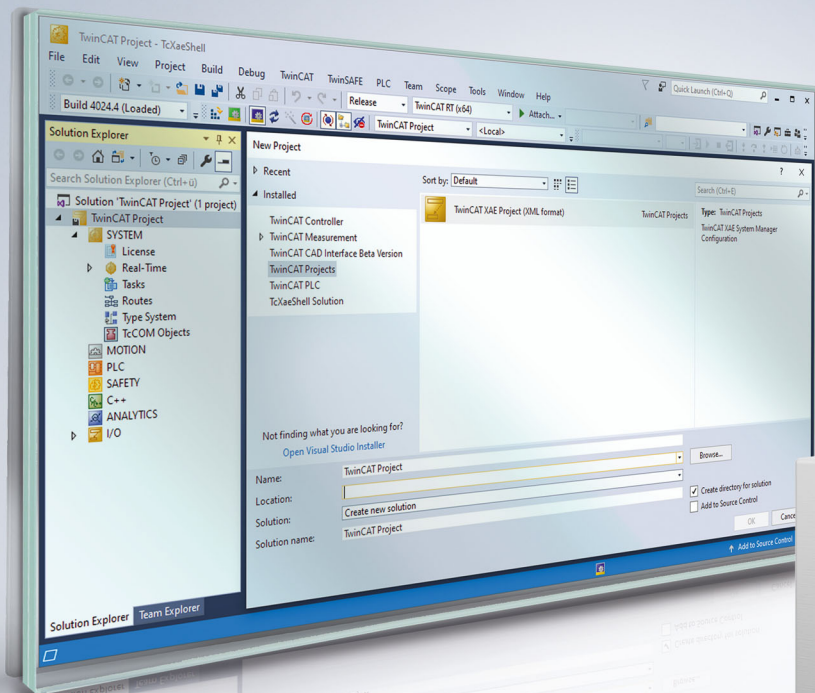


Manual | EN

## TF5110 - TF5113

TwinCAT 3 | Kinematic Transformation





# Table of contents

<b>1 Foreword</b>	<b>5</b>
1.1 Notes on the documentation	5
1.2 For your safety	5
1.3 Notes on information security	7
<b>2 Introduction</b>	<b>8</b>
<b>3 Overview of the new functions</b>	<b>11</b>
<b>4 Installation</b>	<b>12</b>
<b>5 Configuration</b>	<b>13</b>
<b>6 Supported Transformation</b>	<b>17</b>
6.1 General Parameters for the Kinematics	20
6.2 Static Transformation	22
6.3 2D-Kinematics Type 1 (P_2C)	24
6.4 2D-Kinematics Type 2 (P_2C2)	25
6.5 2D-Kinematics Type 3 (S_CC)	27
6.6 2D-Kinematics H-Bot (P_2Y)	28
6.7 2D-Kinematics Type 5 (S_CC)	29
6.8 2D-Kinematics Type 6 (P_2X)	30
6.9 2D-Scissor Kinematics Type 1 (P_2X)	31
6.10 3D-Kinematics Type 8 (S_CCC)	32
6.11 3D-Delta Type 1 (P_3C)	34
6.12 3D-Delta Type 2 (P_3C2)	36
6.13 3D-Delta T Type 3 (P_3C3)	37
6.14 3D-Delta Y Type 4 (P_3C4)	40
6.15 3D-Tripod Type 1 (P_3Z)	41
6.16 3D-Tripod Type 2 (P_3L)	43
6.17 3D-Cable Kinematics Type 1 (P_3Z)	44
6.18 3D-Cable Kinematics Type 2 (P_3L)	45
6.19 3D-Kinematics Type 7 (PXX SZ)	46
6.20 4D-SCARA (S_CCZC)	48
6.21 4D-Kinematics Type 6 (S_XCZC)	49
6.22 4D-Cable-Kinematics (P_4L)	50
6.23 5D-Kinematics Type 2 (XYZab)	51
6.24 5D-Kinematics Type 3 (XYZAB)	53
6.25 6D-Stewart Platform (P_6L)	55
6.26 Six Axis Articulated (S_CBBCBC)	57
6.27 Drive Torque	58
6.28 Tool Offset	59
6.29 Tool Linear	60
6.30 Coordinate system (Coordinate Frame)	61
<b>7 User-specific transformations</b>	<b>65</b>
<b>8 Plc Library</b>	<b>77</b>
8.1 Function Blocks	78

8.1.1	FB_KinConfigGroup .....	78
8.1.2	FB_KinResetGroup .....	80
8.1.3	FB_KinCalcTrafo .....	82
8.1.4	FB_KinCalcMultiTrafo .....	84
8.1.5	FB_KinUnlockTrafoParam .....	86
8.1.6	FB_KinLockTrafoParam .....	89
8.1.7	FB_KinExtendedRotationRange .....	90
8.1.8	FB_KinPresetRotation .....	91
8.2	Functions .....	93
8.2.1	F_KinGetChnOperationState .....	93
8.2.2	F_KinGetAcsMcsAxisIds .....	94
8.2.3	F_KinAxesInTolerance .....	94
8.3	Datatypes .....	95
8.3.1	ST_KinAxes .....	95
8.3.2	E_KinStatus .....	96
8.3.3	CalcTrafo .....	96
8.4	Legacy .....	98
8.4.1	FB_KinCheckActualStatus .....	98



# 1 Foreword

## 1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.

The documentation and the following notes and explanations must be complied with when installing and commissioning the components.

The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

### Disclaimer

The documentation has been compiled with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without notice.

Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

### Trademarks

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar®, and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

If third parties make use of the designations or trademarks contained in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.



EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The distribution and reproduction of this document, as well as the use and communication of its contents without express authorization, are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event that a patent, utility model, or design are registered.

### Third-party trademarks

Trademarks of third parties may be used in this documentation. You can find the trademark notices here: <https://www.beckhoff.com/trademarks>.

## 1.2 For your safety

### Safety regulations

Read the following explanations for your safety.

Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

### Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.


## Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

## Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

### Personal injury warnings

 <b>DANGER</b>
Hazard with high risk of death or serious injury.
 <b>WARNING</b>
Hazard with medium risk of death or serious injury.
 <b>CAUTION</b>
There is a low-risk hazard that could result in medium or minor injury.

### Warning of damage to property or environment

<b>NOTICE</b>
The environment, equipment, or data may be damaged.

### Information on handling the product



This information includes, for example:  
recommendations for action, assistance or further information on the product.

## **1.3 Notes on information security**

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our <https://www.beckhoff.com/secguide>.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at <https://www.beckhoff.com/secinfo>.

## 2 Introduction

The TF5110 - TF5113 TwinCAT Kinematic Transformation software package is installed together with the TF5400 software package.

### TwinCAT Kinematic Transformation

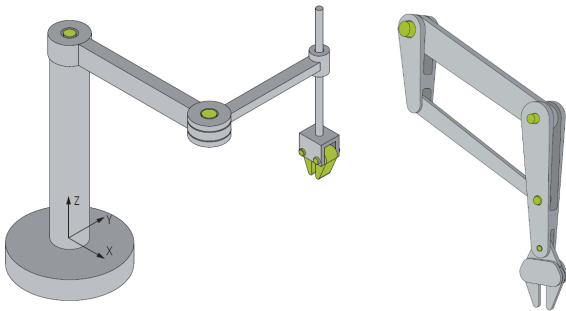
The TF5110 - TF5113 TwinCAT Kinematic Transformation is a software solution that combines robot control and conventional PLC in one system (see also <https://www.beckhoff.com/tf5113/>). The implementation of the entire control in one system eliminates interface losses between different CPUs for PLC, motion control and robot control. In practice, this implementation leads to a reduction of engineering costs and to reduced cycle times in the production process. In addition to the elimination of interfaces and components, the merging of PLC, robotics and motion control into one application makes the system homogeneous. Therefore, for the user there is no apparent difference in the treatment of the individual functions. Conveniently, a part on a conveyor belt operated with standard motion control can be taken and set aside by the robot quickly and handily.

Since the working area of the robot is determined by the configuration and the number of axes, it depends on a number of parameters: arm lengths, angular range, center of mass, maximum load, etc. The configuration of the arms and joints determines the kinematic structure, which is divided in two main classes: serial kinematics and parallel kinematics.

### Serial kinematics

The current position of any axis always depends on the position of the preceding axis, i.e. all axes are arranged sequentially.

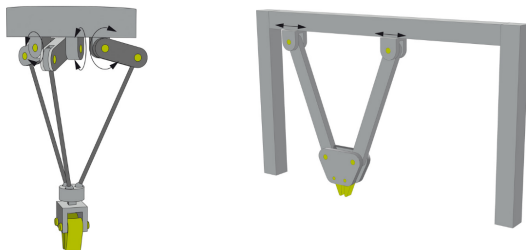
Examples: SCARA and crane kinematics



### Parallel kinematics

All axes directly engage with the working platform via the kinematics.

Examples: delta kinematics, shear kinematics

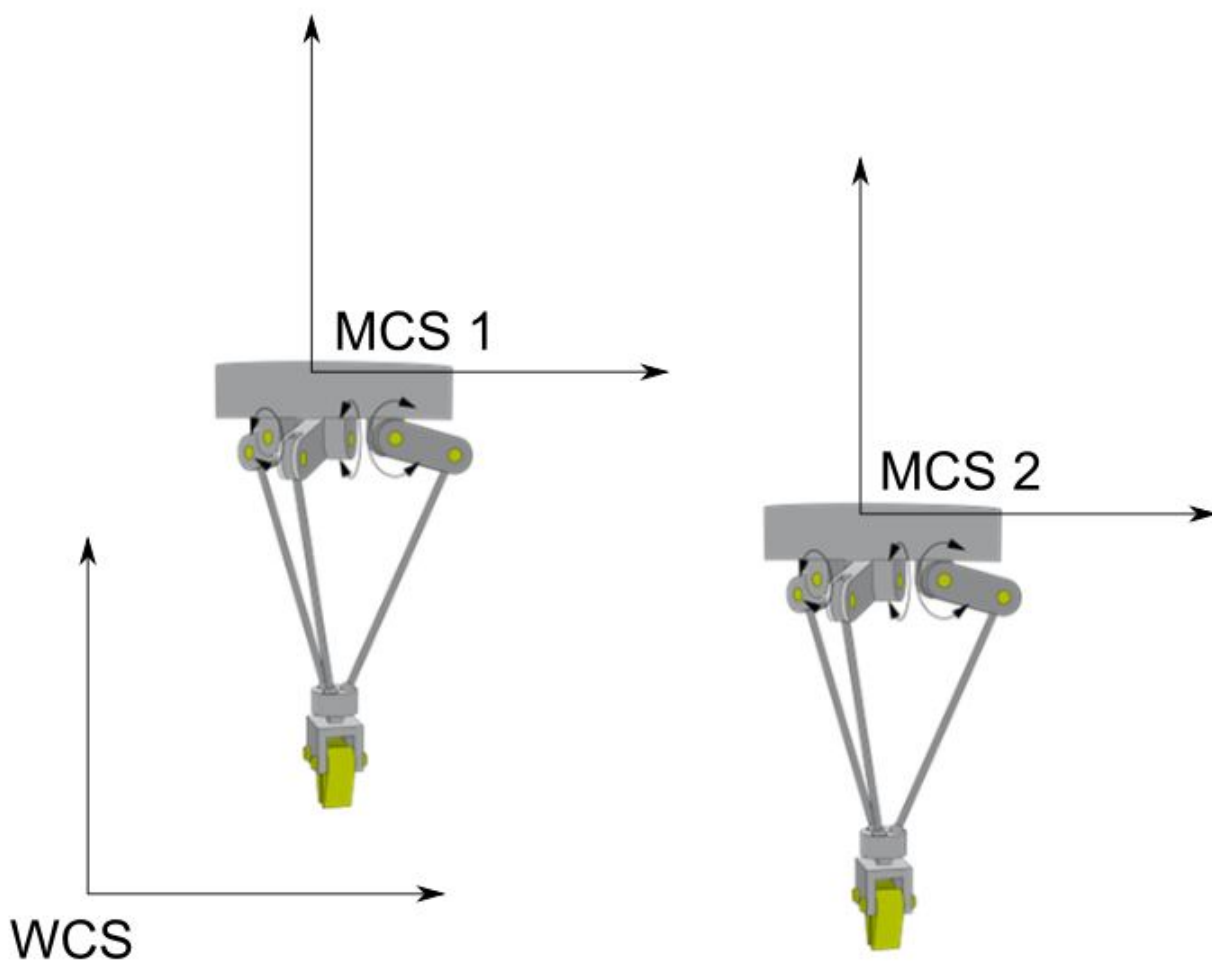


### Coordinate systems

Coordinate systems are required in order to describe the positional behavior of a system. Different coordinate systems can be used as a basis for programming:

- The machine coordinate system (MCS) is a robot-based cartesian coordinate system, which usually has its origin in the robot base.

- The world coordinate system (WCS ) is a cartesian coordinate system, which describes the whole modelled 'world'. It therefore does not refer to a specific robot, but to the whole system. The origin of a robot-based machine coordinate system (MCS) is at a particular point of the WCS. In other words, the user can specify, at which point of his "world" an industrial robot is located and how it is oriented. A WCS can contain several robots. When using a robot, world coordinates can coincide with the machine coordinates to improve transparency.
- The user can position the user coordinate system (UCS) at any position and with any orientation within the world coordinate system.
- The axis coordinate system (ACS) describes the position of the physical axes. It is generally not a cartesian coordinate system. Many robot joint axes perform rotary movements. Using the ACS makes it easier to take into account the limit values for angle, velocity and acceleration. If a robot axis performs rotary movements, it is often difficult for the user to predict and control the path. The axis coordinate system is usually used for referencing/homing.



### Kinematic transformation

In many cases, robots are programmed in the MCS. Due to the way humans think, movements are usually programmed in Cartesian coordinate systems. To execute such movements, it is therefore necessary to convert between the axis coordinate system and the Cartesian space.

Transformation describes, in the context of the kinematics, the calculation necessary in order to change from one coordinate system to another. There are basically two problems in considering the kinematics of robots:

- The conversion from the axis coordinate system (ACS) to a Cartesian coordinate system is referred to as forward transformation. The Cartesian position of the tool center point (TCP) is calculated from the axis-specific joint coordinates of the robot.
- The conversion from Cartesian coordinates of the TCP to axis coordinates, which is required in order to move the actual robot axes, is referred to as backward transformation.

## Realization in TwinCAT

TwinCAT Kinematic Transformation can be used to realize robotics applications. All PLC and NC features can be combined on a common hardware and software platform. TwinCAT Kinematic Transformation realizes several robot kinematics (e.g. H-Bot, delta robot, 6-axis robot) on the PC. The axes are controlled directly from the TwinCAT Motion Control system.

The user can program robot movements directly in the Cartesian coordinate system. The software calculates the transformation to the axis coordinate system of the robot in each cycle. To minimize vibrations and to increase the positioning accuracy, for many kinematics a current pre-control can be activated, if the drive amplifier and the fieldbus are fast enough and interfaces for an additional current pre-control are available. EtherCAT and the Beckhoff servo axes meet these requirements.

The TwinCAT function seamlessly integrates into the motion control world of Beckhoff. [TF5420 TwinCAT 3 Motion Pick-and-Place](#) is available especially for the simple realization of handling tasks. [TF5100 TwinCAT 3 NC I](#) enables programming both via G-Code (DIN 66025) and directly from the PLC ([PlcInterpolation](#) library). The functions [TF5055 TwinCAT 3 NC Flying Saw](#) and [TF5050 TwinCAT 3 NC Camming](#) enable synchronization with conveyor belts for picking and placing of workpieces, for example. In addition, standard PTP functions from the familiar Beckhoff PTP motion libraries can be used.

The configuration of the robot takes place entirely in the TwinCAT 3 Engineering environment (XAE).

### 3 Overview of the new functions

**From V3.3.57:**

- New kinematics:
  - [3D-Delta Y Type 4 \(P\\_3C4\)](#) [[▶ 40](#)]
  - [3D-Kinematics Type 8 \(S\\_CCC\)](#) [[▶ 32](#)]

**From V3.3.25:**

- New kinematics:
  - [2D-Scissor Type 1 \(P\\_2X\)](#) [[▶ 31](#)]

**From V3.1.10.66:**

- New kinematics:
  - [3D-Tripod Type 1 \(P\\_3Z\)](#) [[▶ 41](#)]
  - [3D-Tripod Type 2 \(P\\_3L\)](#) [[▶ 43](#)]

**From V3.1.10.63:**

- Requires TwinCAT V3.1.4024.24 or higher

**From V3.1.10.30:**

- New kinematics:
  - [3D-Kinematics Type 7 \(PXX\\_SZ\)](#) [[▶ 46](#)]
  - [3D-Delta T-Type 3 \(P\\_3C3\)](#) [[▶ 37](#)]
  - [3D-Cable Kinematics Type 2 \(P\\_3L\)](#) [[▶ 45](#)]
  - [4D-Kinematics Type 6 \(S\\_XCZC\)](#) [[▶ 49](#)]
  - [4D-Cable Kinematics \(P\\_4L\)](#) [[▶ 50](#)]

**From V3.1.10.1:**

- New function blocks for Extended Rotation Range have been implemented:
  - FB\_KinPresetRotation
  - FB\_KinExtendedRotationRange
- New function F\_KinAxesInTolerance
- Requires TwinCAT V3.1.4024.7 or higher

**From V3.1.6.3:**

- The TF5400 installation package will include the TF511x TwinCAT 3 Kinematic Transformation.

## 4 Installation

The installation of TF511x Kinematic Transformation differs from version TwinCAT 3.1 Build 4024 to the previous versions.

### TwinCAT Package Manager: Installation (TwinCAT 3.1 Build 4026)

From TwinCAT 3.1 Build 4026, TwinCAT products are installed via the TwinCAT Package Manager. Detailed instructions on installing products can be found in the chapter [Installing workloads](#) in the [TwinCAT 3.1 Build 4026 installation instructions](#).

You can obtain a basic TwinCAT 3 installation via the following workloads:

- TwinCAT.Standard.XAE (Engineering)
- TwinCAT.Standard.XAR (Runtime)

For TF5110-TF5112 | TwinCAT 3 Kinematic Transformation L1-L3 please install the workloads:

- TF5400.AdvancedMotionPack.XAE (Engineering)
- TF5400.AdvancedMotionPack.XAR (Runtime)



#### TF5113 | TwinCAT Kinematic Transformation L4

TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload. If required, please get in touch with your sales contact.

---

### TwinCAT setup: Installation (TwinCAT 3.1 Build 4024 and earlier)

TF5110 - TF5112 | TwinCAT 3 Kinematic Transformation L1-L3 is installed up to and including TwinCAT 3.1 Build 4024 with the setup [TF5400 TwinCAT 3 Advanced Motion Pack](#).



#### TF5113 | TwinCAT Kinematic Transformation L4

TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400 TwinCAT Advanced Motion Pack Setup from the web pages. If required, please get in touch with your sales contact.

---

### Function level for TF5110-TF5113

The function TF5110 - TF5113 TwinCAT 3 Kinematic Transformation is divided into four different levels depending on the number of transformation axes. A higher level includes all sublevels.

**Level 1:** supports the static transformation. This includes a translation and rotation of the coordinate system.

**Level 2:** supports Level 1 and simple (mainly 2D) kinematic transformations such as H-Bot and 2D parallel kinematics.

**Level 3:** Supports level 2 and more sophisticated kinematic transformations (3D, 4D), such as delta robots.

**Level 4:** supports Level 3 and complex kinematic transformations (up to 6D).

### Additional licensing requirements

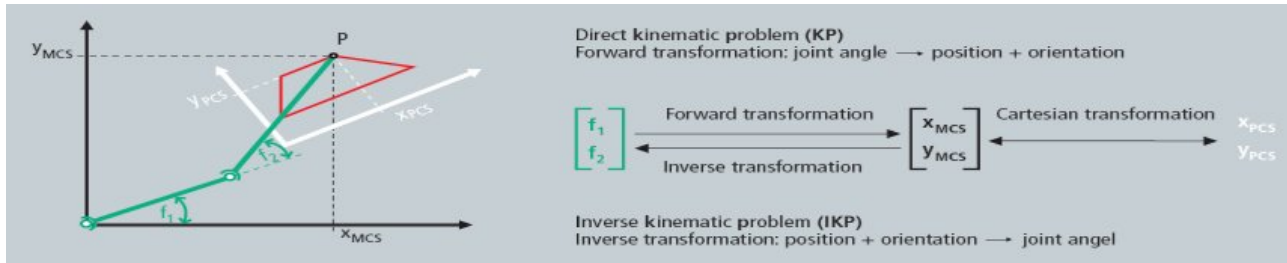
TF5110-TF5113 TwinCAT 3 Kinematic Transformation requires the TC1260 license.



## 5 Configuration

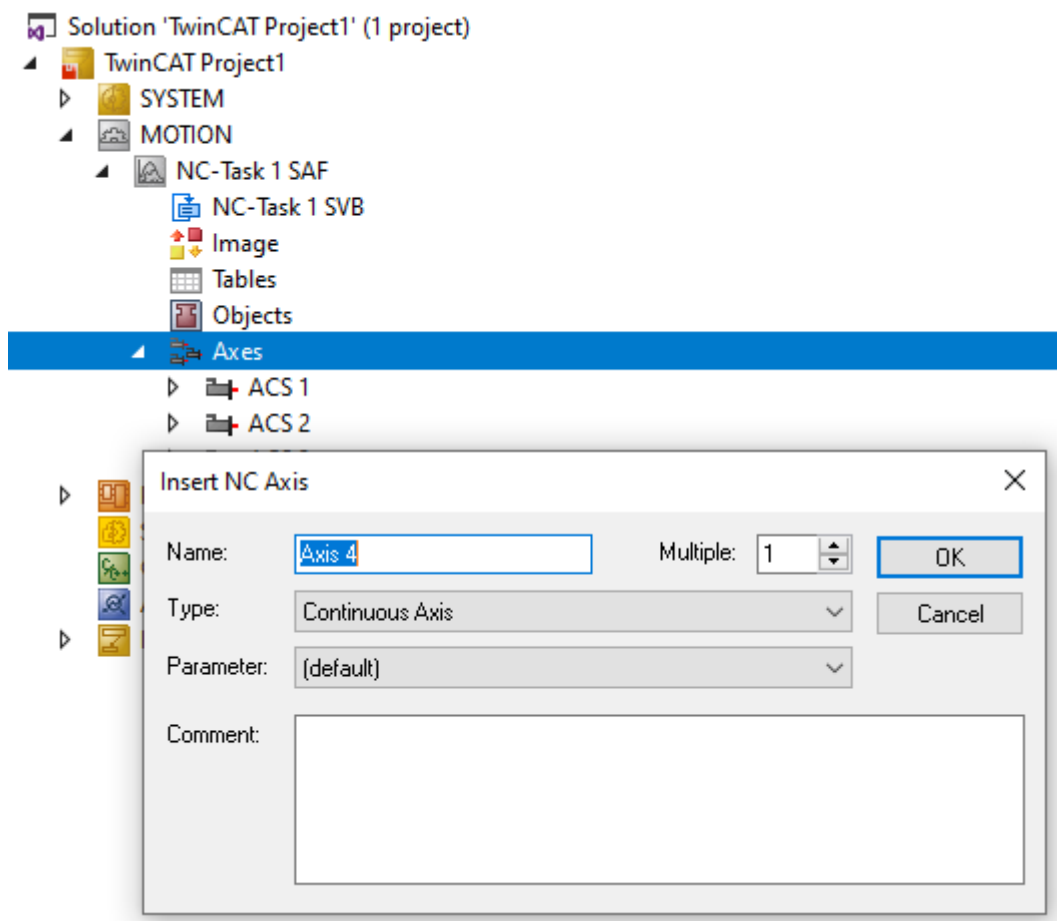
Based on PLCopen, we distinguish between two main coordinate systems (see description [Introduction](#) [► 8]):

- Axis coordinate system (ACS)
- Machine coordinate system (MCS)

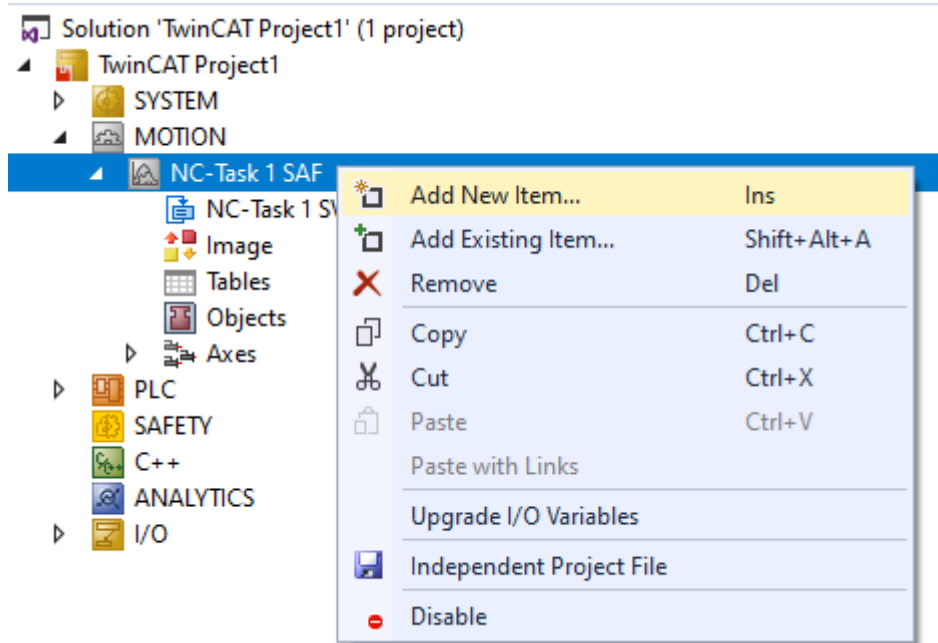


### Configuring the kinematic transformation channel

1. Add all axes (ACS and MCS) to the NC configuration in the XAE, just like PTP axes. The axes of the ACS are hardware axes and are linked with drives. The axes of the MCS are pure software axes of the simulation encoder type. All ACS and MCS axes that are used in a kinematic transformation channel must be created in the XAE. A delta robot, for example, has three ACS axes (M1...M3) and three MCS axes (X, Y, Z).
2. Right-click on "Axes" and select "Add new item".
3. Then create the axes in the "Insert NC Axis" window, according to the kinematics.

