationszustand über die Default-Werte hinaus auszuweiten, zu speichern und wiederherzustellen.

VAR_INPUT

```
VAR_INPUT
    bExecute        : BOOL;
    oidTrafo        : UDINT;
    stRotation      : ARRAY[1..3] OF LREAL;
END_VAR
```

bExecute: Mit einer steigenden Flanke wird das Kommando ausgeführt.

oidTrafo: Objekt ID (OTCID) des kinematischen Transformationsobjekts.

stRotation: Voreinstellung von MCS Rotation1, Rotation2 und Rotation3

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy                : BOOL;
    bDone                : BOOL;
    bError               : BOOL;
    nErrorId             : UDINT;
END_VAR
```

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel: Äquivalente Rotationen (gleiche Werkzeugausrichtung)

Rotation1:= -180	Rotation1:= -180
Rotation2:= 45	Rotation2:= 45
Rotation3:= 157.95	Rotation3:= -202.05

FB_KinPresetRotation muss genutzt werden, bevor *FB_KinConfigGroup* oder *FB_KinCalcTrafo* die Vorwärtstransformation ausführen.

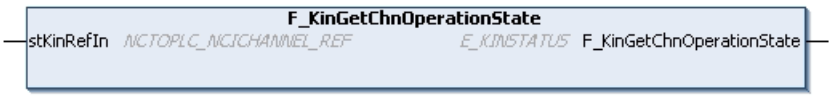
Um den erweiterten Drehbereich mit *FB_KinCalcTrafo(bForward:=TRUE)* ohne eine kinematische Gruppe zu nutzen, muss die Meta-Information uMetaInfo.aData[4] := 1 gesetzt werden.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC oder CX (x86 oder x64)	Tc2_KinematicTransformation (V3.2.7.3 oder neuer)

8.2 Funktionen

8.2.1 F_KinGetChnOperationState



Diese Funktion gibt den Betriebszustand des Kinematikkanals zurück.

Function F_KinGetChnOperationState : E_KINSTATUS

```
VAR_IN_OUT
    stKinRefIn : NCTOPLC_NCICHANNEL_REF
END_VAR
```

stKinRefIn: Bestimmt die Kinematikgruppe der Konfiguration.

Rückgabewert

E_KINSTATUS [▶ 81]: Zustand des Kinematikkanals (siehe unten). Wenn eine ungültige Version des zyklischen Interface verwendet wird, wird *KinStatus_InvalidItfVersion* zurückgegeben.

Beispiel

```
VAR
    stKinRefIn AT %I*       : NCTOPLC_NCICHANNEL_REF;
    nErrId                  : UDINT;
    eKinOperationState      : E_KINSTATUS;
END_VAR
```

```
IF F_KinGetChnOperationState (stKinRefIn)<> KinStatus_InvalidItfVersion THEN
    eKinOperationState := F_KinGetChnOperationState (stKinRefIn);
ELSE
    nErrId := F_KinGetChnOperationState (stKinRefIn);
END_IF
```

8.2.2 F_KinGetAcsMcsAxisIds

Diese Funktion liest die konfigurierten ACS- und MCS-Achsen des zyklischen Interface. Die IDs werden in stAxesList geschrieben.

FUNCTION F_KinGetAcsMcsAxisIds : UDINT

```
VAR_IN_OUT
    stAxesList : ST_KinAxes;
    stKinRefIn : NCTOPLC_NCICHANNEL_REF;
END_VAR
```

stAxesList: Liste der Achs-IDs für das Achskoordinatensystem (ACS) und das Maschinenkoordinatensystem (MCS).

stKinRefIn: Die Struktur des zyklischen Kanalinterface vom Kinematikkanal zur SPS. Auf diese Struktur wird nur lesend zugegriffen.

Rückgabewert

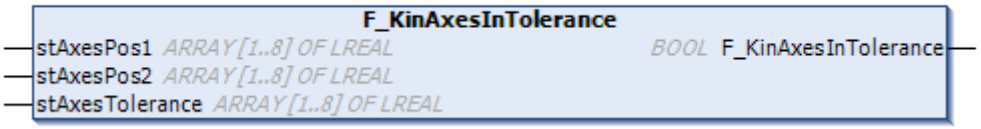
UDINT: Fehlercode

Beispiel