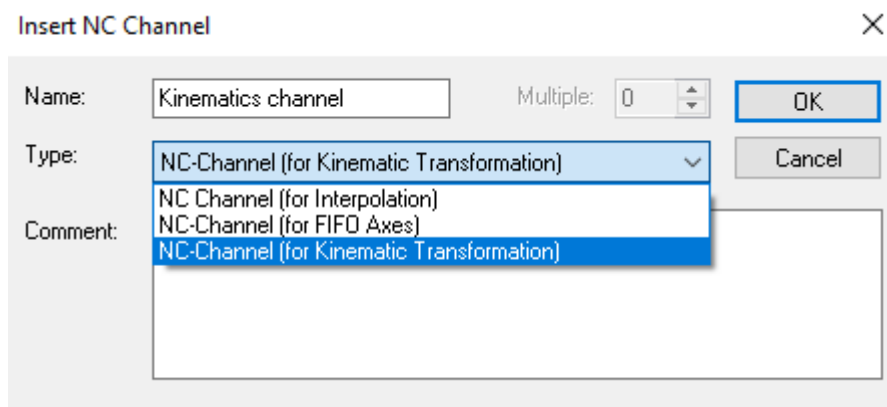


4. Add a kinematic channel to the XAE configuration.

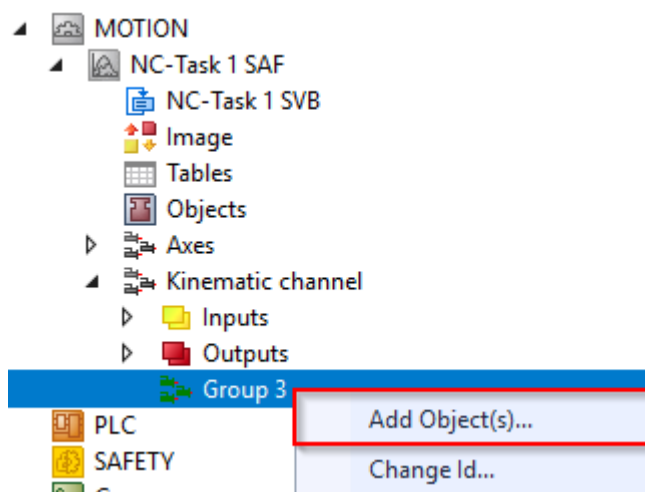


⇒ The addition of a channel creates an instance of a kinematic group.

5. Select the channel type: **NC channel (for kinematic transformation)** for performing a kinematic transformation.



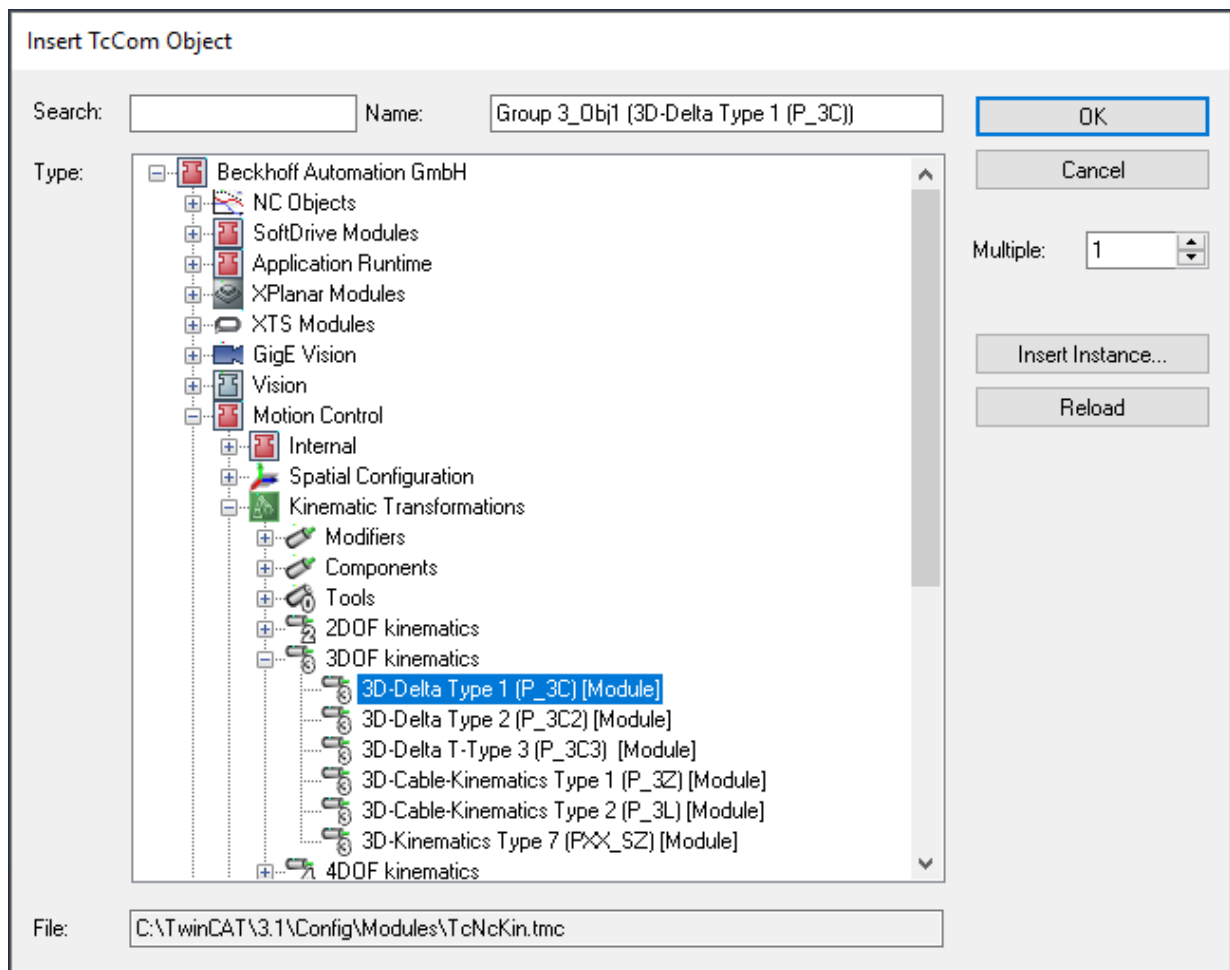
6. Add the objects under the group representing the kinematic configuration of the user.



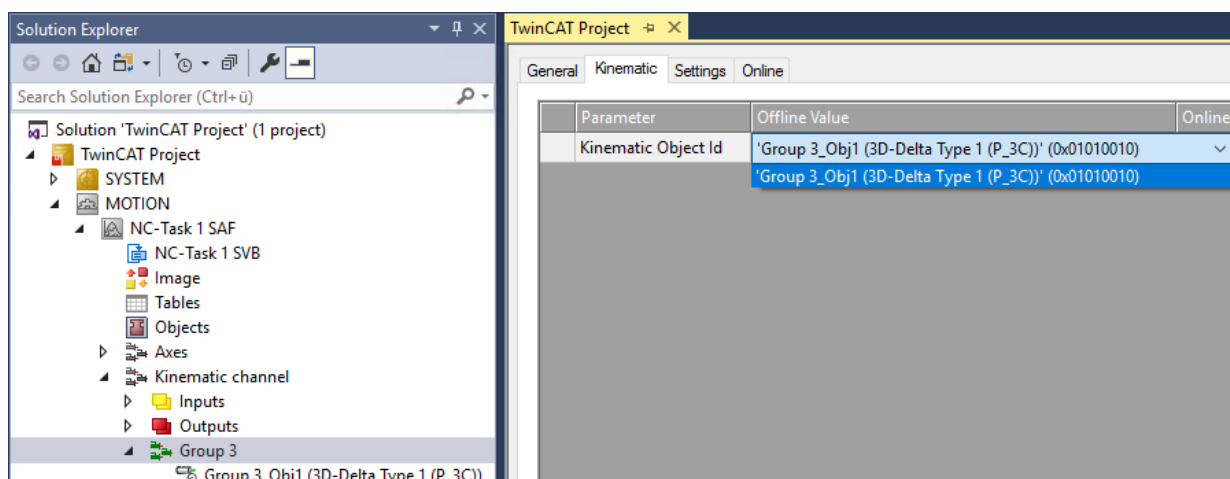
7. To start the transformation for a delta robot, select

- Delta Type 1

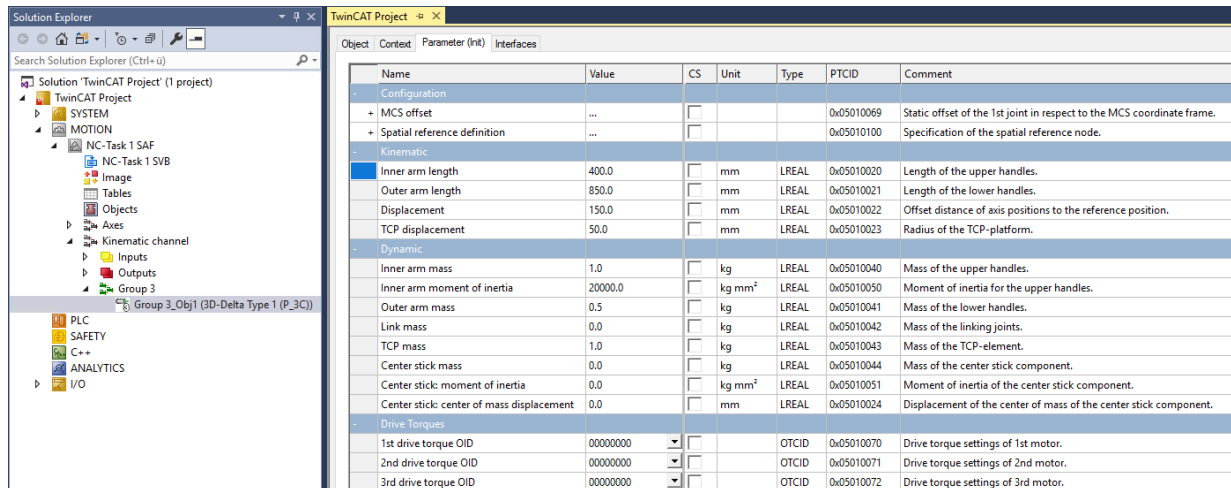
In addition, optional [tools](#) [\[► 59\]](#) and coordinate systems (UCS) can be created.



8. The transformation group must know which root module is to be called. This is why the object ID of the kinematics (in this case Delta Type 1) must be selected. The kinematic object defines the number of ACS and MCS axes to be used in the PLC (see [ST KinAxes](#) [\[► 95\]](#)).



## 9. Parameterize the object parameters according to the kinematics used.



Name	Value	CS	Unit	Type	PTCID	Comment
<b>Configuration</b>						
+ MCS offset	...	<input type="checkbox"/>			0x05010069	Static offset of the 1st joint in respect to the MCS coordinate frame.
+ Spatial reference definition	...	<input type="checkbox"/>			0x05010100	Specification of the spatial reference node.
<b>Kinematic</b>						
Inner arm length	400.0	<input type="checkbox"/>	mm	LREAL	0x05010020	Length of the upper handles.
Outer arm length	850.0	<input type="checkbox"/>	mm	LREAL	0x05010021	Length of the lower handles.
Displacement	150.0	<input type="checkbox"/>	mm	LREAL	0x05010022	Offset distance of axis positions to the reference position.
TCP displacement	50.0	<input type="checkbox"/>	mm	LREAL	0x05010023	Radius of the TCP-platform.
<b>Dynamic</b>						
Inner arm mass	1.0	<input type="checkbox"/>	kg	LREAL	0x05010040	Mass of the upper handles.
Inner arm moment of inertia	20000.0	<input type="checkbox"/>	kg mm <sup>2</sup>	LREAL	0x05010050	Moment of inertia for the upper handles.
Outer arm mass	0.5	<input type="checkbox"/>	kg	LREAL	0x05010041	Mass of the lower handles.
Link mass	0.0	<input type="checkbox"/>	kg	LREAL	0x05010042	Mass of the linking joints.
TCP mass	1.0	<input type="checkbox"/>	kg	LREAL	0x05010043	Mass of the TCP-element.
Center stick mass	0.0	<input type="checkbox"/>	kg	LREAL	0x05010044	Mass of the center stick component.
Center stick: moment of inertia	0.0	<input type="checkbox"/>	kg mm <sup>2</sup>	LREAL	0x05010051	Moment of inertia of the center stick component.
Center stick: center of mass displacement	0.0	<input type="checkbox"/>	mm	LREAL	0x05010024	Displacement of the center of mass of the center stick component.
<b>Drive Torques</b>						
1st drive torque OID	00000000	<input type="checkbox"/>		OTCID	0x05010070	Drive torque settings of 1st motor.
2nd drive torque OID	00000000	<input type="checkbox"/>		OTCID	0x05010071	Drive torque settings of 2nd motor.
3rd drive torque OID	00000000	<input type="checkbox"/>		OTCID	0x05010072	Drive torque settings of 3rd motor.

10. The transformation can now be activated via the PLC (see [Plc Library \[► 77\]](#)). To actuate the transformation, define a cyclic channel interface in the PLC and link it with the I/O of the kinematic channel.

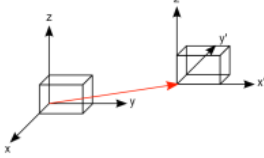
```

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




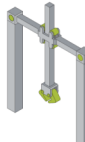
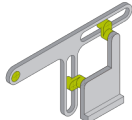
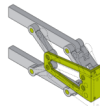

```

## 6 Supported Transformation

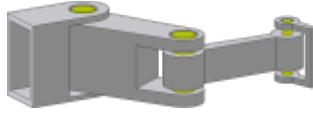


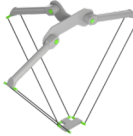
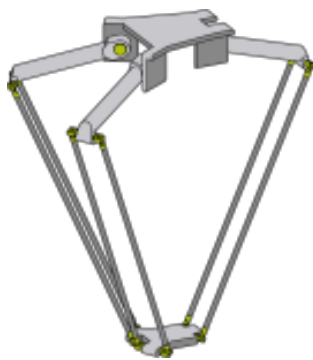

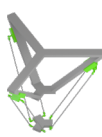
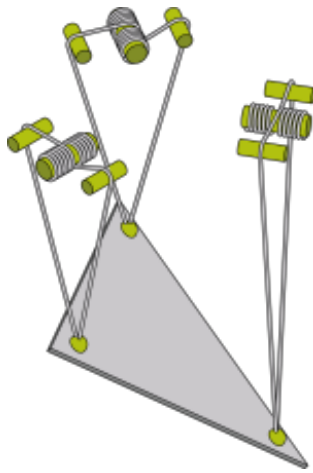
### Overview



Transformation type	Schema	Required TwinCAT function (level)
<a href="#">Static Transformation</a> [ <a href="#">P 22</a> ]		TF5110 TwinCAT 3 Kinematic Transformation (Level 1)

### 2D kinematic transformations

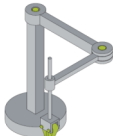
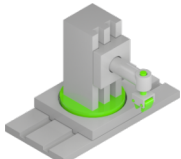
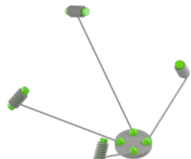
Transformation type	Schema	Required TwinCAT function (level)
<a href="#">2D-Kinematics Type 1 (P 2C)</a> [ <a href="#">P 24</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Kinematics Type 2 (P 2C2)</a> [ <a href="#">P 25</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Kinematics Type 3 (S CC)</a> [ <a href="#">P 27</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Kinematics H-Bot (P 2Y)</a> [ <a href="#">P 28</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Kinematics Type 5 (S CC)</a> [ <a href="#">P 29</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Kinematics Type 6 (P 2X)</a> [ <a href="#">P 30</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)
<a href="#">2D-Scissor Kinematics Type 1 (P 2X)</a> [ <a href="#">P 31</a> ]		TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

**3D kinematic transformations**

Transformation type	Schema	Required TwinCAT function (level)
<u>3D-Kinematics Type 8</u> <u>(S CCC) [► 32]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Delta Type 1 (P 3C)</u> <u>[► 34]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Delta Type 2 (P 3C2)</u> <u>[► 36]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Delta T Type 3 (P 3C3)</u> <u>[► 37]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Delta Y Type 4 (P 3C4)</u> <u>[► 40]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Tripod Type 1 (P 3Z)</u> <u>[► 41]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Tripod Type 2 (P 3L)</u> <u>[► 43]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
<u>3D-Cable Type 1 (P 3Z)</u> <u>[► 44]</u>		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

Transformation type	Schema	Required TwinCAT function (level)
3D-Cable Type 2 (P 3L) [► 45]		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
3D-Kinematics Type 7 (PXX SZ) [► 46]		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

#### 4D kinematic transformations


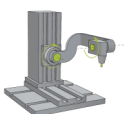


Transformation type	Schema	Required TwinCAT function (level)
4D-SCARA (S CCZC) [► 48]		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
4D-Kinematics Type 6 (S XCZC) [► 49]		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)
4D-Cable (4D P 4L) [► 50]		TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

#### 5D and 6D kinematic transformations (export limited)



##### TF5113 | TwinCAT Kinematic Transformation L4

TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload or included in the TF5400 TwinCAT Advanced Motion Pack Setup from the website. If required, please get in touch with your sales contact.

Transformation type	Schema	Required TwinCAT function (level)
5D-Kinematics Type 2 (XYZab) [► 51]		TF5113 TwinCAT 3 Kinematic Transformation (Level 4)
5D-Kinematics Type 3 (XYZAB) [► 53]		TF5113 TwinCAT 3 Kinematic Transformation (Level 4)
Stewart Platform (P 6L) [► 55]		TF5113 TwinCAT 3 Kinematic Transformation (Level 4)
Six Axis Articulated (S CBBCBC) [► 57]		TF5113 TwinCAT 3 Kinematic Transformation (Level 4)

## Additional objects

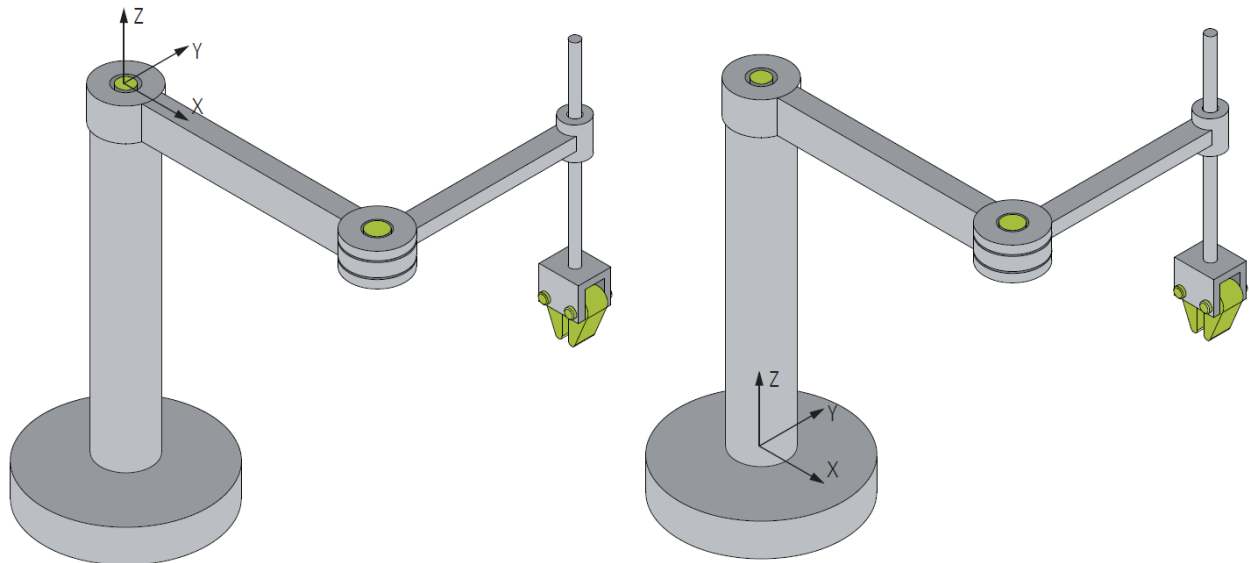
The following objects can be created and selected in the corresponding kinematics. A drop-down parameter list in the kinematics is used for the selection. Select the corresponding object ID (OTCID).

Object type	Description	Required level and version
<a href="#">Tool Offset</a> [► 59]	Tool offset - describes a tool at the level of the kinematics flange.	TF5110 TwinCAT 3 Kinematic Transformation (Level 1)
<a href="#">Tool Linear</a> [► 60]	Linear tool - describes a 1D tool mounted on the kinematics flange, which offers the option to move the TCP towards the tool.	TF5110 TwinCAT 3 Kinematic Transformation (Level 1)
<a href="#">Drive Torque</a> [► 58]	Drive torque - represents the inertia and the efficiency of the motor and drive, to enable more precise calculation of the dynamic model.	TF5110 TwinCAT 3 Kinematic Transformation (Level 1)
<a href="#">Coordinate Frame</a> [► 61]	Coordinate system - describes a user-defined coordinate system.	TF5110 TwinCAT 3 Kinematic Transformation (Level 1)

## 6.1 General Parameters for the Kinematics

### MCS Offset

The MCS offset can be used to parameterize additional offset parameters before the first axis (or before the basis) of the kinematics. For example, in the SCARA kinematics the origin of the MCS is located in the first joint (M1). The parameter Z-shift of the MCS offset can be used to parameterize the additional bar length so that the origin of the MCS resides at the robot base.



Parameter	Description	Type	Unit
X-shift	Static X-offset in the MCS.	LREAL	mm
Y-shift	Static Y-offset in the MCS.	LREAL	mm
Z-shift	Static Z-offset in the MCS.	LREAL	mm

### MCS to Spatial reference

The MCS can be moved in a reference coordinate system using the Spatial reference parameter. All coordinate systems are right-handed (anticlockwise).

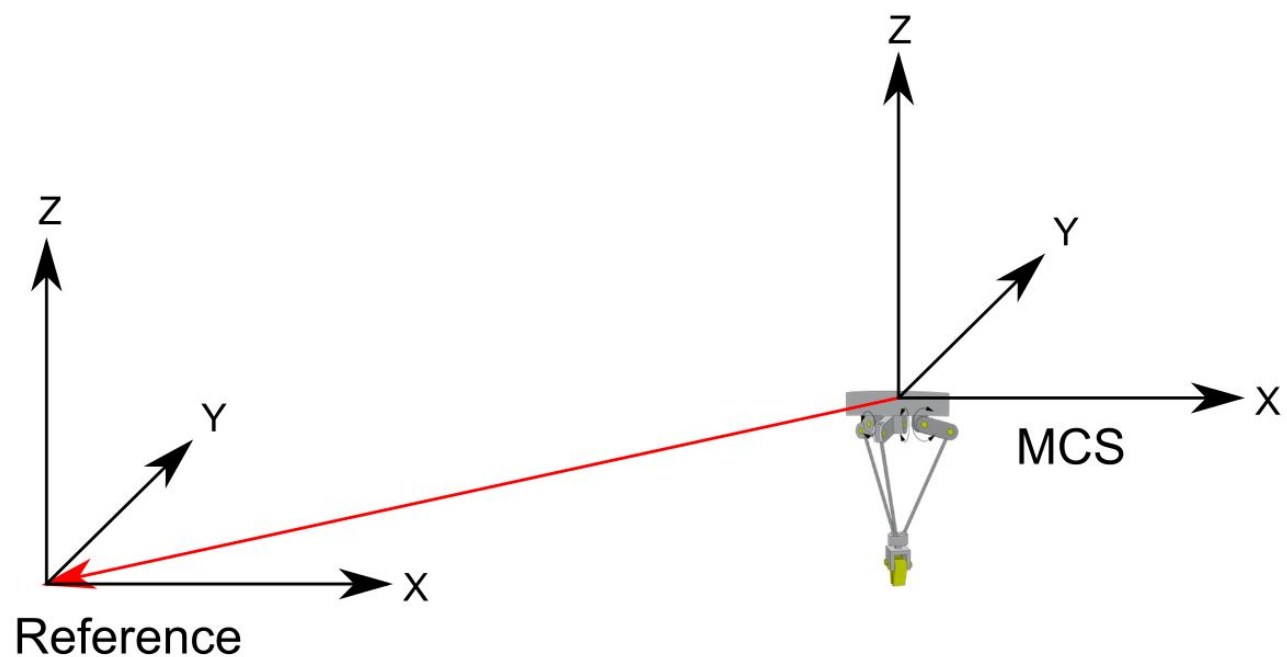
Parameter	Description	Type	Unit
Translation X	Translation in X direction.	LREAL	mm



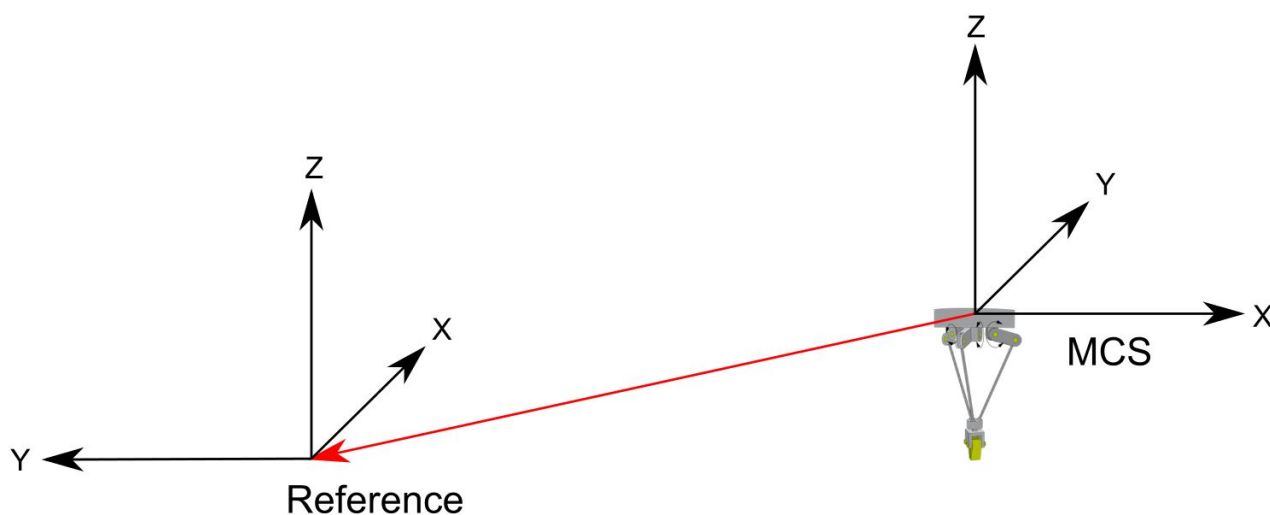
Parameter	Description	Type	Unit
Translation Y	Translation in Y direction.	LREAL	mm
Translation Z	Translation in Z direction.	LREAL	mm
Rotation 1	First rotation angle. The interpretation is defined by the parameter Rotation Convention.	LREAL	°
Rotation 2	Second rotation angle. The interpretation is defined by the parameter Rotation Convention.	LREAL	°
Rotation 3	Third rotation angle. The interpretation is defined by the parameter Rotation Convention.	LREAL	°
Rotation convention	The rotation convention indicates the order of the axis rotations (parameter Rotation 1-3). The letters (X, Y, Z) from left to right indicate the order of the rotation around the corresponding axes. The number indicates the parameter (Rotation 1-3) for the value parameterization. The translation is always performed before the rotation.	MC.CoordInterpretation_SO3	
Spatial reference	The Spatial reference parameter indicates which coordinate system is used as basis for the MCS. If the value 0 is set here, the WCS is used as the basis. To use another coordinate system as starting point for the translation, a <u>Coordinate Frame</u> [► 61] object can be created. The object ID of this coordinate system can be selected via the drop-down menu.	OTCID	
Definition direction	Defines the direction in which the translation is programmed (from the point of view of the reference system or from the point of view of the MCS), see example below.	MC.ReferenceDefDir	

#### Example: Definition Direction

If the Definition Direction MCS -> Reference is used, the translation from the source coordinate system (MCS) to the target coordinate system (Reference) shown below is specified with negative vectors.

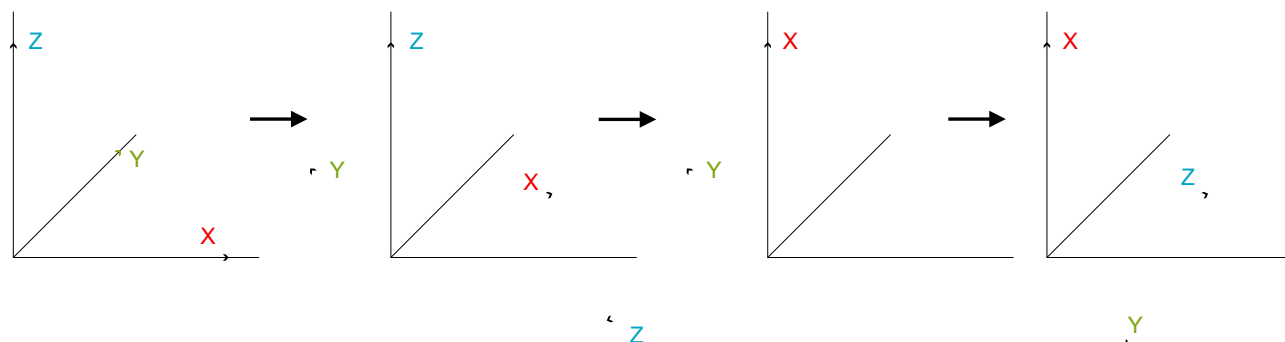


If a positive rotation around the Z-axis (here 90°) is specified in addition to the translation, the translation is carried out first and then the target coordinate system is rotated (here +90° around the Z-axis).



### Example: Rotation

Parameter	Example values
Rotation 1	180°
Rotation 2	90°
Rotation 3	45°
Rotation convention	Rotation_Z3Y2_X1_DIN9300



Initial situation

First rotation around the Z axis with the degree of ".Rotation 3" (45°)

Second rotation around the Y axis with the degree of ".Rotation 2" (90°)

Third rotation around the X axis with the degree of ".Rotation 1" (180°)

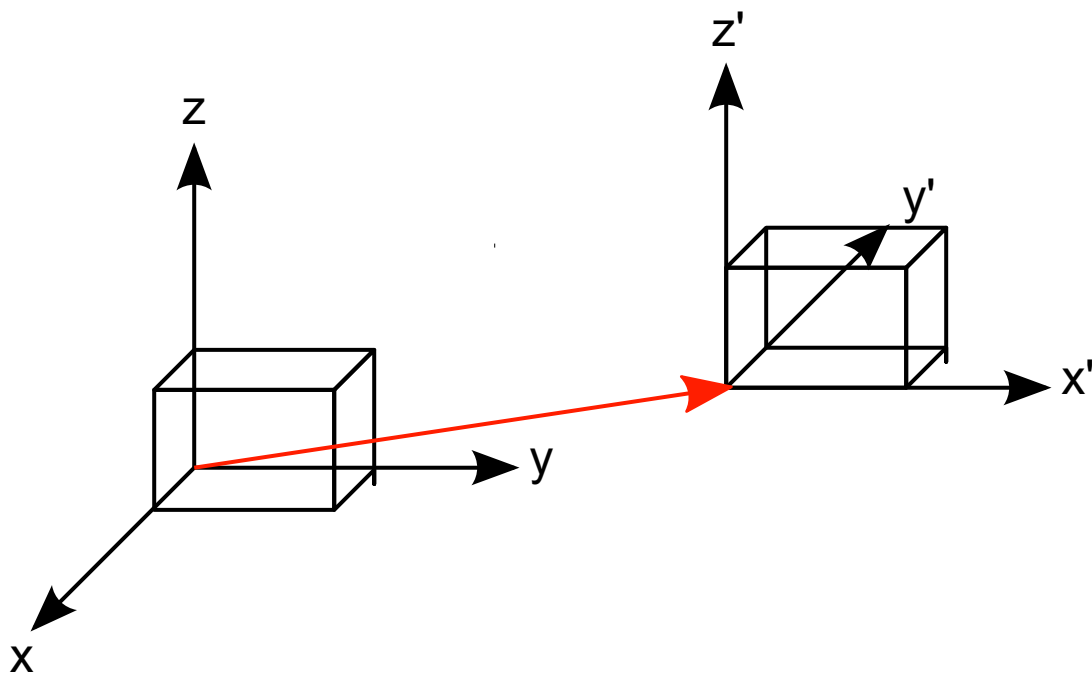
### Tool offset OID

Parameter	Description	Type
Tool offset OID	To define a tool for the kinematics a <a href="#">Tool Offset</a> [► 59] object or a <a href="#">Tool Linear</a> [► 60] object has to be created at first. The object ID of this tool can be selected via the drop-down menu.	OTCID

## 6.2 Static Transformation

The static transformation enables creation of a cartesian gantry. It supports translation and rotation between two cartesian coordinate systems.

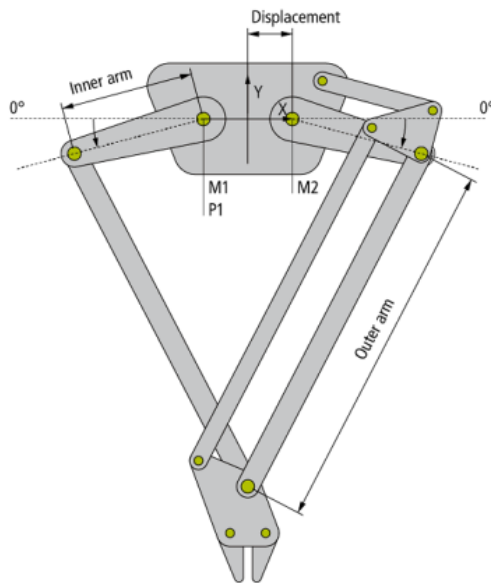
The static transformation forms the kinematic, so that the kinematic group is formed with the static transformation. In contrast, a [Coordinate system \(Coordinate Frame\)](#) [► 61] adds a translation and a rotation to an existing kinematic.



First the translation is calculated, then the rotation. The order of the rotations affects the orientation of the coordinate system. The roll/pitch/yaw rule described in DIN 9300 is used as default for the rotation sequence. The calculation sequence for the forward transformation is Z, Y', X''.

Parameter	Description	Type	Unit
Translation X	Shift in X-direction	LREAL	mm
Translation Y	Shift in Y-direction	LREAL	mm
Translation Z	Shift in Z-direction	LREAL	mm
Rotation 1	First rotation angle. The interpretation is defined by the parameter Rotation convention.	LREAL	°
Rotation 2	Second rotation angle. The interpretation is defined by the parameter Rotation convention.	LREAL	°
Rotation 3	Third rotation angle. The interpretation is defined by the parameter Rotation convention.	LREAL	°
Rotation convention	The rotation convention indicates the order of the axis rotations (parameter Rotation 1-3). The letters (X, Y, Z) from left to right indicate the order of the rotation around the corresponding axes. The number indicates the parameter (Rotation 1-3) for the value parameterization. The translatory shift is always performed before the rotation.	MC.CoordInterpretation_SO3	
Spatial reference	The parameter Spatial reference indicates which coordinate system is used as basis for this coordinate system. If the value is set to 0, the WCS is used as basis. To use another coordinate system as starting point for the shift, a further <a href="#">Coordinate system (Coordinate Frame)</a> [► 61] object can be created. The object ID of this coordinate system can be selected via the dropdown menu.	OTCID	
Definition direction	Indicates the direction in which the shift is programmed (from the perspective of the reference system or this coordinate system).	MC.ReferenceDefDir	

## 6.3 2D-Kinematics Type 1 (P\_2C)



The 2D-Kinematics Type 1 (P\_2C) is configured as shown in the diagram above.

All motor axes are scaled in degrees; 0° is defined as shown in the diagram. The arrow indicates the positive direction.

### Parameters for the Kinematics

Parameter	Description	Type	Unit
Inner arm length	Length between pivots of the inner arm	LREAL	mm
Outer arm length	Length between pivots of the outer arm	LREAL	mm
Displacement	Length between the center of the base plate and the virtual rotation axes of the inner arm	LREAL	mm

### Parameters for the Dynamic Model

Parameter	Description	Type	Unit
Inner arm mass	Total mass of the inner arm	LREAL	kg
Inner arm moment of inertia	Moment of inertia of the inner arm in relation to pivot point P1, which is linked with the motor	LREAL	kg mm <sup>2</sup>
Outer arm mass	The mass of the external arms minus the mass of the joint can optionally be described as a separate parameter.	LREAL	kg
First link mass	Mass of the joint linking the inner and outer arm; can be used if the mass of the joint is not already included in the outer and inner arms. The mass of the joint linking the gripper plate with the outer arm is not specified here. It can be added to TcpMass. The mass of the first joint refers to the inner arm that is linked to motor 1.	LREAL	kg
Second link mass	See FirstLinkMass The mass of the second joint refers to the inner arm that is linked to motor 2.	LREAL	kg
TCP mass	Mass of the tool center point, including gripper plate and gripper. The payload is usually described with a separate parameter.	LREAL	kg

Parameter	Description	Type	Unit
First drive torque OID	Object ID of the first drive torque (see <a href="#">Drive Torque</a> [► 58]) If the motors and gear units of all motors behave in a similar way, all drive torques can be represented via an OID. Both parameters therefore refer to the same object ID.	OTCID	
Second drive torque OID	Object ID of the second drive torque	OTCID	

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

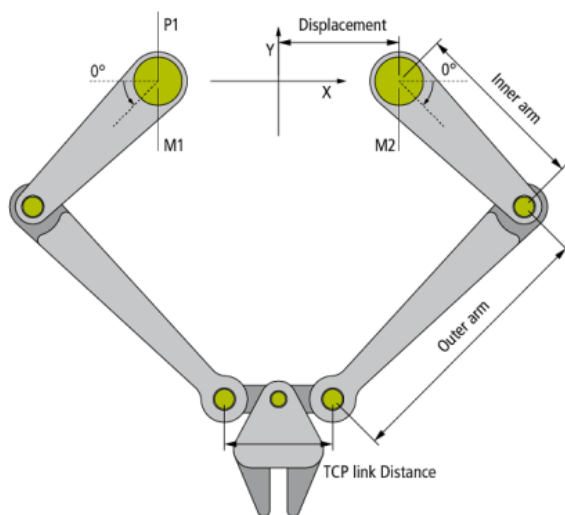
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.4 2D-Kinematics Type 2 (P\_2C2)



The 2D-Kinematics Type 2 (P\_2C2) is configured as shown in the diagram above.

All motor axes are scaled in degrees; 0° is defined as shown in the diagram. The arrow indicates the positive direction.

### Parameters for the Kinematics

Parameter	Description	Type	Unit
Inner arm length	Length between pivots of the inner arm	LREAL	mm
Outer arm length	Length between pivots of the outer arm	LREAL	mm

Parameter	Description	Type	Unit
Displacement	Length between the center of the base plate and the virtual rotation axes of the inner arm	LREAL	mm
TCP link distance	Distance between pivots of the outer arm	LREAL	mm

### Parameters for the Dynamic Model

Parameter	Description	Type	Unit
Inner arm mass	Total mass of the inner arm	LREAL	kg
Inner arm moment of inertia	Moment of inertia of the inner arm in relation to pivot point P1, which is linked with the motor.	LREAL	kg mm <sup>2</sup>
Outer arm mass	The mass of the external arms minus the mass of the joint can optionally be described as a separate parameter.	LREAL	kg
First link mass	Mass of the joint linking the inner and outer arm. Can be used if the mass of the joint is not already included in the outer and inner arms. The mass of the joint linking the gripper plate with the outer arm is not specified here. It can be added to the TCP mass. The mass of the first joint refers to the inner arm that is linked to M1.	LREAL	kg
Second link mass	see First link mass The mass of the second joint refers to the inner arm that is linked to M2.	LREAL	kg
TCP mass	Mass of the TCP, including gripper plate and gripper. The payload is usually described with a separate parameter.	LREAL	kg
First drive torque OID	Object ID of the first drive torque (see <a href="#">Drive Torque</a> [► 58]) If the motors and gear units of all motors behave in a similar way, all drive torques can be represented via an OID. Both parameters therefore refer to the same object ID.	OTCID	
Second drive torque OID	Object ID of the second drive torque	OTCID	

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

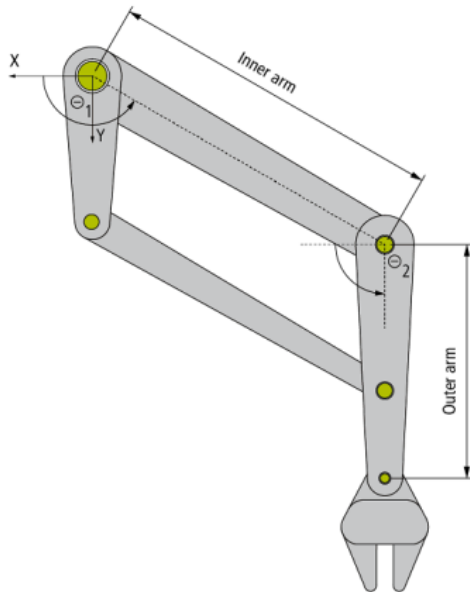
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.5 2D-Kinematics Type 3 (S\_CC)



2D-Kinematics Type 3 (S\_CC) is configured as shown in the diagram above.

All motor axes are scaled in degrees;  $0^\circ$  is defined as shown in the diagram. The arrow indicates the positive direction.

This kinematics type is implemented as a left-handed system. The shafts of motor M1 and M2 are located at the origin of the coordinate system.

### Parameters for the Kinematics

Parameter	Description	Type	Unit
Inner arm length	Length between the motor shaft and the pivot point of the external arm	LREAL	mm
Outer arm length	Length between the pivot point and the tool center point of the outer arm	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

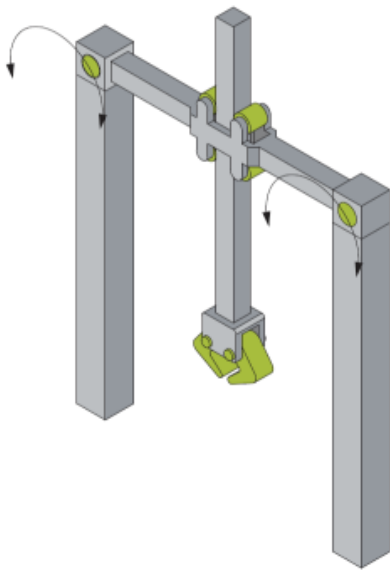
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.6 2D-Kinematics H-Bot (P\_2Y)



The 2D-Kinematics H-Bot (P\_2Y) is configured as shown in the diagram above.

The motor axes have to be scaled in millimeters. All the other position parameters result from the kinematic constraints.

The point of origin of the machine coordinate system MCS is defined by the point for that the positions of the two motors are zero.

### Parameters for the Dynamic Model

Parameter	Description	Type
FirstDriveTorqueOID	Object ID of the first drive torque (see <a href="#">Drive Torque [► 58]</a> ). If the motors and gear units of all motors behave in a similar way, all drive torques can be represented via an OID. Both parameters therefore refer to the same object ID.	OTCID
SecondDriveTorqueOID	Object ID of the second drive torque	OTCID

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

For all kinematics with tool also applies:

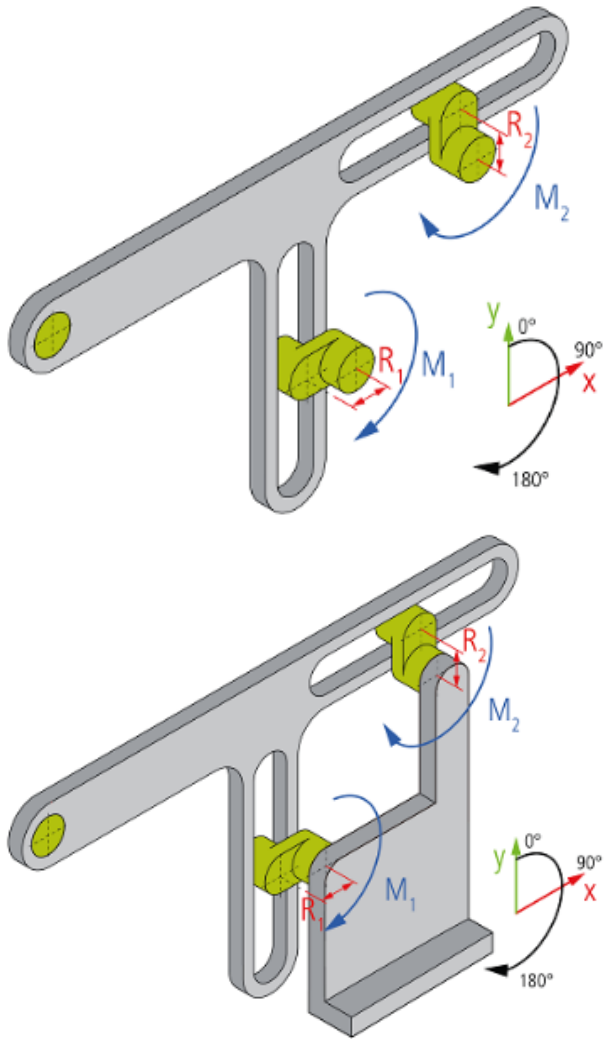
- [Tool Offset OID \[► 22\]](#).