```
VAR
    stAxesList          : ST_KinAxes;
    stKinRefIn AT %I*   : NCTOPLC_NCICHANNEL_REF;
    nErrId              : UDINT;
END_VAR
```

```
nErrId := F_KinGetAcsMcsAxisIds (stAxesList, stKinRefIn);
IF nErrId=0 THEN
    ;(*Axes List is valid*)
END_IF
```

8.2.3 F_KinAxesInTolerance



Der Funktionsblock F_KinAxesInTolerance vergleicht zwei Arrays elementweise.

Die Funktion gibt TRUE zurück, wenn die Differenz zwischen den jeweiligen Array-Elementen in der erwarteten Toleranz liegt.

VAR_INPUT

```
VAR_INPUT
    stAxesPosIn1    : ARRAY[1..8] OF LREAL;
    stAxesPosIn2    : ARRAY[1..8] OF LREAL;
    stAxesTolerance : ARRAY[1..8] OF LREAL;
END_VAR
```

stAxesPosIn1: Erstes Array. Dieses wird mit dem zweiten Array verglichen.

stAxesPosIn2: Zweites Array. Dieses wird mit dem ersten Array verglichen.

stAxesTolerance: Enthält die Toleranz für jedes zu vergleichende Array-Element.

Rückgabewert

BOOL: Die Funktion gibt TRUE zurück, wenn die Differenz zwischen den jeweiligen Array-Elementen in der erwarteten Toleranz liegt.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC oder CX (x86 oder x64)	Tc2_KinematicTransformation (V3.2.7.3 oder neuer)

8.3 Datentypen

8.3.1 ST_KinAxes

Diese Struktur definiert die Achsen, die eine Kinematik bilden.

```
TYPE ST_KinAxes :
STRUCT
    nAxisIdsMcs: ARRAY[1..8] OF DWORD;
    nAxisIdsAcs: ARRAY[1..8] OF DWORD;
END_STRUCT
END_TYPE
```

nAxisIdsMcs: Liste der Achs-IDs der Achsen, die das MCS bilden. Normalerweise beinhalten die ersten drei Array-Elemente die kartesischen Achsen (X,Y,Z) und die folgenden Array-Elemente die Rotationsachsen.

nAxisIdsAcs: Liste der Achs-IDs der Achsen, die das ACS bilden.

Beispiel

```
VAR
    stAxesConfig          : ST_KinAxes;
    io_X                  : AXIS_REF;
    io_Y                  : AXIS_REF;
    io_Z                  : AXIS_REF;
    io_M1                 : AXIS_REF;
    io_M2                 : AXIS_REF;
    io_M3                 : AXIS_REF;
END_VAR
```

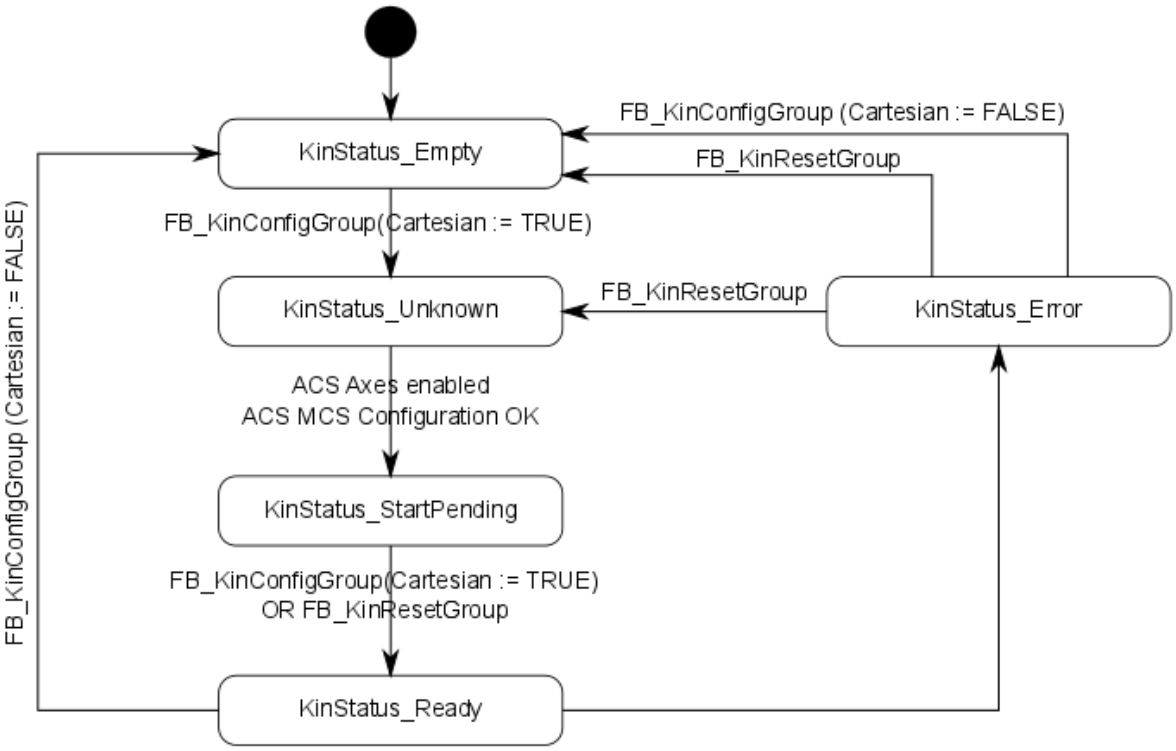
```
(* read the IDs from the cyclic axis interface so the axes can mapped later to the kinematic group
*)
stAxesConfig.nAxisIdsAcs[1] := io_M1.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[2] := io_M2.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[3] := io_M3.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[1] := io_X.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[2] := io_Y.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[3] := io_Z.NcToPlc.AxisId;
```

8.3.2 E_KinStatus

Diese Aufzählung definiert den Zustand der Kinematikgruppe.

```
TYPE E_KinStatus :
(
    KinStatus_Error,
    KinStatus_Empty,
    KinStatus_Unknown,
    KinStatus_StartPending,
```

```
    KinStatus_Ready,
    KinStatus_InvalidItfVersion := 16#4000
);
END_TYPE
```



KinStatus_Empty: ACS-Achsen können bewegt werden. Keine Transformation aktiviert.

KinStatus_Ready: MCS-Achsen können bewegt werden. Transformation aktiv.

KinStatus_InvalidItfVersion: Eine Funktion oder Funktionsbaustein wird nicht von dieser Version des zyklischen Kanalinterface unterstützt. Um die Funktion verwenden zu können, muss eine Aktualisierung vorgenommen werden.

Konfiguration freigeben

Die ACS-Achsen müssen durch MC_Power freigegeben sein, damit der Zustand den Wert **KinStatus_Ready** erreichen kann.

8.3.3 CalcTrafo

8.3.3.1 E_KinMetaInfo5DType1

```
TYPE E_KinMetaInfo5DType1 :
(
    d5Type1Quad14 := 1,
    d5Type1Quad23 := 2,
    d5Type1ActualConfig := 3
);
END_TYPE
```

8.3.3.2 E_KinMetaInfoScara

Enum zur Definition der Armstellung bei einer 4D-SCARA [▶ 41] Kinematik.

```
TYPE E_KinMetaInfoScara :
(
    scaraLeftArm      := 1,
    scaraRightArm     := 2,
    scaraActualConfig := 3
);
END_TYPE
```

8.3.3.3 ST_KinMultiTrafoHeader

Header der Eingangsdatenstruktur für den Funktionsbaustein FB_KinCalcMultiTrafo [▶ 71].