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)



## 6.7 2D-Kinematics Type 5 (S\_CC)



A crank consists of a wheel with an eccentrically located pin. Two cranks whose ends lead to bearings facilitate two dimensional movements of the TCP. The cranks are moved by motors that are obstructed in a stationary machine.

### Parameters for the Kinematics

Parameter	Description	Type	Unit
Radius $R_1$	<ul style="list-style-type: none"> <li>The quantity <math>R_1</math> describes the lever arm of crank 1. This lever arm is measured from the crank center to the center of the pin in the bearing.</li> <li>The rotational center of crank 1 is fixed.</li> <li>Crank 1 is moved by motor M1.</li> <li>Motor M1 evokes movements in X-direction of the TCP.</li> </ul>	LREAL	mm
Radius $R_2$	<ul style="list-style-type: none"> <li>The quantity <math>R_2</math> describes the lever arm of crank 2. This lever arm is measured from the crank center to the center of the pin in the bearing.</li> <li>The rotational center of crank 2 is fixed.</li> <li>Crank 2 is moved by motor M2.</li> <li>Motor M2 evokes movements in Y-direction of the TCP.</li> </ul>	LREAL	mm

## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

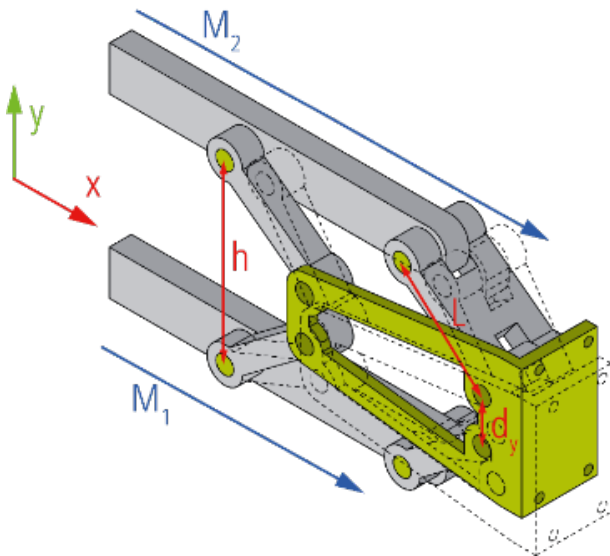
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.8 2D-Kinematics Type 6 (P\_2X)



With the Kinematics the two linear axes M1 and M2 enable movements within the XY-plane.

## Parameters for the Kinematics

Parameter	Description	Type	Unit
Flange translation X	Spatial shift in X-direction.	LREAL	mm
Arm length (L)	Length of the joint segments each measured from the motor to the joint of the flange.	LREAL	mm
Motor distance (h)	The motor distance $h$ is the Y-distance of the joints at the motors. (The motors M1 and M2 both move in X-direction. The motor distance $h$ is measured from joint center point to joint center point.)	LREAL	mm
Link distance ( $d_y$ )	The link distance $d_y$ is the distance between the joints in the flange.	LREAL	mm

## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

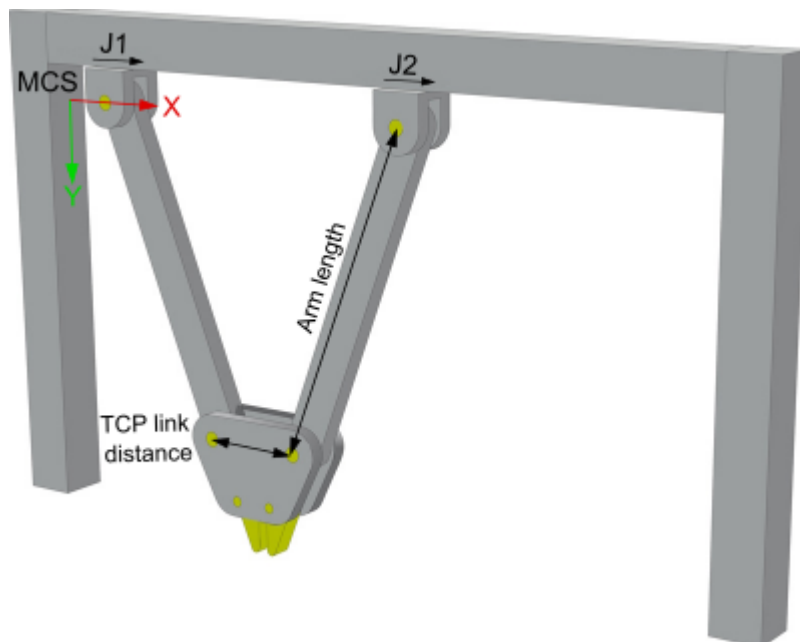
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.9 2D-Scissor Kinematics Type 1 (P\_2X)



The 2D-Scissor Kinematics Type 1 is structured as shown in the diagram above.

### Kinematics parameters

Parameter	Description	Type	Unit
Arm length	Arm length from the motor axis pivot point to the TCP pivot point	LREAL	mm
TCP link distance	Distance between the pivot points of the two arms on the TCP	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20].
- [Spatial reference definition](#) [► 20].

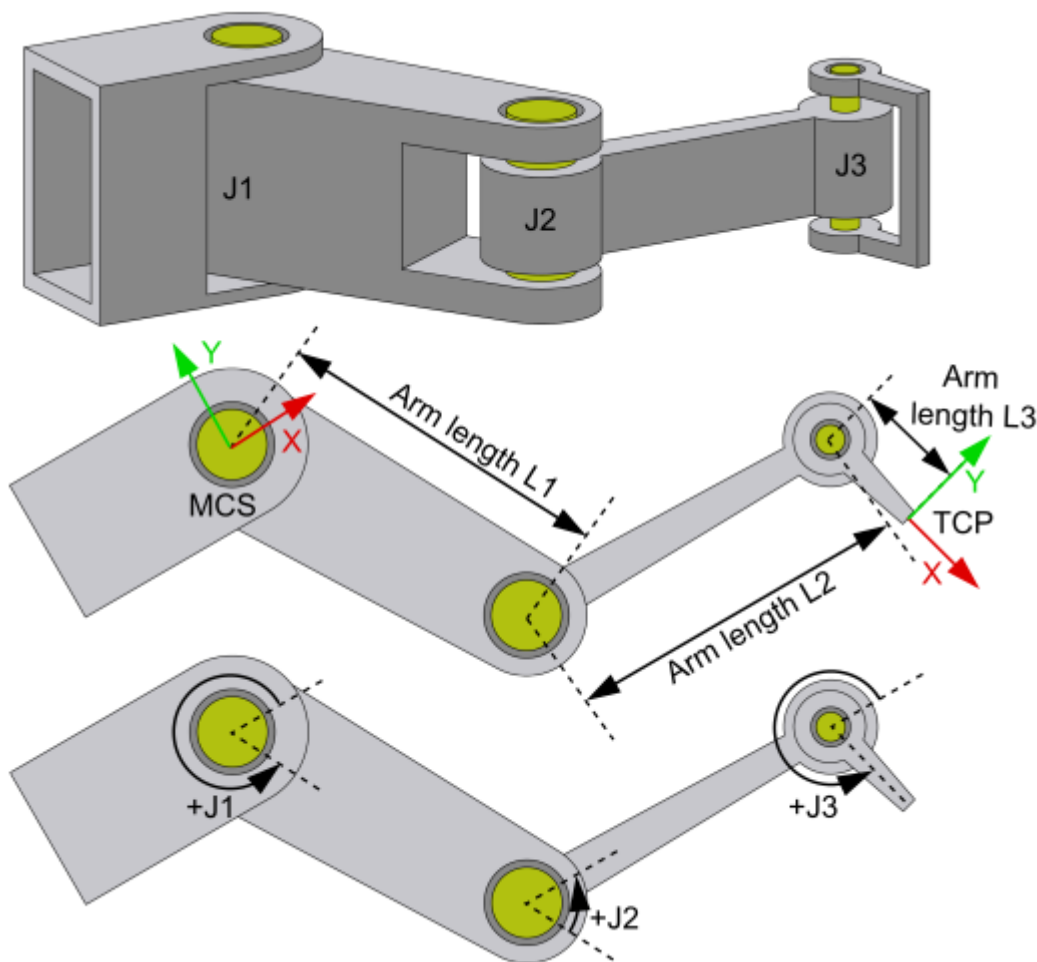
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Prerequisites

Installation package	Target platform	TwinCAT function
TF5400 TwinCAT 3 Advanced Motion Pack V3.3.25	PC or CX (x86 or x64)	TF5111 TwinCAT 3 Kinematic Transformation (Level 2)

## 6.10 3D-Kinematics Type 8 (S\_CCC)



The 3D-Kinematics Type 8 (S\_CCC) describes a serial kinematic transformation that is structured as shown in the diagram above.

All joints are scaled in degrees, with the positive direction of rotation being in the direction of the arrow. The origin of the machine coordinate system (MCS) is located in joint J1.

### ● Singular positions

**i**

Singular positions cannot be approached in Cartesian mode. It is only possible to approach these positions in axis mode (direct mode).

### Kinematics parameters

Parameter	Description	Type	Unit
Arm length L1	Distance between joints J1 and J2	LREAL	mm
Arm length L2	Distance between joints J2 and J3	LREAL	mm
Arm length L3	Distance between joint J3 and the flange	LREAL	mm

### Parameters for a gear coupling

If there are other motors between the physical position of a motor and the corresponding joint, there is a coupling between them. The respective coupling factor must be parameterized.

Parameter	Description	Type
Gear coupling 1 to 2	The parameter describes the influence of the motor M1 on the joint J2. $J2 = M2 + M1 * [\text{Gear coupling 1 to 2}]$	LREAL
Gear coupling 1 to 3	The parameter describes the influence of the motor M1 on the joint J3. $J3 = M3 + M1 * [\text{Gear coupling 1 to 3}]$	LREAL
Gear coupling 2 to 3	The parameter describes the influence of the motor M2 on the joint J3. $J3 = M3 + M2 * [\text{Gear coupling 2 to 3}]$	LREAL

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

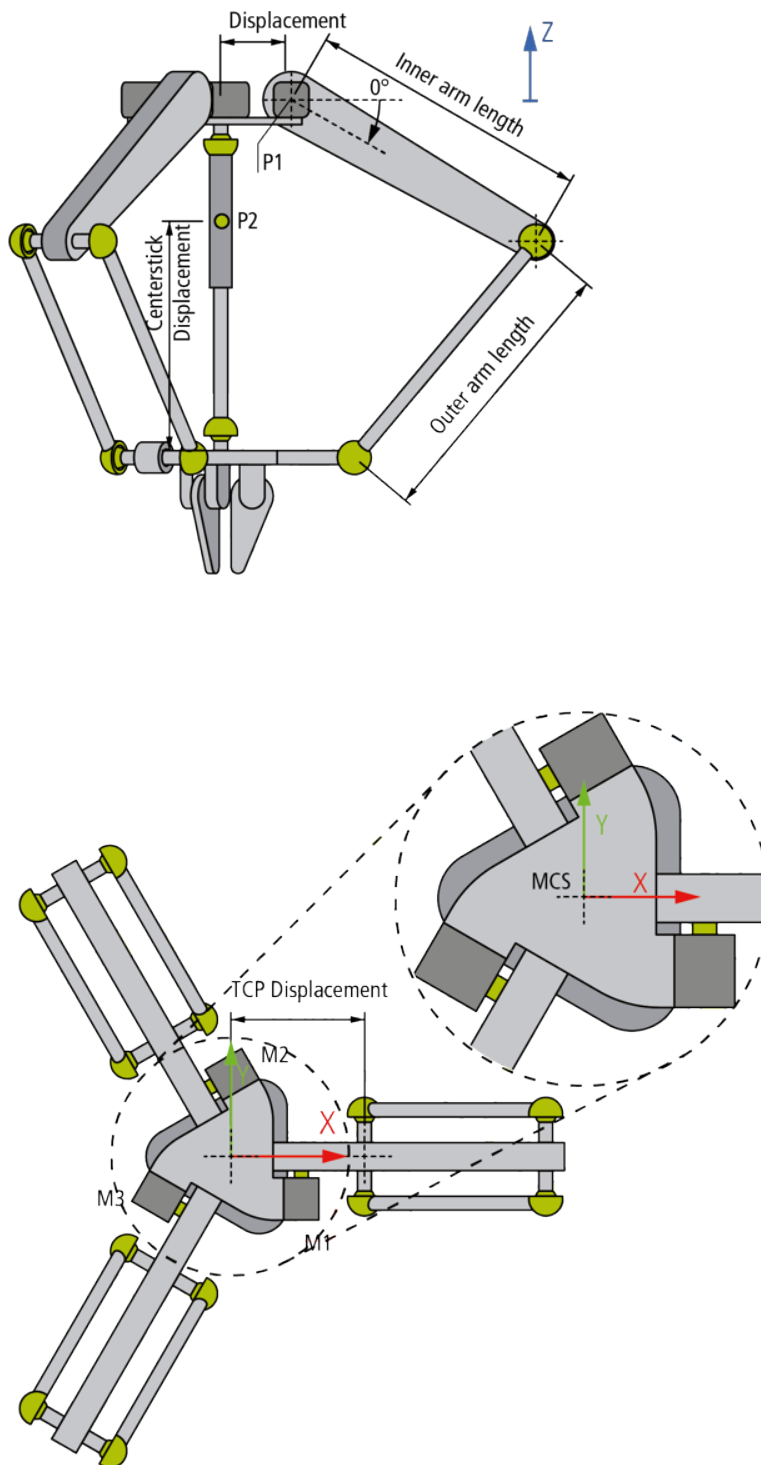
For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

### Prerequisites

Installation package	Target platform	TwinCAT function
TF5400 TwinCAT 3 Advanced Motion Pack V3.3.57	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.11 3D-Delta Type 1 (P\_3C)



The 3D-Delta Kinematics Type 1 (P\_3C) is configured as shown in the diagram above. The Kinematic Transformation expects ball joints (or elements with the same behavior) in the link between the arms and the lower plate.

Parameterization of the center stick for aligning the gripper is optional.

All motor axes are scaled in degrees; 0° is defined as shown in the diagram. The arrow indicates the positive direction. This applies for all three motors.

**Parameters for the Kinematics**

Parameter	Description	Type	Unit
Inner arm length	Length between the pivot points of the inner arm; this is the arm that is directly linked with the motor.	LREAL	mm
Outer arm length	Length between pivots of the outer arm	LREAL	mm
Displacement	Length between the center of the base plate and the virtual rotation axes of the inner arm	LREAL	mm
TCP displacement	Length between the center of the gripper plate and the virtual rotation axes of the outer arm	LREAL	mm

**Parameters for the Dynamic Model**

Parameter	Description	Type	Unit
Inner arm mass	Total mass of the inner arm	LREAL	kg
Inner arm moment of inertia	Moment of inertia of the inner arm in relation to pivot point P1, which is linked with the motor	LREAL	kg mm <sup>2</sup>
Outer arm mass	Mass of the outer arm. If two bars are used, the total mass is required. The mass of the joint can optionally be described as a separate parameter.	LREAL	kg
Link mass	Mass of the joint linking the inner and outer arm. Can be used if the mass of the joint is not already included in the outer and inner arms. The mass of the joint linking the gripper plate with the outer arm is not specified here. It can be added to TcpMass.	LREAL	kg
TCP mass	Mass of the TCP, including gripper plate and gripper. The payload is usually described with a separate parameter.	LREAL	kg
Center stick mass	Total mass of the center stick	LREAL	kg
Center stick: moment of inertia	Moment of inertia of the center stick in relation to the center of gravity (P2)	LREAL	kg mm <sup>2</sup>
Center stick: center of mass displacement	Length between the gripper plate and the center of gravity of the bar	LREAL	mm
First drive torque OID	Object ID of the first drive torque (see <a href="#">Drive Torque</a> [► 58]) If the motors and gear units of all motors behave in a similar way, all drive torques can be represented via an OID. All three parameters refer to the same object ID.	OTCID	
Second drive torque OID	Object ID of the second drive torque	OTCID	
Third drive torqueOID	Object ID of the third drive torque	OTCID	

**General Parameters for the Kinematics**

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

For all kinematics with tool also applies:

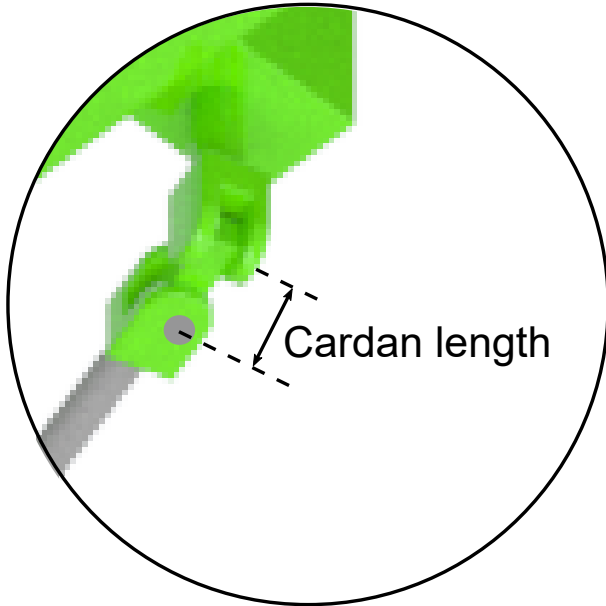
- [Tool Offset OID](#) [► 22].

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.12 3D-Delta Type 2 (P\_3C2)

The 3D-Delta Type 2(P\_3C2) essentially corresponds to the [3D-Delta Type 1 \(P\\_3C\)](#) [► 34], but no ball joints are required. Instead, the offset of cardan joints can be parameterized.



## Kinematics parameters

Parameter	Description	Type	Unit
Inner arm length	Length between the pivot points of the inner arm; this is the arm that is directly linked with the motor.	LREAL	mm
Outer arm length	Length between pivot points of the outer arm	LREAL	mm
Displacement	Length between the center of the base plate and the virtual rotation axes of the inner arm	LREAL	mm
TCP displacement	Length between the center of the gripper plate and the virtual rotation axes of the outer arm	LREAL	mm
Upper cardan length	If a cardan joint is used at the upper arm suspension points, this parameter can be used to specify the offset of the two joints within the cardan joint. When using a ball joint, enter length 0.	LREAL	mm
Lower cardan length	If a cardan joint is used at the lower arm suspension points, this parameter can be used to specify the offset of the two joints within the cardan joint. When using a ball joint, enter length 0.	LREAL	mm

## Parameters for the Dynamic Model

Parameter	Description	Type	Unit
Inner arm mass	Total mass of the inner arm	LREAL	kg



Parameter	Description	Type	Unit
Inner arm moment of inertia	Moment of inertia of the inner arm in relation to pivot point P1, which is linked with the motor	LREAL	kg mm <sup>2</sup>
Outer arm mass	Mass of the outer arm. If two bars are used, the total mass is required. The mass of the joint can optionally be described as a separate parameter.	LREAL	kg
Link mass	Mass of the joint linking the inner and outer arm. Can be used if the mass of the joint is not already included in the outer and inner arms. The mass of the joint linking the gripper plate with the outer arm is not specified here. It can be added to TcpMass.	LREAL	kg
TCP mass	Mass of the TCP, including gripper plate and gripper. The payload is usually described with a separate parameter.	LREAL	kg
Center stick mass	Total mass of the center stick	LREAL	kg
Center stick: moment of inertia	Moment of inertia of the center stick in relation to the center of gravity (P2)	LREAL	kg mm <sup>2</sup>
Center stick: center of mass displacement	Length between the gripper plate and the center of gravity of the bar	LREAL	mm
First drive torque OID	Object ID of the first drive torque (see <a href="#">Drive Torque</a> [► 58]) If the motors and gear units of all motors behave in a similar way, all drive torques can be represented via an OID. All three parameters refer to the same object ID.	OTCID	
Second drive torque OID	Object ID of the second drive torque	OTCID	
Third drive torqueOID	Object ID of the third drive torque	OTCID	

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

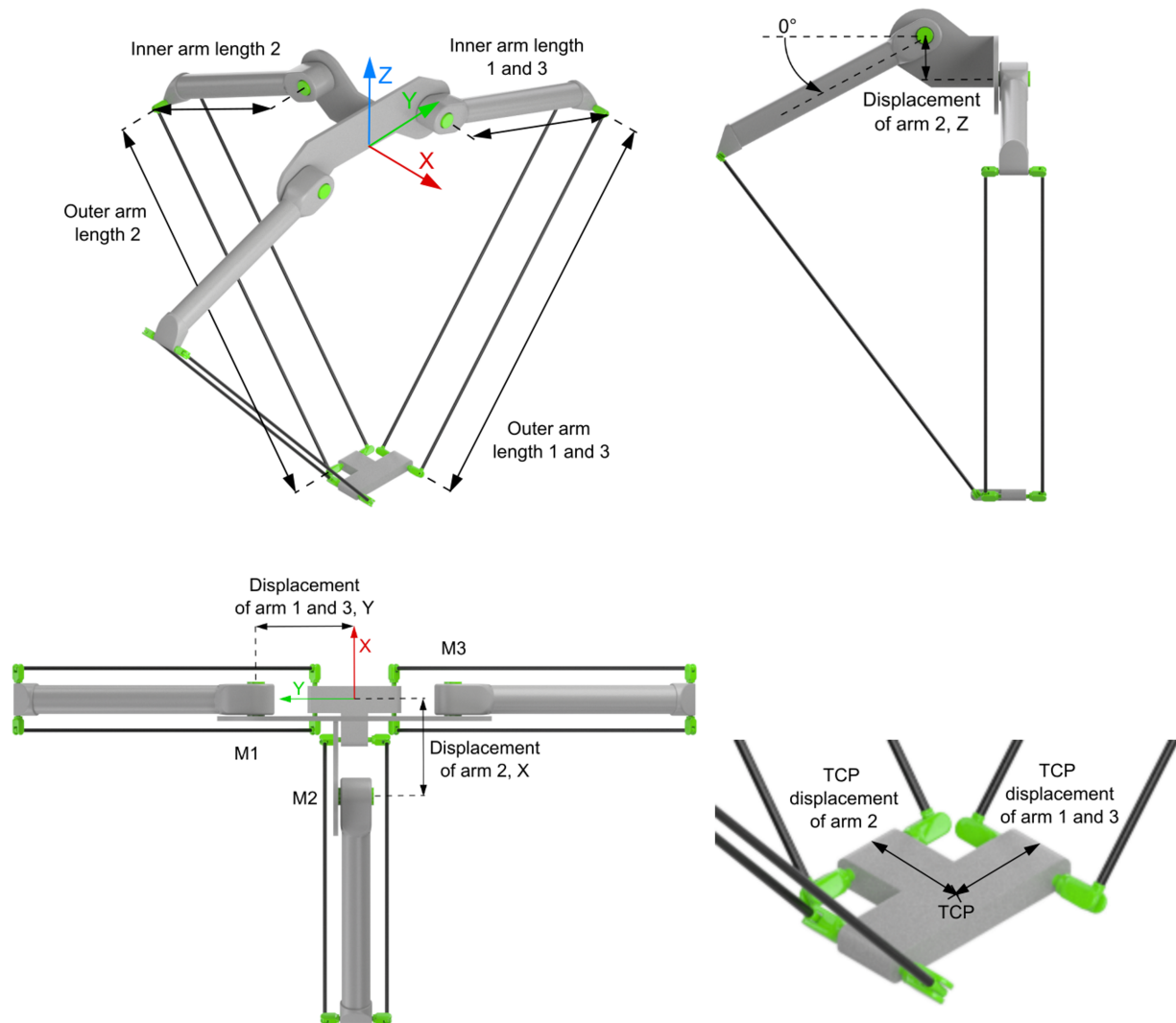
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Prerequisites

	Target platform	TwinCAT function
	PC or CX (x86 or x64)	TF5112 TC3 Kinematic Transformation (Level 3)

## 6.13 3D-Delta T Type 3 (P\_3C3)



The 3D-Delta T Type 3 (P\_3C3) is structured as shown above. Two arms are directly opposite each other, while the third arm is at an angle of 90 degrees to them. This arm configuration allows two robots of this type to be placed very close together.