```
Type ST_KinMultiTrafoHeader
STRUCT
    nColumnsIn   : UDINT;
    nColumnsOut  : UDINT;
    nLines       : UDINT;
    bGetMetaInfo : BOOL;
    uMetaInfo    : U_KinMetaInfo;
END_STRUCT
END_TYPE
```

Name	Typ	Beschreibung
nColumnsIn	UDINT	Echte Spaltenanzahl im Eingangsarray.
nColumnsOut	UDINT	Echte Spaltenanzahl im Ausgangsarray.
nLines	UDINT	Anzahl der zu berechenenden Zeilen, kann geringer sein als die tatsächlich deklarierte Zeilenanzahl.
bGetMetaInfo	BOOL	Wenn TRUE, werden MetaInfos mit ausgegeben. Entsprechend muss dann Speicher der Ausgangsdatenstruktur hierfür bereitstehen.
uMetaInfo	U_KinMetaInfo [▶ 83]	

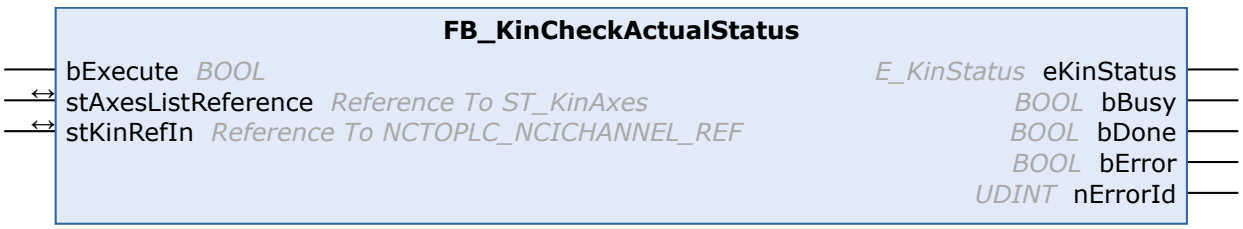
8.3.3.4 U_KinMetaInfo

```
Type U_KinMetaInfo
UNION
    aData    : ARRAY[1..4] OF UDINT;
    eScara   : E_KinMetaInfoScara;
    e5dType1 : E_KinMetaInfo5DType1;
END_UNION
END_TYPE
```

Name	Typ	Beschreibung
aData	ARRAY[1..4] OF UDINT	
eScara	E_KinMetaInfoScara [▶ 82]	Definition der Armstellung bei einer 4D-SCARA [▶ 41] Kinematik.
e5dType1	E_KinMetaInfo5DType1 [▶ 82]	

8.4 Legacy

8.4.1 FB_KinCheckActualStatus



Veraltete Version

Dieser Funktionsbaustein existiert ausschließlich zur Gewährleistung der Kompatibilität mitbestehenden Projekten. Für neue Projekte verwenden Sie bitte F_KinGetChnOperationState [▶ 79]. Dieser Funktionsbaustein benötigt mehr als einen SPS-Zyklus, um den Status des Kinematikkanals zu lesen. Um ihn für jeden Zyklus zu erhalten, benutzen Sie F_KinGetChnOperationState [▶ 79].

Der Funktionsbaustein FB_KinCheckActualStatus gibt den Status des Kinematikkanals zurück.

VAR_INPUT

```
VAR_INPUT
    bExecute              : BOOL;
END_VAR
```

bExecute: Der Befehl wird durch eine steigende Flanke an diesem Eingang ausgelöst.

VAR_IN_OUT

```
VAR_IN_OUT
    stAxesList            : ST_KinAxes;
    stKinRefIn            : NCTOPLC_NCICHANNEL_REF;
END_VAR
```

stAxesList: Bestimmt die ACS- und MCS-Achsen, die in der Konfiguration enthalten sind. Siehe ST_KinAxes.

stKinRefIn: Bestimmt die Kinematikgruppe der Konfiguration.

VAR_OUTPUT