The machine coordinate system (MCS) is located centrally between the two opposite arms at the height of motors M1 and M3.

All motor axes are scaled in degrees and 0° is defined, as shown in the schematic, with the arrow indicating the positive direction of rotation. This applies to all three motors.

### Kinematics parameters

Parameter	Description	Type	Unit
Inner arm length 1 and 3	Arm 1, Arm 3: Length from center of rotation to center of rotation of the inner arm (connected directly to the motor)	LREAL	mm
Inner arm length 2	Arm 2: Length from center of rotation to center of rotation of the inner arm (connected directly to the motor)	LREAL	mm
Outer arm length 1 and 3	Arm 1, Arm 3: Length between pivots of the outer arm	LREAL	mm
Outer arm length 2	Arm 2: Length between pivots of the outer arm	LREAL	mm

Parameter	Description	Type	Unit
Displacement of arm 1 and 3, Y	Arm 1, Arm 3: The distance between the MCS origin and each motor axis	LREAL	mm
Displacement of arm 2, X	Arm 2: The distance between the MCS origin and the motor axis	LREAL	mm
Displacement of arm 2, Z	Arm 2: The distance between the MCS origin and the motor axis	LREAL	mm
TCP displacement of arm 1 and 3	Arm 1, Arm 3: Length between the center of the gripper plate and the virtual rotation axes of the outer arm	LREAL	mm
TCP displacement of arm 2	Arm 2: Length between the center of the gripper plate and the virtual rotation axis of the outer arm	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

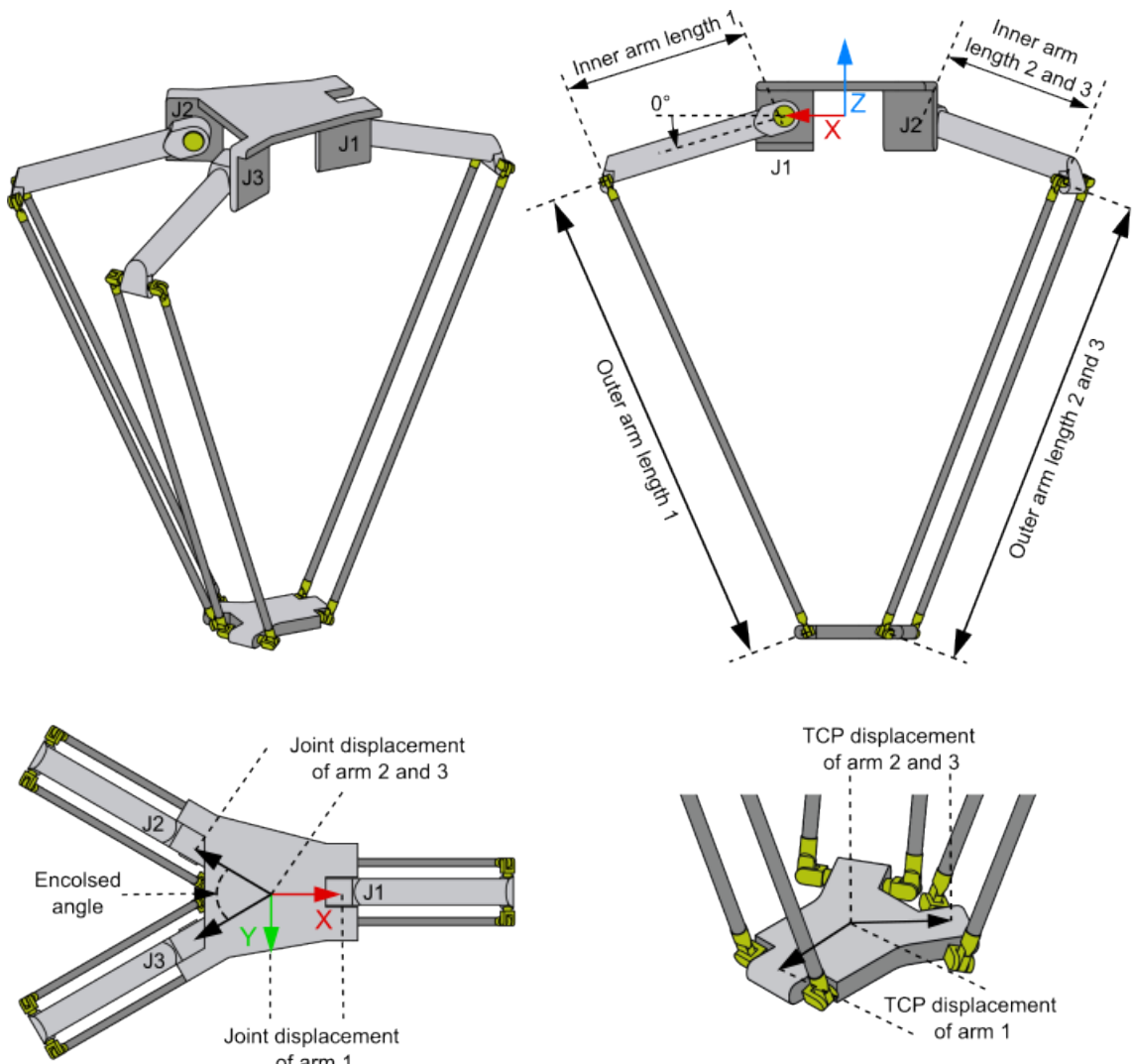
For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4024.7 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.14 3D-Delta Y Type 4 (P\_3C4)



The 3D-Delta Y Type 4 (P\_3C4) is structured as shown in the diagram above.

All three arms are suspended at the same level. Unlike 3D-Delta Type 1 (P\_3C) [P\_34], the angle between arms 2 and 3 can be specified.

The machine coordinate system (MCS) is located in the middle between the arms at the height of the motors.

All motor axes are scaled in degrees and  $0^\circ$  is defined, as shown in the diagram, with the arrow indicating the positive direction of rotation. This applies to all three motors.

### Kinematics parameters

Parameter	Description	Type	Unit
Inner arm length 1	Arm 1: Length from pivot point to pivot point of the inner arm (directly connected to the motor)	LREAL	mm
Outer arm length 1	Arm 1: Length from pivot point to pivot point of the outer arm	LREAL	mm
Joint displacement of arm 1	Arm 1: The distance from the MCS origin to the motor axis	LREAL	mm
TCP displacement of arm 1	Arm 1: Length between the center of the gripper plate and the virtual rotation axes of the outer arm	LREAL	mm

Parameter	Description	Type	Unit
Enclosed angle	Angle between arms 2 and 3	LREAL	
Inner arm length 2 and 3	Arm 2, arm 3: Length from pivot point to pivot point of the inner arm (directly connected to the motor)	LREAL	mm
Outer arm length 2 and 3	Arm 2, arm 3: Length from pivot point to pivot point of the outer arm	LREAL	mm
Joint displacement of arm 2 and 3	Arm 2, arm 3: The distance from the MCS origin to the motor axis	LREAL	mm
TCP displacement of arm 2 and 3	Arm 2, arm 3: Length from the center of the gripper plate to the virtual axes of rotation of the outer arm	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

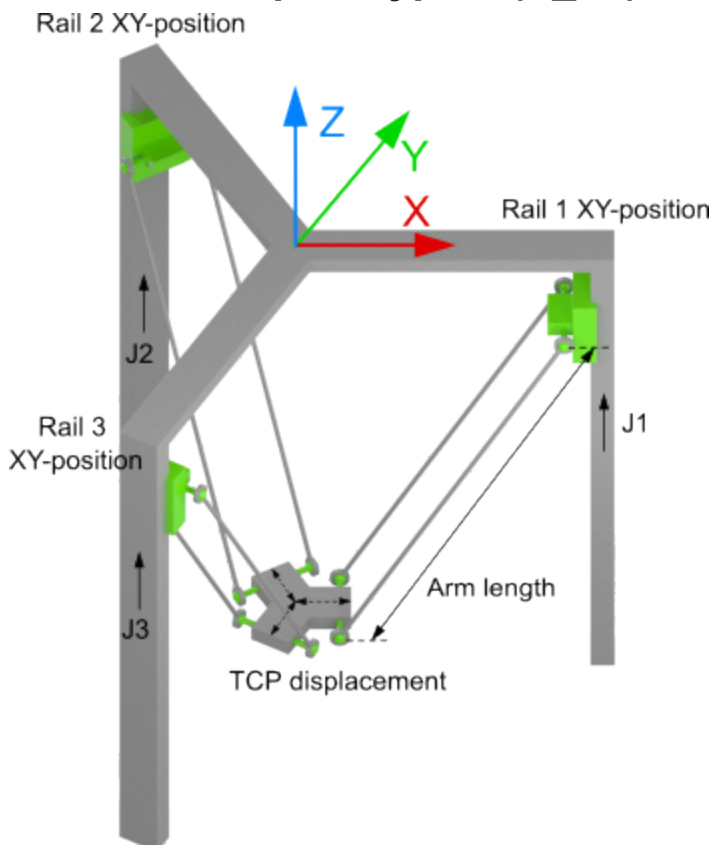
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Prerequisites

Installation package	Target platform	TwinCAT function
TF5400 TwinCAT 3 Advanced Motion Pack V3.3.57	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.15 3D-Tripod Type 1 (P\_3Z)



The 3D-Tripod Type 1 (P\_3Z) is structured as shown in the figure above.

All linear axes (ACS) are scaled in mm.

### Kinematics parameters

Parameter	Description	Type	Unit
Arm length	Arm length from pivot point to pivot point	LREAL	mm
Rail 1 X-position	X-position of the 1st rail in relation to the MCS	LREAL	mm
Rail 1 Y-position	Y-position of the 1st rail in relation to the MCS	LREAL	mm
Rail 2 X-position	X-position of the 2nd rail in relation to the MCS	LREAL	mm
Rail 2 Y-position	Y-position of the 2nd rail in relation to the MCS	LREAL	mm
Rail 3 X-position	X-position of the 3rd rail in relation to the MCS	LREAL	mm
Rail 3 Y-position	Y-position of the 3rd rail in relation to the MCS	LREAL	mm
TCP displacement	Length between the center of the gripper plate and the virtual rotation axis of the arm	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

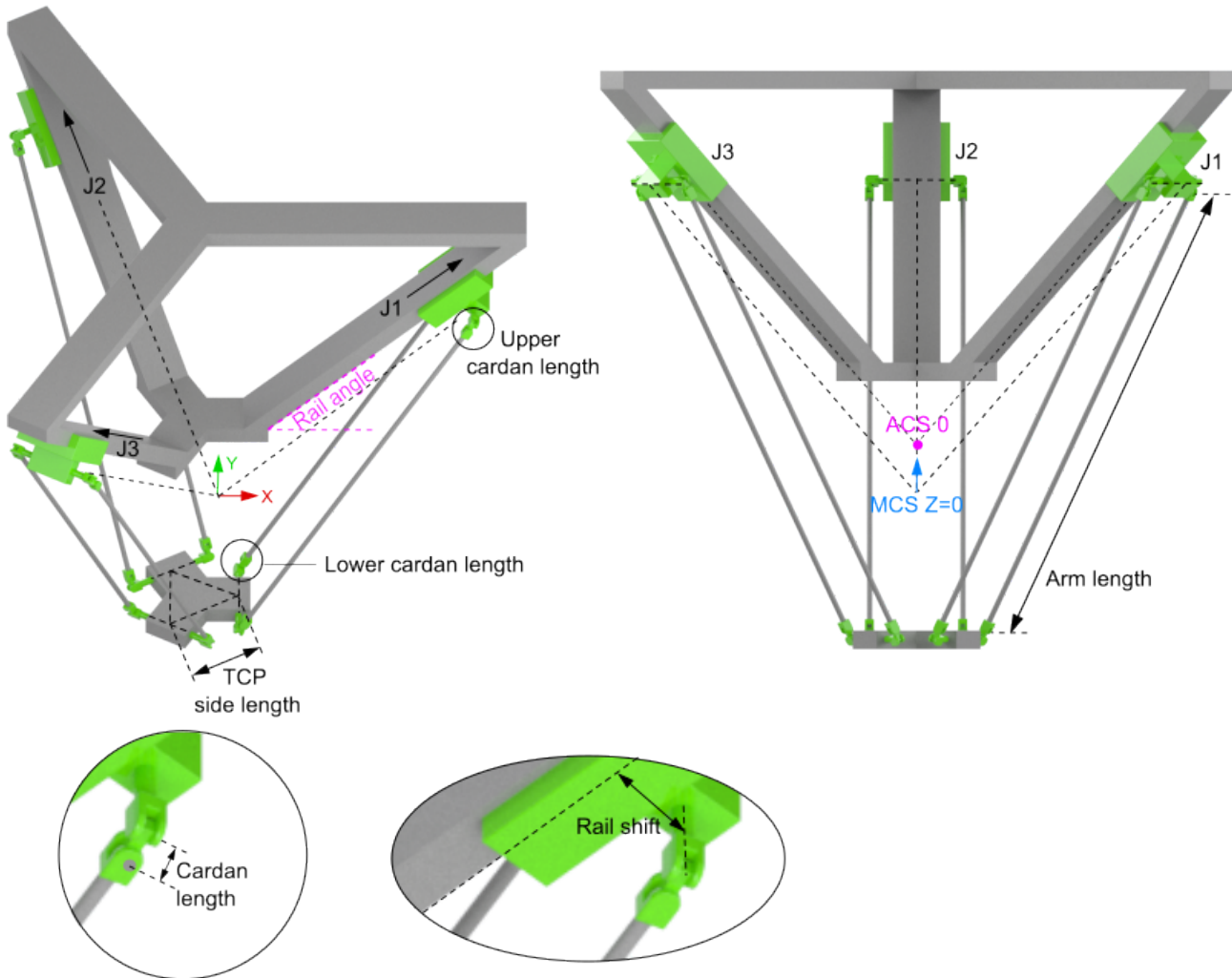
For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

### Requirements

Development environment Installation package	Target platform	TwinCAT function
TwinCAT V3.1.4024.24 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.66	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.16 3D-Tripod Type 2 (P\_3L)



The 3D-Tripod Type 2 (P\_3L) is structured as shown in the figure above.

All linear axes (ACS) are scaled in millimeters (mm). The 0-position of the axes is only a "virtual" point, which cannot be approached. A positive velocity of the motors moves the tool upwards so that the linear axes cannot reach a negative position.

### Kinematics parameters

Parameter	Description	Type	Unit
Arm length	Arm length from pivot point to pivot point	LREAL	mm
Rail angle	Angle in which the guide rails of the linear motors are mounted.	LREAL	°
Rail shift	Shift of the arm suspension points to the guide rails of the linear motors.	LREAL	mm
Upper cardan length	If a cardan joint is used at the upper arm suspension points, this parameter can be used to specify the shift of the two joints within the cardan joint. When using a ball joint, enter length 0.	LREAL	mm
Lower cardan length	If a cardan joint is used at the lower arm suspension points, this parameter can be used to specify the shift of the two joints within the cardan joint. When using a ball joint, enter length 0.	LREAL	mm
TCP side length	Side length of the virtual triangle in the TCP.	LREAL	mm

## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20].
- [Spatial reference definition](#) [► 20].

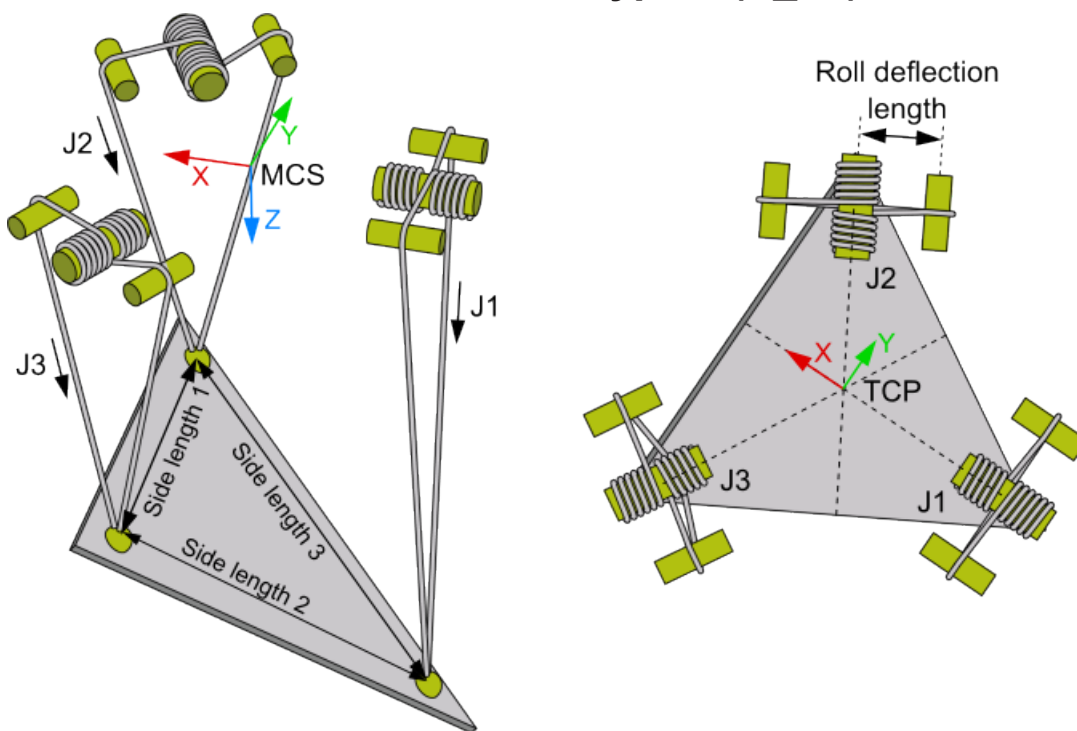
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## Requirements

Development environment Installation package	Target platform	TwinCAT function
TwinCAT V3.1.4024.24 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.66	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.17 3D-Cable Kinematics Type 1 (P\_3Z)



The 3D-Cable Kinematics Type 1 (P\_3Z) is structured as shown in the diagram above.

- The zero point of the machine coordinate system (MCS) is centered between the three cable suspension points, with the Z-axis pointing downwards.
- All motor axes are scaled in mm, the arrow indicating the positive direction.

## Kinematics parameters

Parameter	Description	Type	Unit
Side length L1	Distance between suspension points 2 and 3	LREAL	mm
Side length L2	Distance between suspension points 1 and 3	LREAL	mm
Side length L3	Distance between suspension points 1 and 2	LREAL	mm
Roll deflection length	Distance between the motor shafts and deflection rollers	LREAL	mm



## General Parameters for the Kinematics

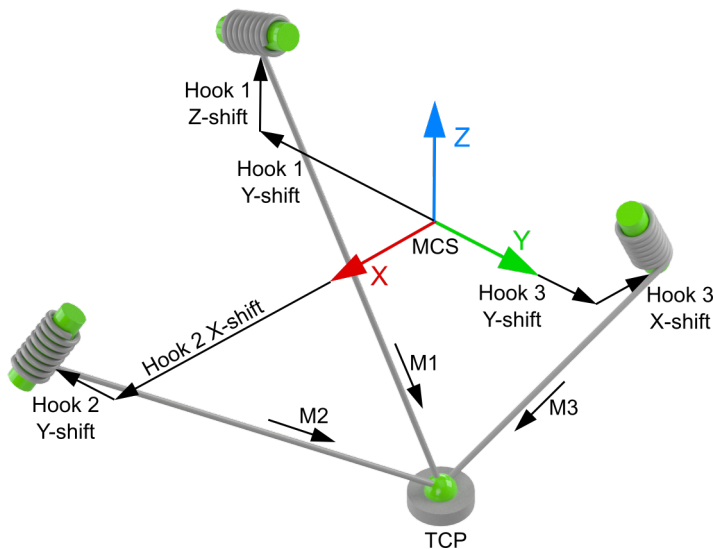
General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## 6.18 3D-Cable Kinematics Type 2 (P\_3L)



The 3D-Cable Kinematics (P\_3L) is structured as shown in the above schematic.

The zero point of the machine coordinate system (MCS) can be anywhere in space. The hooks of the cable/rope are defined starting from the MCS origin.

All motor axes are scaled in millimeters, the arrow indicating the positive direction.

### Parameters for joint hooks

Parameter	Description	Type	Unit
<b>Hook 1</b>	Hook of the first cable		
X-shift	X-position of Hook 1 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 1 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 1 in relation to the MCS	LREAL	mm
<b>Hook 2</b>	Hook of the second cable		
X-shift	X-position of Hook 2 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 2 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 2 in relation to the MCS	LREAL	mm
<b>Hook 3</b>	Hook of the central cable		
X-shift	X-position of Hook 3 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 3 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 3 in relation to the MCS	LREAL	mm

## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],

- [Spatial reference definition](#) [► 20].

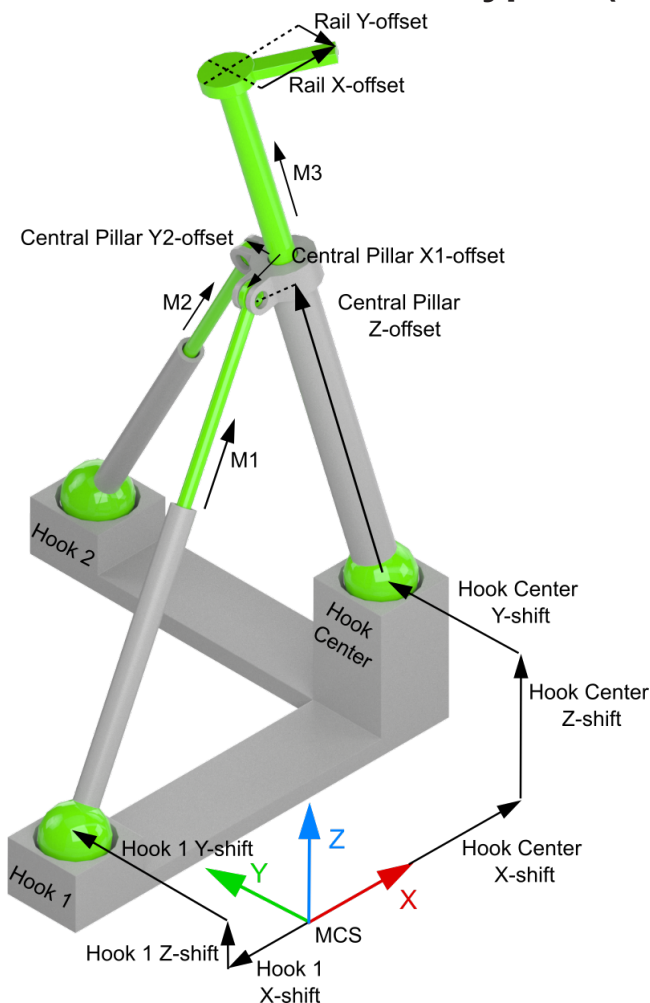
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4024.7 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.19 3D-Kinematics Type 7 (PXX SZ)



The 3D-Kinematics Type 7 (PXX SZ) is structured as shown in the schematic above.

The zero point of the machine coordinate system (MCS) can be anywhere in space. The hooks of the three arms are defined starting from the MCS origin. For the definition of the offset, the starting point for the central pillar is the position parallel to the Z-axis of the MCS, so that the fixed length from the central pillar to the suspension of the other two arms is the "central pillar Z-offset". At the end of the central pillar there may also be a rail, whose tip is specified by the "rail offset".

All motor axes are scaled in millimeters, the arrow indicating the positive direction.

### Parameters Joint Hooks

Parameter	Description	Type	Unit
<b>Hook 1</b>	Hook of the first arm		
X-shift	X-position of Hook 1 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 1 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 1 in relation to the MCS	LREAL	mm
<b>Hook 2</b>	Hook of the second arm		
X-shift	X-position of Hook 2 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 2 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 2 in relation to the MCS	LREAL	mm
<b>Hook Center</b>	Hook of the central pillar		
X-shift	X-position of Hook Center in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook Center in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook Center in relation to the MCS	LREAL	mm

### Parameters Central Pillar

The parameters for the central pillar including the rail are specified in relation to the arm position parallel to the Z-axis.

Parameter	Description	Type	Unit
X1 offset	Distance between the hook of the first arm and the central pillar in the X-direction	LREAL	mm
X2 offset	Distance between the hook of the second arm and the central pillar in the X-direction	LREAL	mm
Y1 offset	Distance between the hook of the first arm and the central pillar in the Y-direction	LREAL	mm
Y2 offset	Distance between the hook of the second arm and the central pillar in the Y-direction	LREAL	mm
Z offset	Distance between the hook of the central pillar and the connection of the arms to the central pillar	LREAL	mm
<b>Rail offset</b>			
X rail offset	Distance between the central point of the tip of the central pillar and the tip of the rail in the X-direction	LREAL	mm
Y rail offset	Distance between the central point of the tip of the central pillar and the tip of the rail in the Y-direction	LREAL	mm
Z rail offset	Distance between the central point of the tip of the central pillar and the tip of the rail in the Z-direction	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

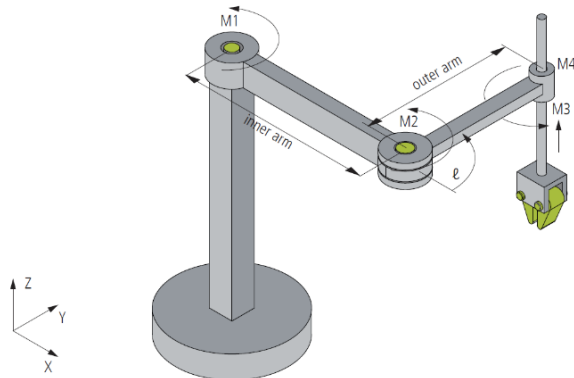
For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4024.7 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.20 4D-SCARA (S\_CCZC)



The 4D-SCARA (**S**elective **C**ompliance **A**ssembly **R**obot **A**rm) Kinematics (S\_CCZC) are configured as shown in the diagram above.

The motor axes M1, M2 and M4 are scaled in degrees whereat the arrow indicates the positive direction of rotation. The third motor axis M3 is scaled in millimeters.

The origin of the MCS is located in the first joint (M1). The X-axis is determined by the SCARA arm, when all rotational motor axes reside at 0°.



The extension position of the SCARA arm (all rotary motor axes at position 0°) cannot be reached in cartesian mode because the robot is in a singular position there. It is only possible to drive to these positions in axis mode (Direct Mode).

## Parameters for the Kinematics

Parameter	Description	Type	Unit
Inner arm length	Length from pivot point to pivot point of the inner arm; this is the arm on the origin side.	LREAL	mm
Outer arm length	Length from pivot point to pivot point of the outer arm; this is the arm on the TCP side.	LREAL	mm
Gear coupling	Coupling factor between the axes M4 and M3.	LREAL	mm/°
Flange rotation A	Rotation angle around the local X-axis.	LREAL	°
Tool offset OID	Object ID of a tool mounted on the kinematics flange. The flange coordinate system is rotated 180° around the X-axis so that its Z-axis points downwards.	OTCID	

## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

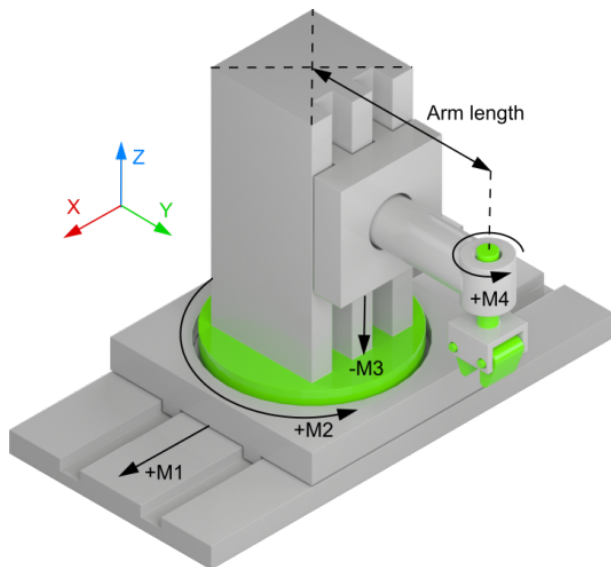
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.21 4D-Kinematics Type 6 (S\_XCZC)



The 4D-Kinematics Type 6 (S\_XCZC) describes a serial kinematic transformation that is structured as shown in schematic above.

The motor axes M2 and M4 are scaled in degrees, the positive direction of rotation being in the direction of the arrow.

The motor axes M1 and M3 are scaled in millimeters. In relation to the MCS, M1 specifies a movement on the X-axis and M3 a movement on the Z-axis. The origin of the MCS coordinate system is located on the linear axis M1 in joint M2.



### Moving to singular positions

Singular positions, as with this robot type, e.g.  $M2 = +90^\circ$ , cannot be moved to in cartesian mode. It is only possible to drive to these positions in axis mode (Direct Mode).

### Kinematics parameters

Parameter	Description	Type	Unit
Arm length	Distance between rotary axis M2 and rotary axis M4 Arm length > 0	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

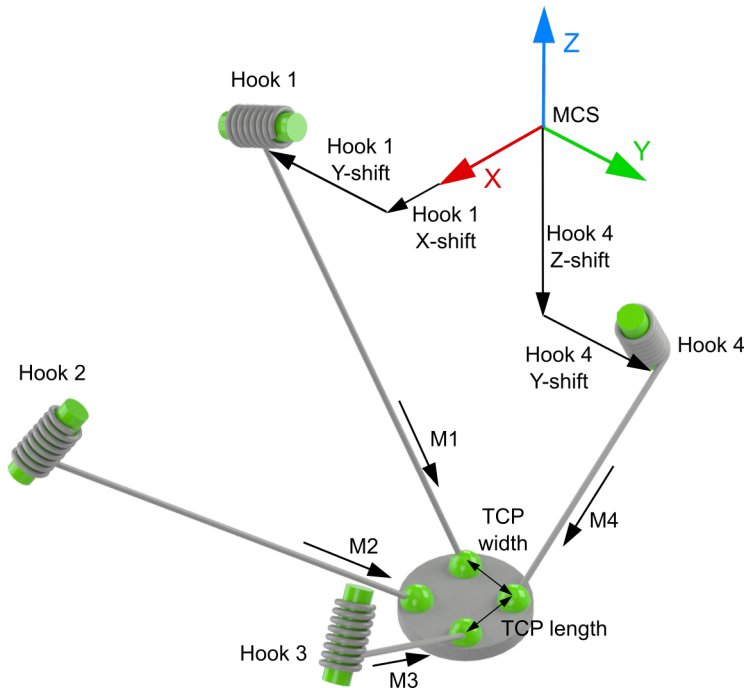
For all kinematics with tool also applies:

- [Tool Offset OID](#) [► 22].

## Requirements

Development Environment	Target System	TwinCAT Function
<b>Installation Package</b> TwinCAT V3.1.4024.7 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.22 4D-Cable-Kinematics (P\_4L)



The 4D-Cable Kinematics (P\_4L) is structured as shown in the above schematic.

The zero point of the machine coordinate system (MCS) can be anywhere in space. The hooks of the cable/rope are defined starting from the MCS origin.

All motor axes are scaled in millimeters, the arrow indicating the positive direction.

### Parameters for joint hooks

Parameter	Description	Type	Unit
<b>Hook 1</b>	Hook of the first cable		
X-shift	X-position of Hook 1 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 1 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 1 in relation to the MCS	LREAL	mm
<b>Hook 2</b>	Hook of the second cable		
X-shift	X-position of Hook 2 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 2 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 2 in relation to the MCS	LREAL	mm
<b>Hook 3</b>	Hook of the third cable		
X-shift	X-position of Hook 3 in relation to the MCS	LREAL	mm
Y-shift	Y-position of Hook 3 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 3 in relation to the MCS	LREAL	mm
<b>Hook 4</b>	Hook of the fourth cable		
X-shift	X-position of Hook 4 in relation to the MCS	LREAL	mm

Parameter	Description	Type	Unit
Y-shift	Y-position of Hook 4 in relation to the MCS	LREAL	mm
Z-shift	Z-position of Hook 4 in relation to the MCS	LREAL	mm
<b>TCP</b>	Tool Center Point		
TCP length	Distance between the hooks at the TCP along the X-axis	LREAL	mm
TCP width	Distance between the hooks at the TCP along the Y-axis	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

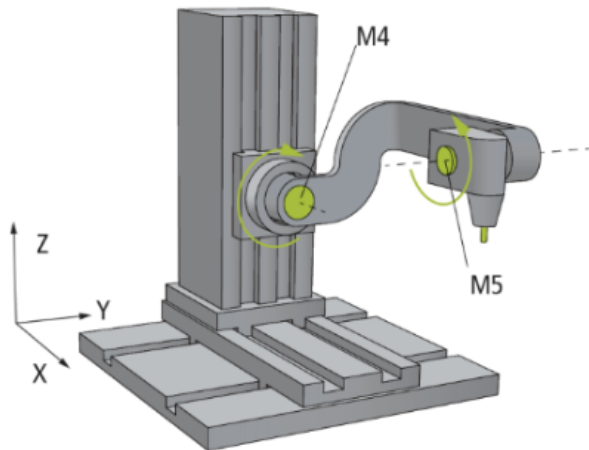
For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4024.7 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.10.30	PC or CX (x86 or x64)	TF5112 TwinCAT 3 Kinematic Transformation (Level 3)

## 6.23 5D-Kinematics Type 2 (XYZab)



The 5D-Kinematics Type 2 (XYZab) are configured as shown in the drawing above.

The motors M1 to M3 (X, Y, Z) are scaled in millimeters. The motors M4 and M5 are scaled in degrees. The 0° position is the axis position shown in the drawing; the arrows indicate the positive direction of rotation.

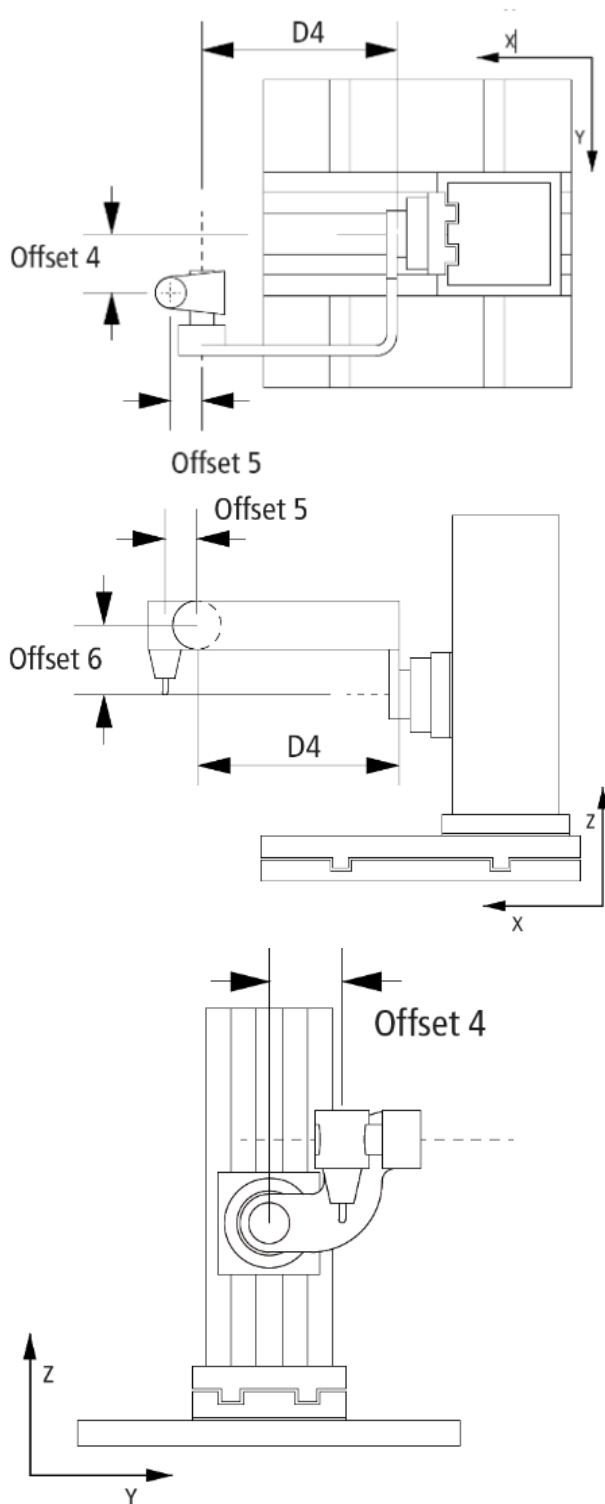


### Difference of Type 2

The 5D-Kinematics Type 2 differ from the 5D-Kinematics Type 3 in the orientation of the positive direction of axis rotation around the motor axes M4 and M5.

## Parameters for the Kinematics

Parameter	Description	Type	Unit
Handle D4	Arm length in X-direction between motor axis 4 and motor axis 5 as shown in the drawing.	LREAL	mm
Offset 4	Offset in y-direction between motor axis 4 and TCP.	LREAL	mm
Offset 5	Offset in X-direction between motor axis 5 and TCP.	LREAL	mm
Offset 6	Offset in Z-direction between motor axis 4 and motor axis 5.	LREAL	mm
Tool offset OID	Object ID of a tool mounted on the kinematics flange. The flange coordinate system is rotated 180° around the X-axis so that its Z-axis points downwards.	OTCID	





## General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

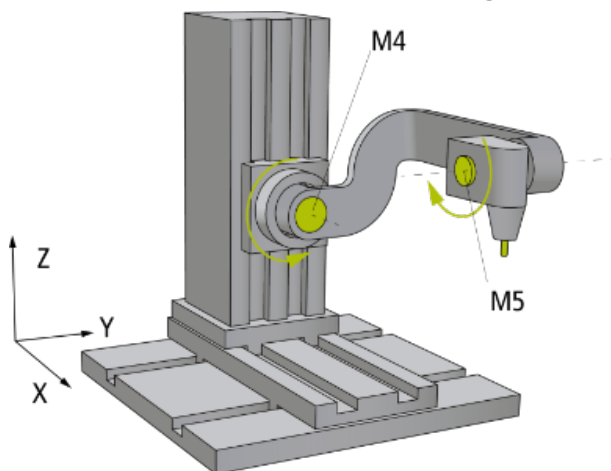
## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5113 TwinCAT 3 Kinematic Transformation (Level 4)

### TF5113 | TwinCAT Kinematic Transformation L4

**i** TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload or included in the TF5400 TwinCAT Advanced Motion Pack Setup from the website. If required, please get in touch with your sales contact.

## 6.24 5D-Kinematics Type 3 (XYZAB)



The 5D-Kinematics Type 3 (XYZAB) are configured as shown in the drawing above.

The motors M1 to M3 (X, Y, Z) are scaled in millimeters. The motors M4 and M5 are scaled in degrees. The 0° position is the axis position shown in the drawing; the arrows indicate the positive direction of rotation.

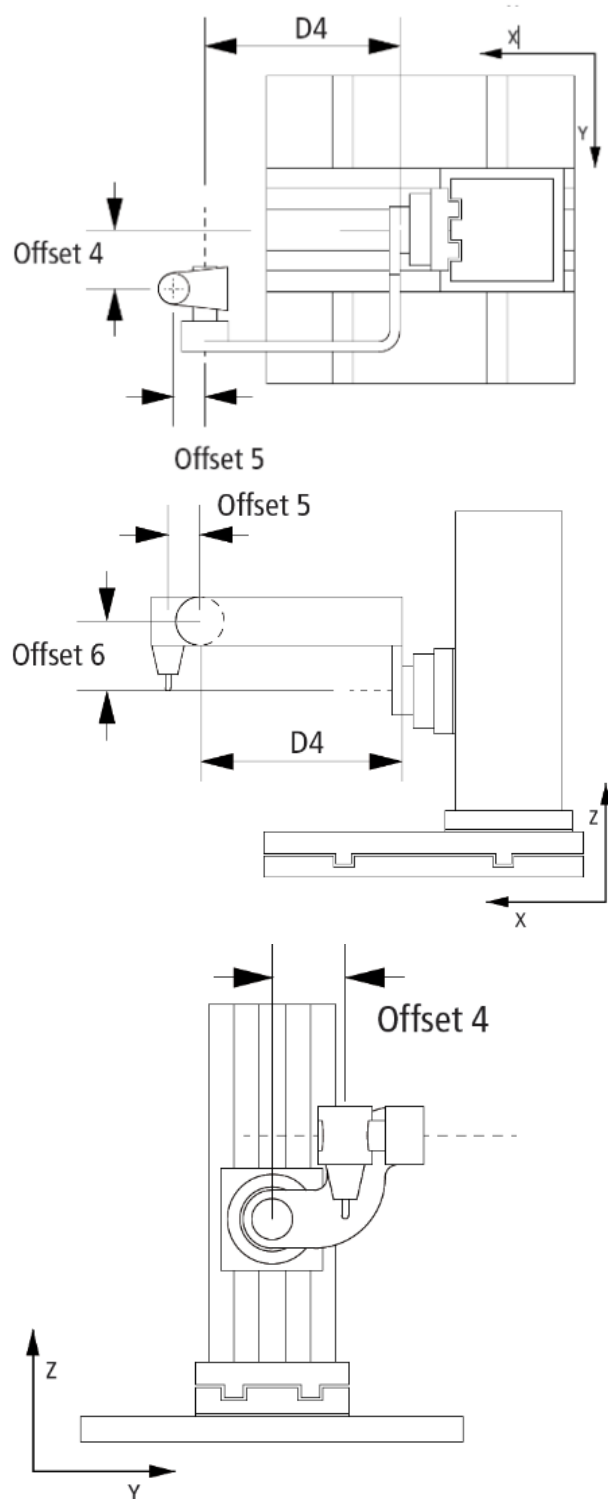
### Difference of Type 3

**i** The 5D-Kinematics Type 3 differ from the 5D-Kinematics Type 2 in the orientation of the positive direction of axis rotation around the motor axes M4 and M5.

## Parameters for the Kinematics

Parameter	Description	Type	Unit
Handle D4	Arm length in X-direction between motor axis 4 and motor axis 5 as shown in the drawing.	LREAL	mm
Offset 4	Offset in y-direction between motor axis 4 and TCP.	LREAL	mm
Offset 5	Offset in X-direction between motor axis 5 and TCP.	LREAL	mm
Offset 6	Offset in Z-direction between motor axis 4 and motor axis 5.	LREAL	mm

Parameter	Description	Type	Unit
Tool offset OID	Object ID of a tool mounted on the kinematics flange. The flange coordinate system is rotated 180° around the X-axis so that its Z-axis points downwards.	OTCID	



### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset](#) [► 20],
- [Spatial reference definition](#) [► 20].

For all kinematics with tool also applies:

- Tool Offset OID [► 22].

## Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4018.26 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.14	PC or CX (x86 or x64)	TF5113 TwinCAT 3 Kinematic Transformation (Level 4)

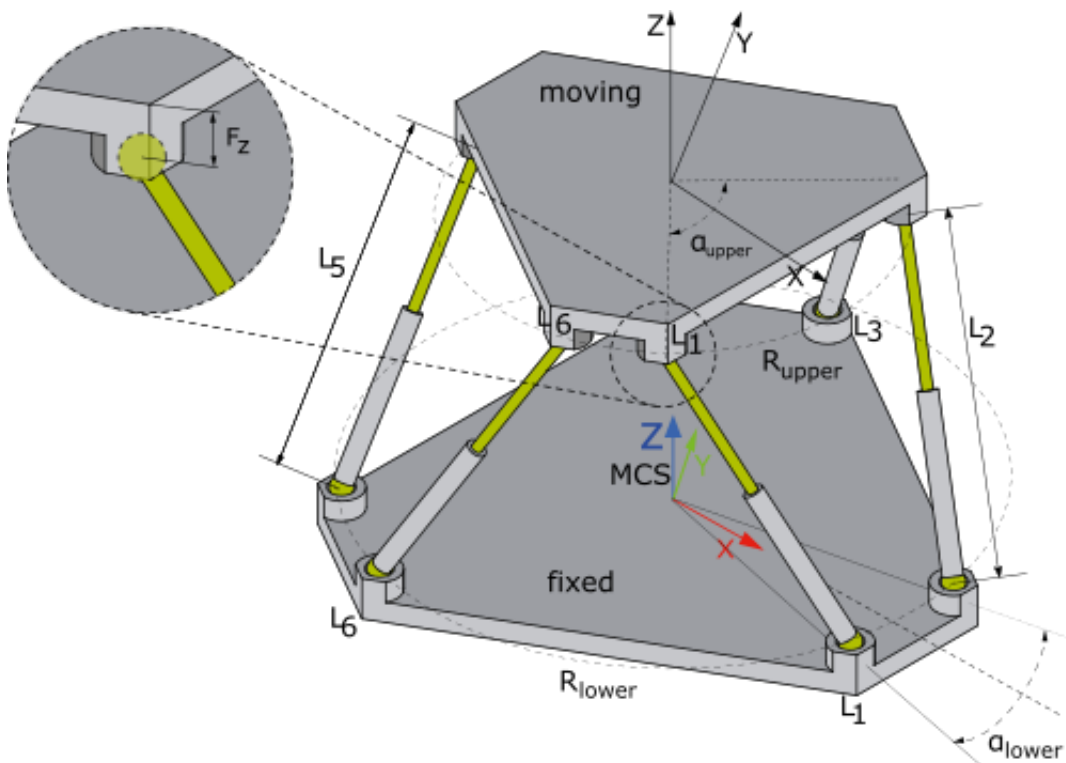


### TF5113 | TwinCAT Kinematic Transformation L4

TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload or included in the TF5400 TwinCAT Advanced Motion Pack Setup from the website. If required, please get in touch with your sales contact.

## 6.25 6D-Stewart Platform (P\_6L)

For the kinematic transformation 6D-Stewart Platform (P\_6L), a moving platform is supported by six cylinders. The Stewart Platform is a parallel kinematics with six degrees of freedom.



The X axis of the machine coordinate system (MCS) points toward the point that is in the center between joints 1 and 2. The zero point in the Z direction is usually slightly above the lower (base) platform and in the plane that is spanned by the joint centers of the anchor points.

All joints on a platform are on a circular path and thus have the same distance to the center of the platform. This distance must be specified with  $R_{lower}$  (lower platform) and  $R_{upper}$  (upper platform).

The angle between joints 1 and 2, which is equally between joints 3 and 4 and between joints 5 and 6, is specified with  $\alpha_{lower}$  (lower platform) and  $\alpha_{upper}$  (upper platform).

The ACS axis positions always refer to the entire cylinder length L. Startup with an ACS axis position equal to 0 is therefore not possible.