```
VAR_OUTPUT
    eKinStatus            : E_KINSTATUS;
    bBusy                 : BOOL;
    bDone                 : BOOL;
    bError                : BOOL;
    nErrorId              : UDINT;
END_VAR
```

eKinStatus: Gibt den Status des Kinematikkanals zurück. Siehe E_KINSTATUS [▶ 81].

bBusy: Der Ausgang wird TRUE, wenn der Befehl mit *bExecute* gestartet ist und bleibt es dann so lange, wie der Funktionsbaustein den Befehl ausführt. Während *bBusy* gleich TRUE ist, wird an den Eingängen kein neuer Befehl angenommen. Wenn *bBusy* wieder FALSE wird, ist der Funktionsbaustein bereit für einen neuen Befehl. Gleichzeitig wird einer der Ausgänge *bDone* oder *bError* gesetzt.

bDone: Der Ausgang wird TRUE, wenn der Befehl erfolgreich ausgeführt wurde.

bError: Der Ausgang *bError* wird auf TRUE gesetzt, wenn bei der Ausführung des Befehls ein Fehler aufgetreten ist.

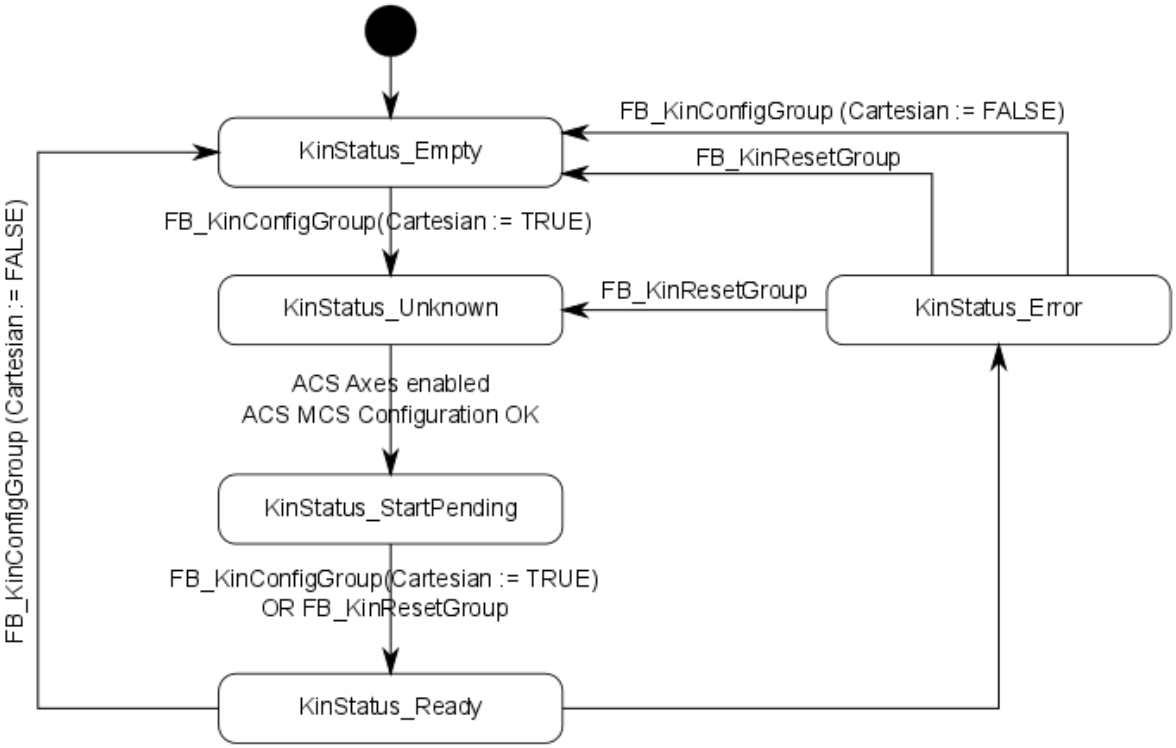
nErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Der Fehlercode kann in der ADS-Fehlerdokumentation oder in der NC-Fehlerdokumentation (Fehlercodes ab 0x4000) nachgeschlagen werden.

Beispiel

```
VAR
    fbFB_KinCheckActualStatus  : FB_KinCheckActualStatus;
    in_stKinToPlc AT %I*       : NCTOPLC_NCICHANNEL_REF;
    stAxesConfig               : ST_KinAxes;
    eKinStatus                 : E_KINSTATUS;
END_VAR
```

```
fbFB_KinCheckActualStatus(
    bExecute            := TRUE,
    stAxesListReference := stAxesConfig,
    stKinRefIn          := in_stKinToPlc,
    eKinStatus          => eKinStatus      );
```

Zustand der Kinematikgruppe



Mehr Informationen:
www.beckhoff.de/tf5110

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