### Kinematics parameters

The following parameters are available for Stewart kinematics:

Parameter	Description	Type	Unit
Flange translation Z	Moves the upper level to the surface of the upper platform so that the thickness of the platform is taken into account.	LREAL	mm
Tool offset OID	Set the tool elongation.	OTCID	
Lower radius $R_{\text{lower}}$	Radius of the lower platform. Describes the distance from the origin in the center of the lower platform to the arm joints of the lower platform.	LREAL	mm
Upper radius $R_{\text{upper}}$	Radius of the upper platform. Describes the distance from the origin in the center of the upper platform to the arm joints of the upper platform.	LREAL	mm
Lower angle $\alpha_{\text{lower}}$	The angle $\alpha_{\text{lower}}$ describes the angle between the anchor points of L1 and L2, L3 and L4, L5 and L6 on the lower platform.	LREAL	°
Upper angle $\alpha_{\text{upper}}$	The angle $\alpha_{\text{upper}}$ describes the angle between the anchor points of L1 and L2, L3 and L4, L5 and L6 on the upper platform.	LREAL	°

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

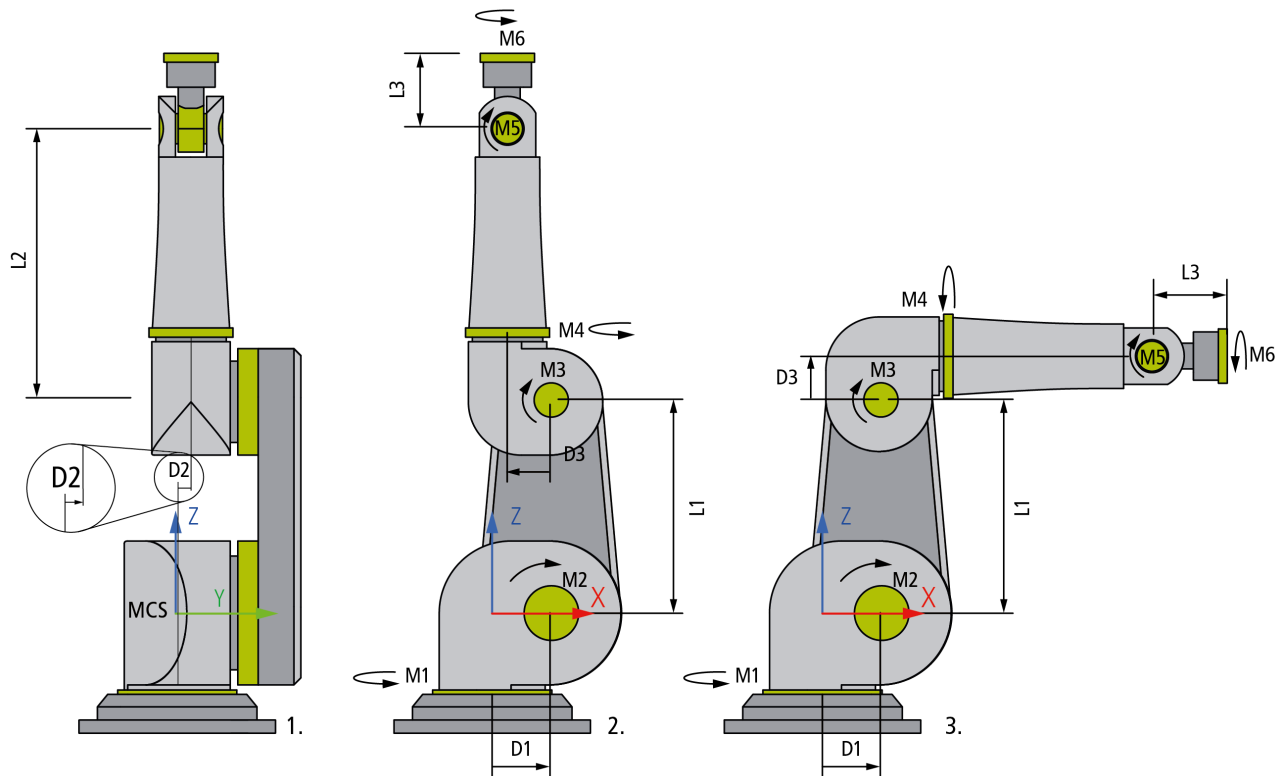
### Requirements

Development Environment Installation Package	Target System	TwinCAT Function
TwinCAT V3.1.4024.11 TF5400 TwinCAT 3 Advanced Motion Pack V3.1.6.67	PC or CX (x86 or x64)	TF5113 TwinCAT 3 Kinematic Transformation (Level 4)

#### ● TF5113 | TwinCAT Kinematic Transformation L4

**i** TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload or included in the TF5400 TwinCAT Advanced Motion Pack Setup from the website. If required, please get in touch with your sales contact.

## 6.26 Six Axis Articulated (S\_CBBCBC)



The motor axes of the Serial Six Axis Articulated (S\_CBBCBC) Kinematics each are referred to in units of degree. The drawings 1. and 2. above show the kinematics with all axes in zero position. The zero positions of the axes M4 and M6 are defined in a way that the machine coordinate system and the flange coordinate system exhibit the same orientation. The drawing 3. shows the axis M3 in 90° position.

The MCS origin is located within the intersection of the first kinematic joint M1 with the second kinematic joint M2. It is oriented in a way that joint M2 prescribes a rotation around the Y-axis. The center of M1 delivers the X zero coordinate. The intersection of M1 and M2 delivers the Y zero coordinate. The center of M2 delivers the Z zero coordinate.

### **i** Singular Positions

The positions displayed in figures 1., 2. and 3. cannot be approached in cartesian Mode because the robot resides in a singular position. Approaching these positions is only possible in axis mode (Direct Mode).

### Configuration Parameters

Parameter	Description	Type	Unit
MCS offset	Static offset to the MCS coordinate frame.		
X-shift	Static X-offset in MCS coordinate frame.	LREAL	mm
Y-shift	Static Y-offset in MCS coordinate frame.	LREAL	mm
Z-shift	Static Z-offset in MCS coordinate frame.	LREAL	mm
Spatial reference definition	Specification of the spatial reference node.		
.Translation X	Translation in X-direction.	LREAL	mm
.Translation Y	Translation in Y-direction.	LREAL	mm
.Translation Z	Translation in Z-direction.	LREAL	mm
.Rotation 1	Rotation angle 1, its interpretation is set by the rotation convention.	LREAL	°
.Rotation 2	Rotation angle 2, its interpretation is set by the rotation convention.	LREAL	°

Parameter	Description	Type	Unit
.Rotation 3	Rotation angle 3, its interpretation is set by the rotation convention.	LREAL	°
.Rotation convention	Set the interpretation of the rotation angles.	MC.CoordInterpretation_SO3	
.Spatial reference	Set the spatial frame of reference, 0x0 refers to WCS.	OTCID	
.Definition direction	Set the definition direction.	MC.ReferenceDefDir	

### Parameters for the Kinematics

For the Six Axis Articulated Kinematics, a serial six axis kinematic, there are the following joint parameters.

Parameter	Description	Type	Unit
Arm length L1	Distance between the motor axes M2 and M3.	LREAL	mm
Arm length L2	Distance between the motor axes M3 and M5.	LREAL	mm
Arm length L3	Distance between the motor axis M5 and the flange.	LREAL	mm
Arm offset D1	Distance between the motor axes M1 and M2 in X-direction.	LREAL	mm
Arm offset D2	Distance in Y-direction between the motor axes M1 and M4.	LREAL	mm
Arm offset D3	Distance in X-direction between the motor axes M3 and M5. The sign within the example sketch is positive.	LREAL	mm

### General Parameters for the Kinematics

General parameters that apply to any kinematics are described in the following sections:

- [MCS Offset \[► 20\]](#),
- [Spatial reference definition \[► 20\]](#).

For all kinematics with tool also applies:

- [Tool Offset OID \[► 22\]](#).

## **i** TF5113 | TwinCAT Kinematic Transformation L4

TF5113 | TwinCAT 3 Kinematic Transformation L4 is subject to legal restrictions and is not included in the TF5400.AdvancedMotionPack workload or included in the TF5400 TwinCAT Advanced Motion Pack Setup from the website. If required, please get in touch with your sales contact.

## 6.27 Drive Torque

The drive torque represents the inertia and the efficiency of the motor and gear unit. It is used for the precise computation of the dynamic model.

A parameter in the kinematics can be used to assign an object drive torque to a kinematics.

### Parameter for drive

Parameter	Description	Unit
Drive moment of inertia	Rotor Moment of inertia of the motor	kg mm <sup>2</sup>

## Gear unit parameters

Parameter	Description	Unit
Ratio	Gear ratio	
Gearbox moment of inertia	Moment of inertia of the gear unit in relation to the drive	kg mm <sup>2</sup>
Coulomb friction	Represents the kinetic friction coefficient	Nm
Stokes friction	Represents the friction ratio that increases proportionally to the speed	Nms

### Required product level:

Level 1

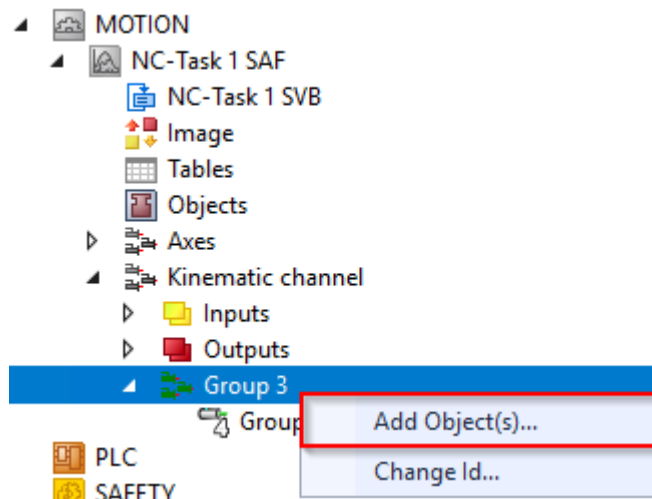
## 6.28 Tool Offset

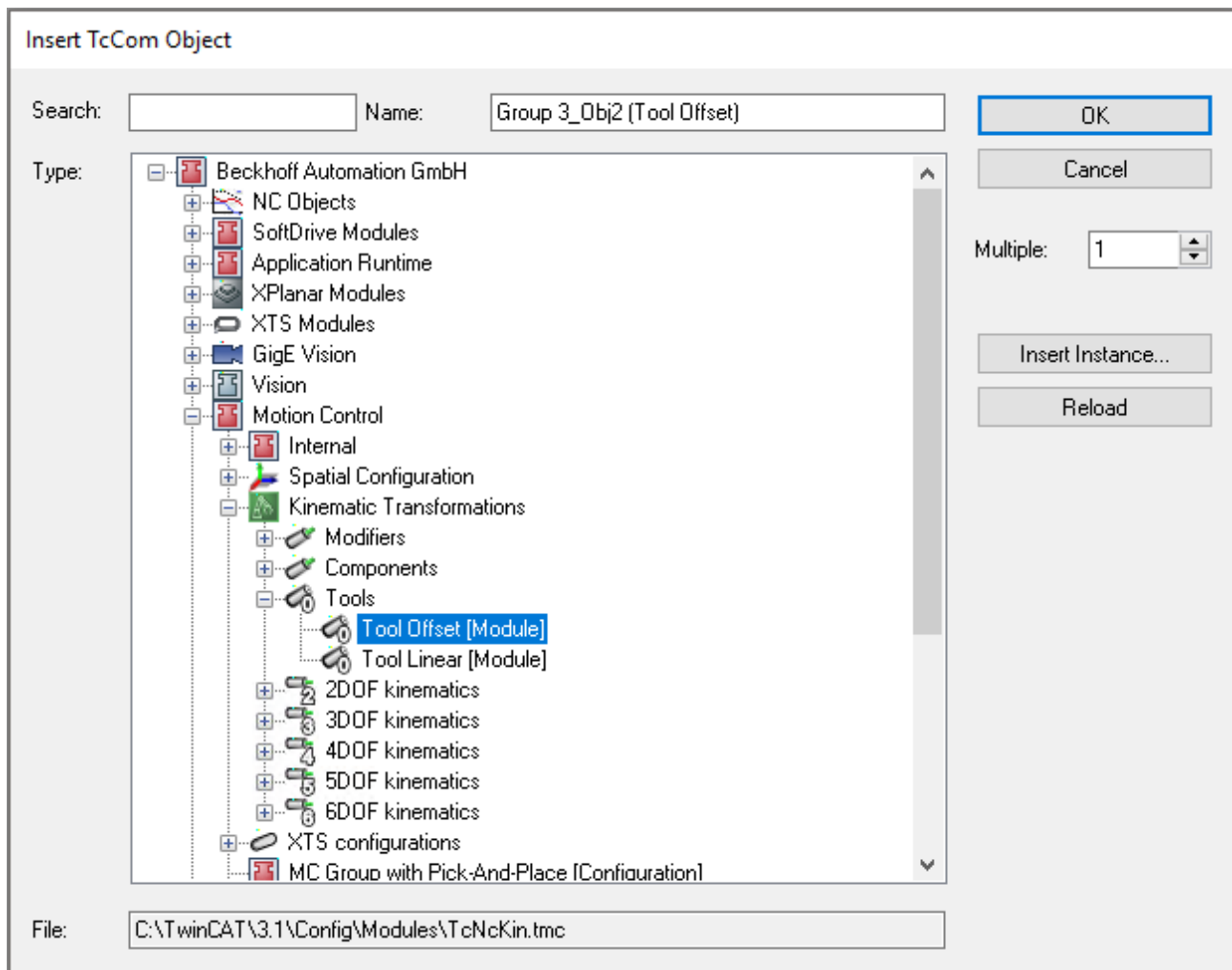
The user can use the tool offset to link a tool with the flange of the kinematics. Unless specified otherwise in the kinematics, the flange coordinate system is defined such that the orientation of the flange coordinate system matches that of the machine coordinate system MCS, if all axes are at 0.

Parameter	Description	Unit
Extension X	X-offset of the static tool, which is mounted at the coordinate system of the flange of the higher-level transformation	mm
Extension Y	Y-offset of the static tool, which is mounted at the coordinate system of the flange of the higher-level transformation	mm
Extension Z	Z-offset of the static tool, which is mounted at the coordinate system of the flange of the higher-level transformation	mm

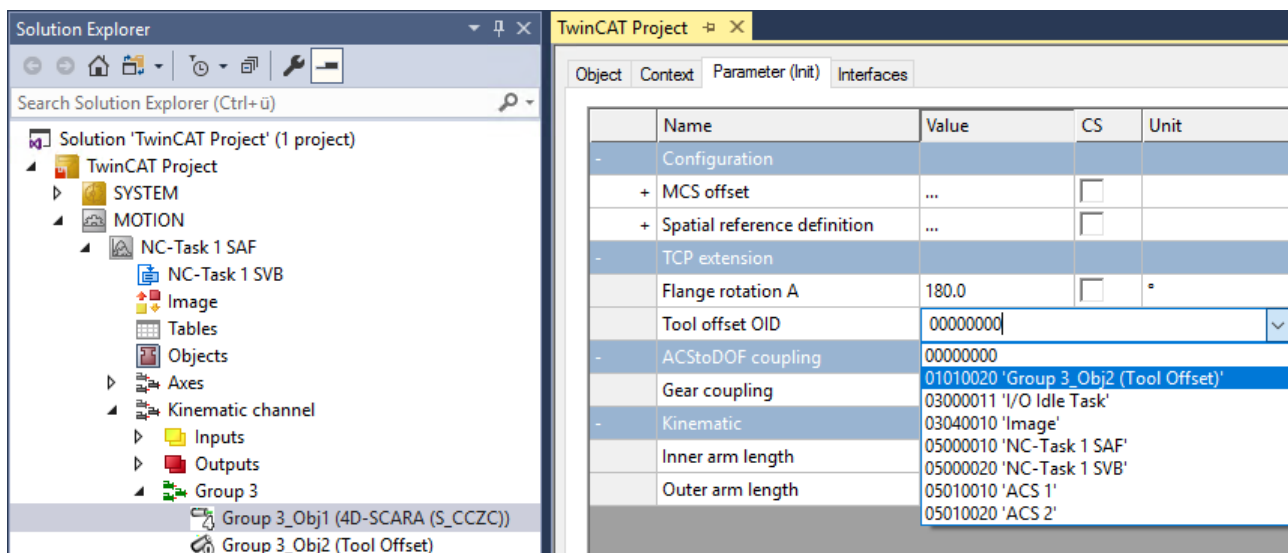
### Creating a tool

1. First, a tool has to be created under the group of the kinematics.





2. The created tool object can be allocated to the kinematics in the parameters via its tool OID.



3. The tool can now be configured via its object parameters.

## 6.29 Tool Linear

The Tool Linear describes a 1D tool, which is mounted at the flange of the kinematics. An additional simulation axis can be used for movement in tool direction. The 1D tool can be used to move the TCP at a certain distance from a workpiece.



Unless specified otherwise in the kinematics, the flange coordinate system is defined such that the orientation of the flange coordinate system matches that of the machine coordinate system, if all axes are at 0.

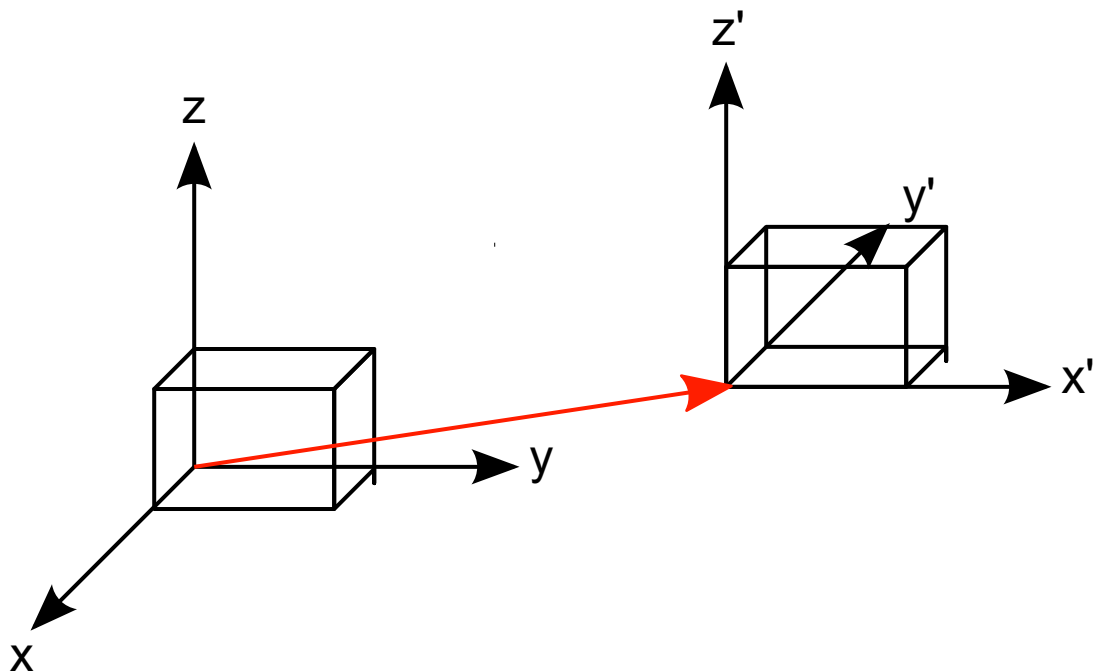
If the axis position of the additional simulation axes is 0, the TCP is at the position of the tool offset (parameter `L_init`).

Parameter	Description	Type	Unit
Length offset	Tool length	LREAL	mm
Length axis ID	Axis ID of the simulation axis; when this axis is moved, the TCP moves in the direction of the linear tool.	LREAL	

To create a tool see [Tool Offset](#) [► 59].

## 6.30 Coordinate system (Coordinate Frame)

The coordinate system supports a translation and a rotation. This transformation can be used to define a user coordinate system (UCS). For a general introduction, see [Introduction](#) [► 8].



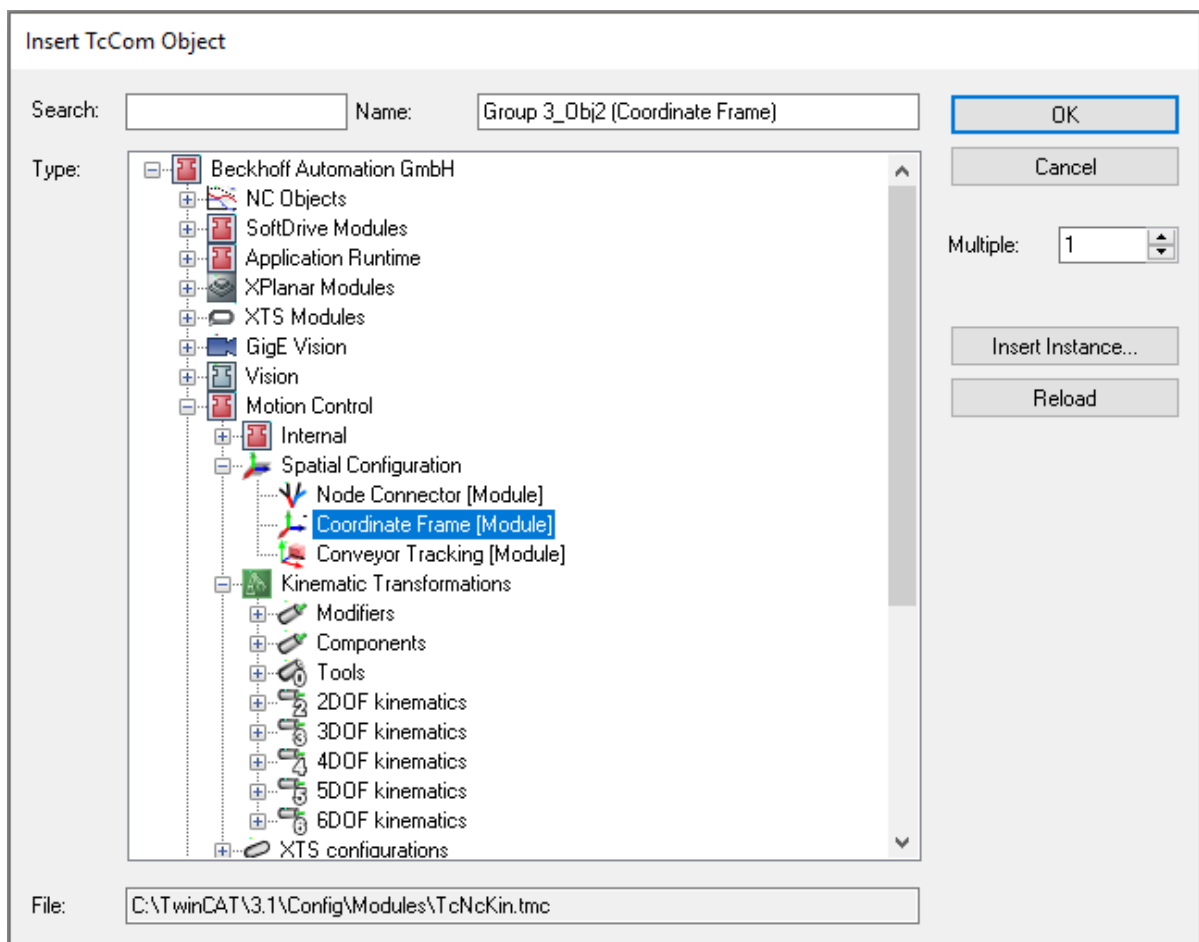
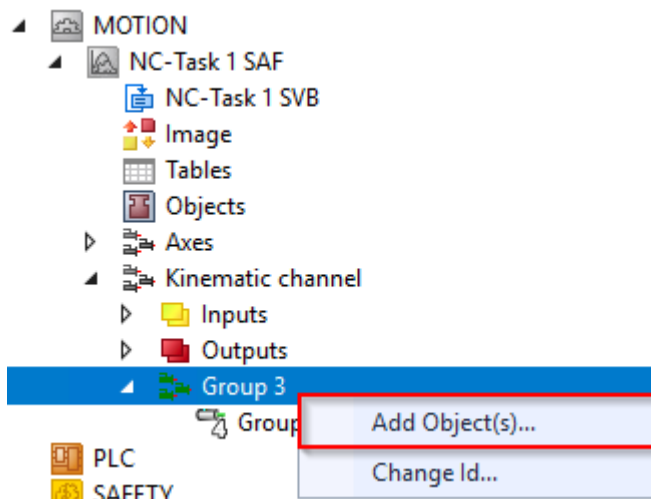
First the translation is calculated, then the rotation. The order of the rotations affects the orientation of the coordinate system. The roll/pitch/yaw rule described in DIN 9300 is used as default for the rotation sequence. The calculation sequence for the forward transformation is Z, Y', X''.

Parameter	Description	Unit
Translation X	Shift in x-direction	mm
Translation Y	Shift in y-direction	mm
Translation Z	Shift in z-direction	mm
Rotation 1	First rotation angle. The interpretation is defined by the parameter Rotation Convention.	°
Rotation 2	Second rotation angle. The interpretation is defined by the parameter Rotation Convention.	°
Rotation 3	Third rotation angle. The interpretation is defined by the parameter Rotation Convention.	°
Rotation convention	The rotation convention indicates the order of the axis rotations (parameter Rotation 1-3). The letters (X, Y, Z) from left to right indicate the order of the rotation around the	

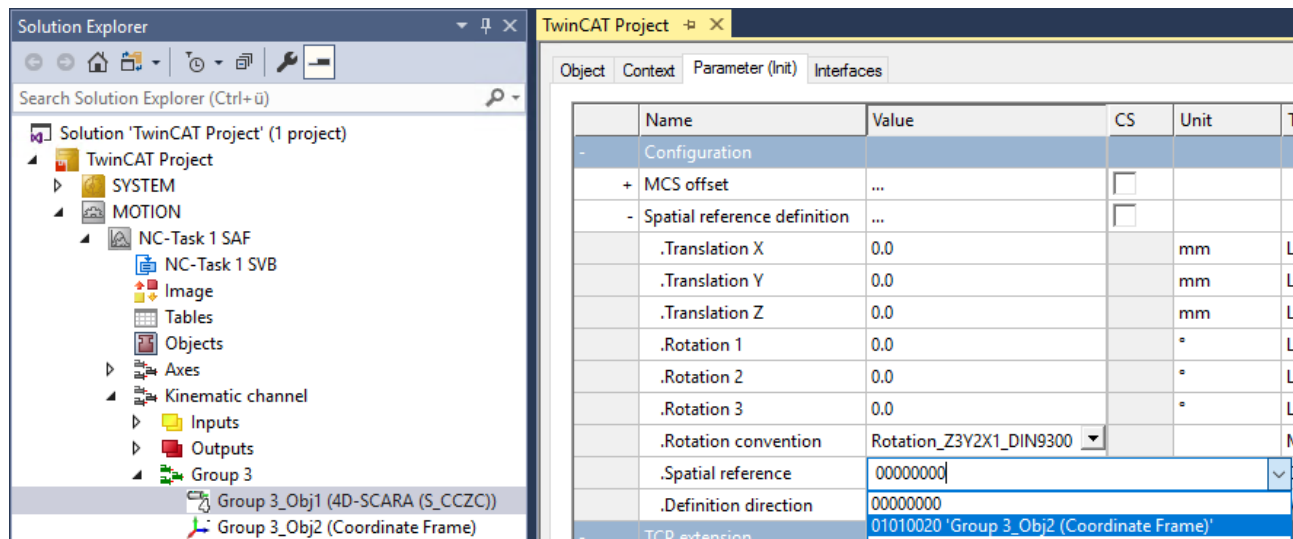
Parameter	Description	Unit
	corresponding axes. The number indicates the parameter (Rotation 1-3) for the value parameterization. The translatory shift is always performed before the rotation.	
Spatial reference	The parameter Spatial reference indicates which coordinate system is used as basis for this coordinate system. If the value is set to 0, the WCS is used as basis. To use another coordinate system as starting point for the shift, a further coordinate system object can be created. The object ID of this coordinate system can be selected via the dropdown menu.	
Definition direction	Indicates the direction in which the shift is programmed (from the perspective of the reference system or this coordinate system).	

## Creating a coordinate system

1. First create the coordinate system under the kinematic group.



2. The created coordinate system object can be defined in the kinematics via the parameter Spatial reference as origin of the MCS of the kinematics.



3. The coordinate system can now be configured via its object parameters.

## 7 User-specific transformations

In addition to the [provided transformations](#) [► 17], own kinematic transformations can also be implemented and integrated as TwinCAT C++ module.

### Restrictions

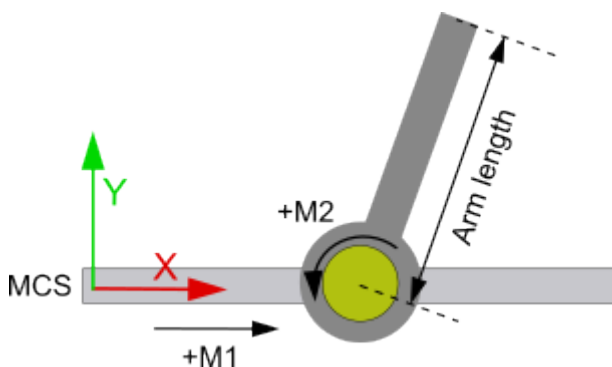
- The function blocks [FB\\_KinCalcTrafo](#) [► 82] and [FB\\_KinCalcMultiTrafo](#) [► 84] cannot be used for user-specific transformations.
- An Online Change is only allowed with TwinCAT C++ modules of the project type "TwinCAT 3 Versioned C++ Project". Otherwise, discontinuities may occur on the C++ side.

### Preparation

Beforehand, the development computer must be set up once for the development of TwinCAT C++ modules. Details can be found in the [TwinCAT C++ documentation](#).

### Development of a TwinCAT C++ module for 2D XC kinematics

The following example of 2D XC kinematics shows how to create user-specific kinematics. The kinematics is composed of a linear axis (M1), on which there is a rotation axis (M2) with an arm. The arm length should be adjustable.



### NOTICE

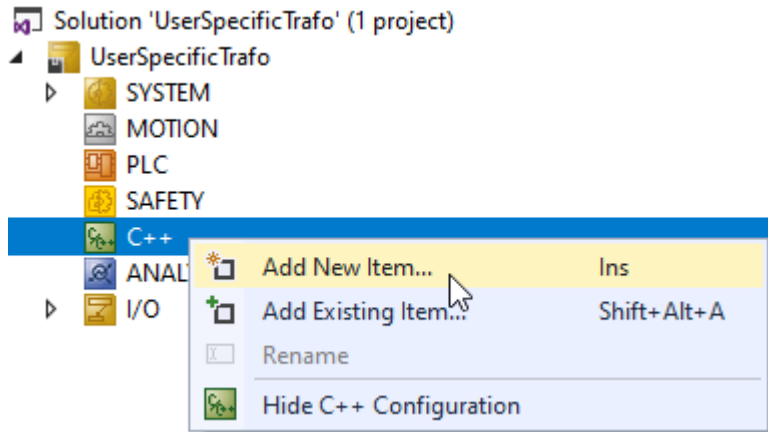
#### Online Change

An Online Change is only allowed with TwinCAT C++ modules of the project type "TwinCAT 3 Versioned C++ Project". Otherwise, discontinuities may occur on the C++ side.

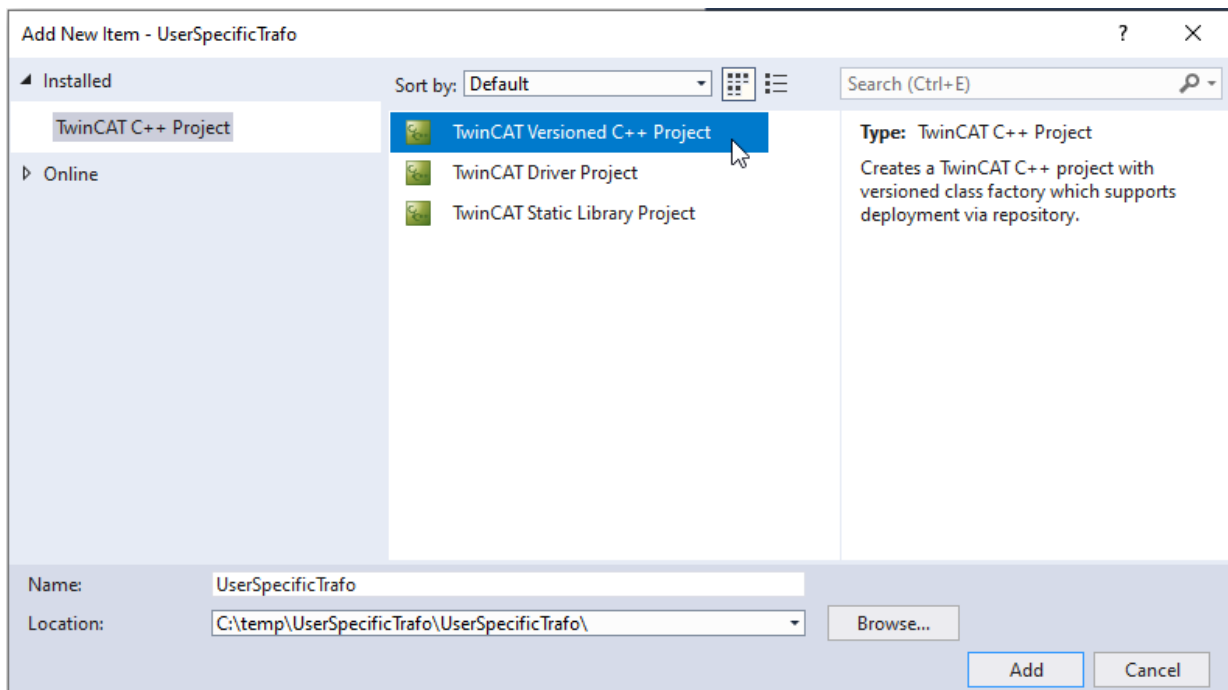
### Configuring the transformation module in the XAE

- ✓ Requirement: An empty TwinCAT project must be located on a development system that is set up for TwinCAT C++ development.

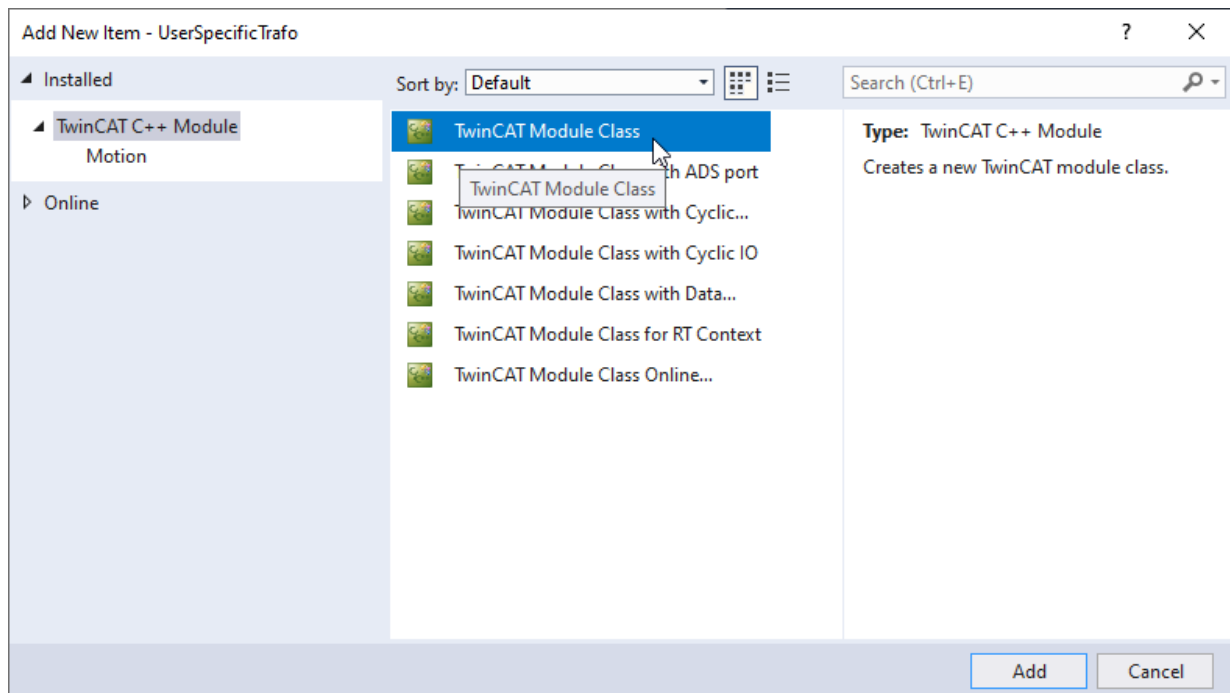
1. Add a TwinCAT C++ project to the empty TwinCAT project.  
To do this, right-click on the subtree C++ > *Add New Item...*



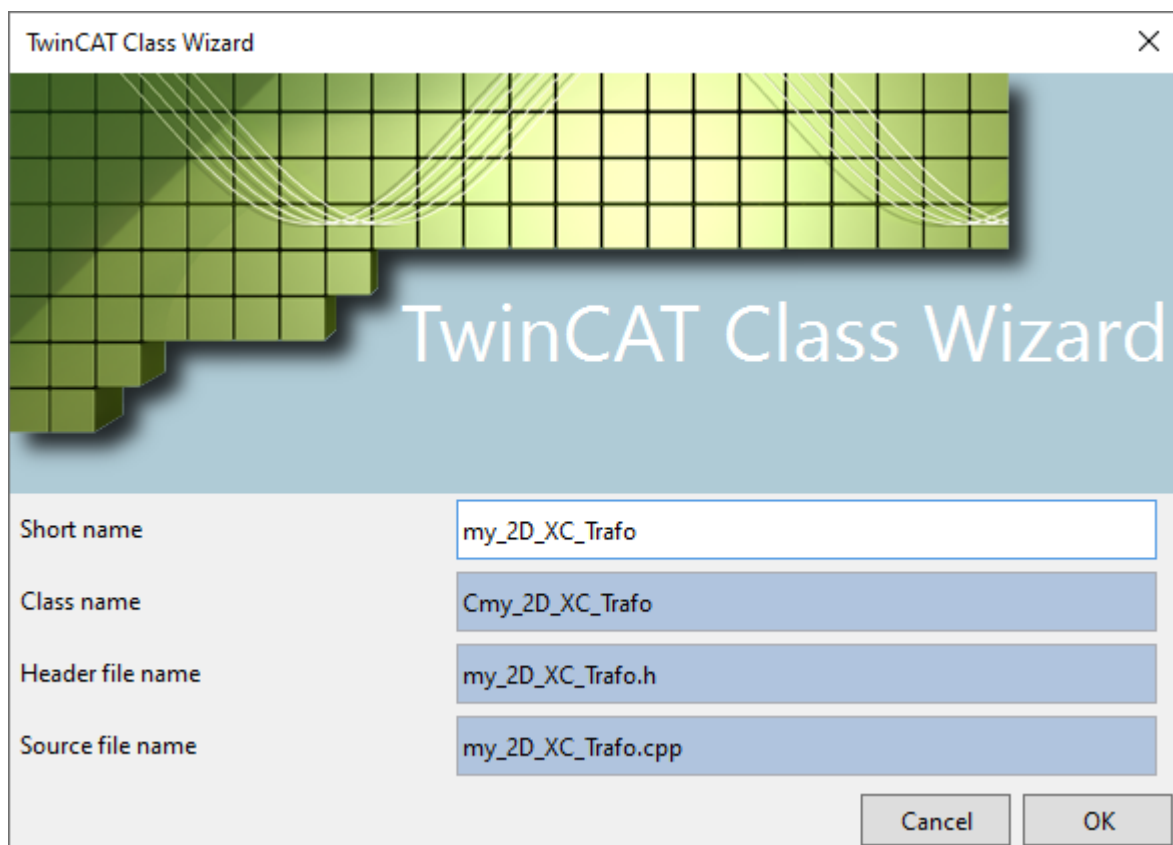
2. Select **TwinCAT Versioned C++ Project** as the project type and assign a name. Confirm the selection with the **Add** button.



3. Select **TwinCAT Module Class** as module type. Confirm the selection with the **Add** button.



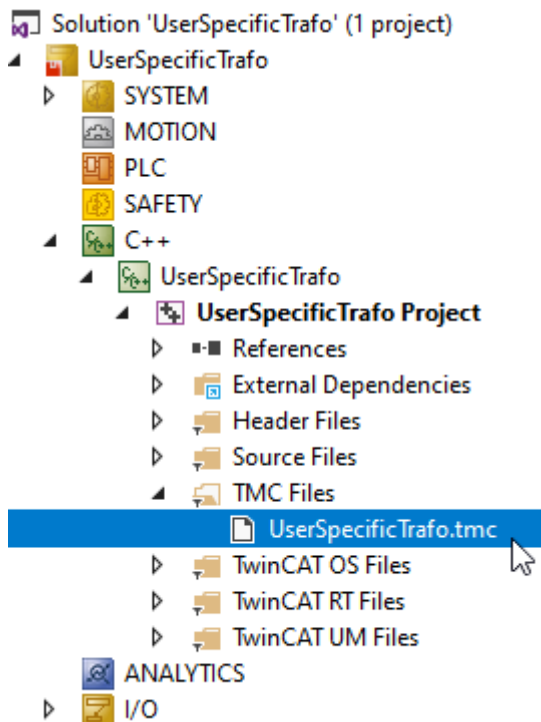
4. Give the module a name in the next step. Confirm the information with the **OK** button.



⇒ A TwinCAT C++ project with module was created.

### Adding interfaces and parameters

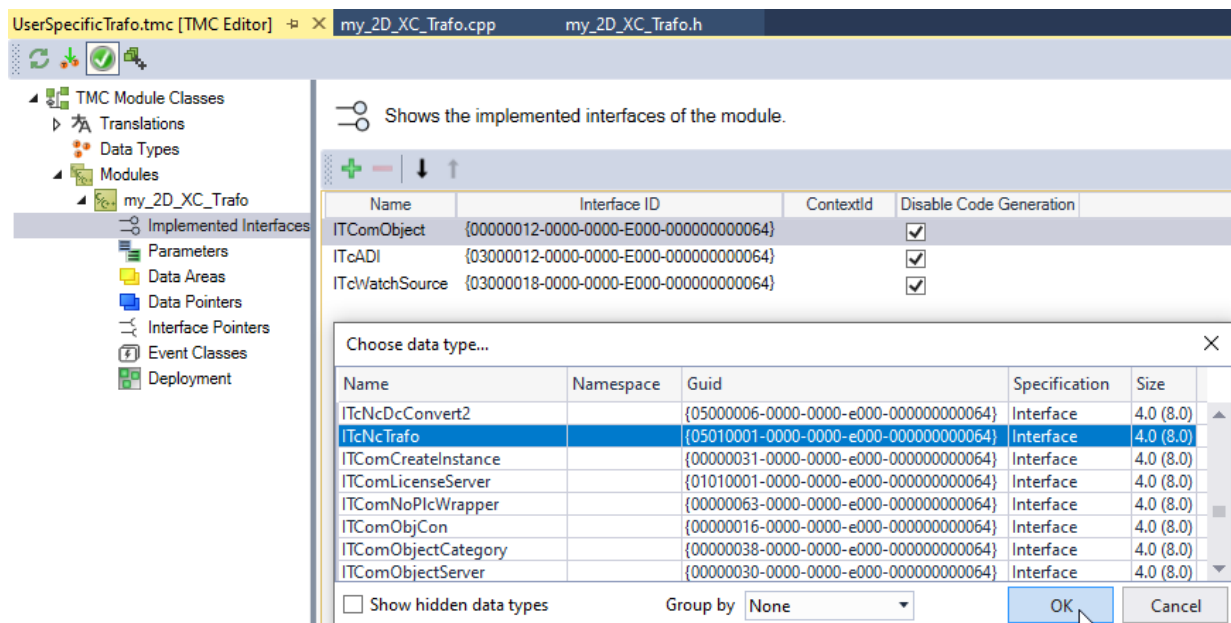
5. Double-click on the tmc file in the Solution tree to open the TMC Editor to add interfaces and parameters.



6. For a user specific transformation the interface ITcNcTrafo is required.

Open the *Choose data type...* dialog in the TMC editor via *Modules > <Module Name> > Implemented Interfaces > Add new interface (insert)*.

7. Select "ITcNcTrafo" and confirm with **OK**.

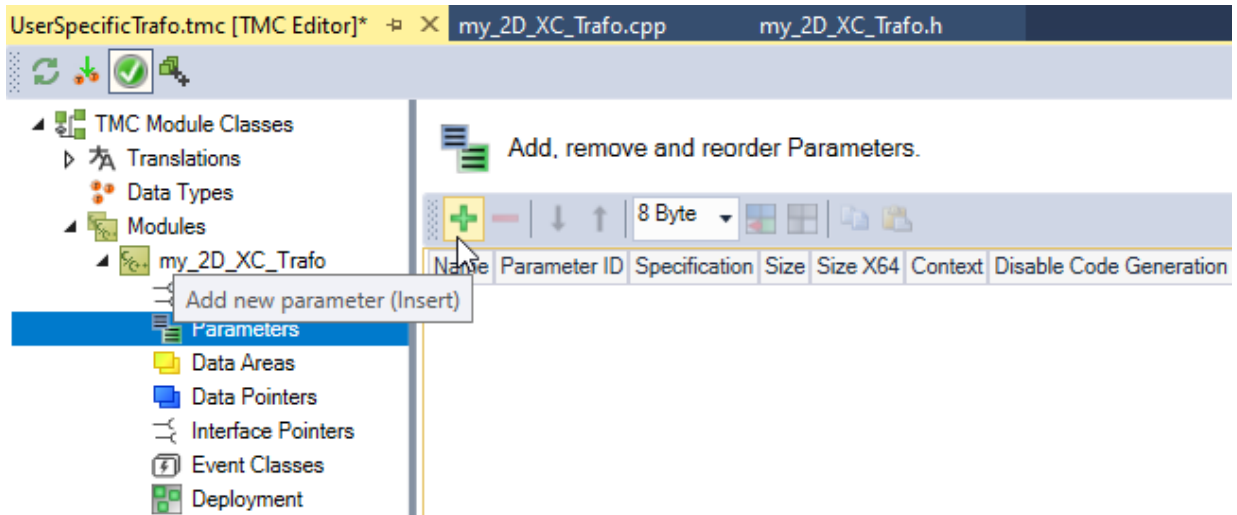


User-specific transformations can also have parameters with which the transformation can be configured. For the 2D-XC kinematics in the example, the arm length should be adjustable.

### Adjust arm length



8. Add a new parameter in the TMC editor via *Modules > <Module Name> > Parameters > Add new Parameter (Insert)*.



⇒ With a double click on the new parameter it can be configured.

9. Configure the adjustable arm length parameter as follows:

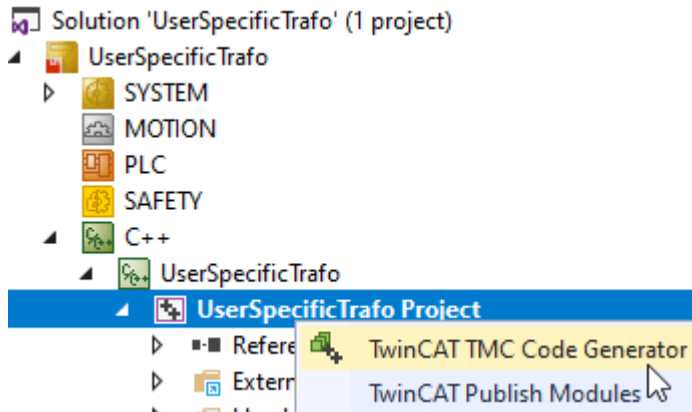
The screenshot shows the TwinCAT TMC Editor interface. On the left, a tree view displays the project structure: 'TMC Module Classes' > 'Translations' > 'Data Types' > 'Modules' > 'my\_2D\_XC\_Trafo' > 'Parameters' > 'ArmLength'. The 'ArmLength' parameter is selected. The main area on the right is titled 'Edit the properties of the parameter.' and contains several sections:

- General properties:**
  - Name:
  - Specification:
- Configure the parameter ID:**
  - Unique ID Value:
  - Constant Name:
- Choose data type:**
  - Select:
  - Description:
- Type Information:**
  - Namespace:
  - Guid:
- Optional parameter settings:**
  - Size [Bits]:  x64 specific
  - Unit:
  - Comment:
  - Context ID:
  - ☐ Create symbol
  - ☐ Disable code generation
  - ☐ Hide parameter
  - ☐ Hide sub items
  - ☐ Online parameter
  - ☐ Read-only

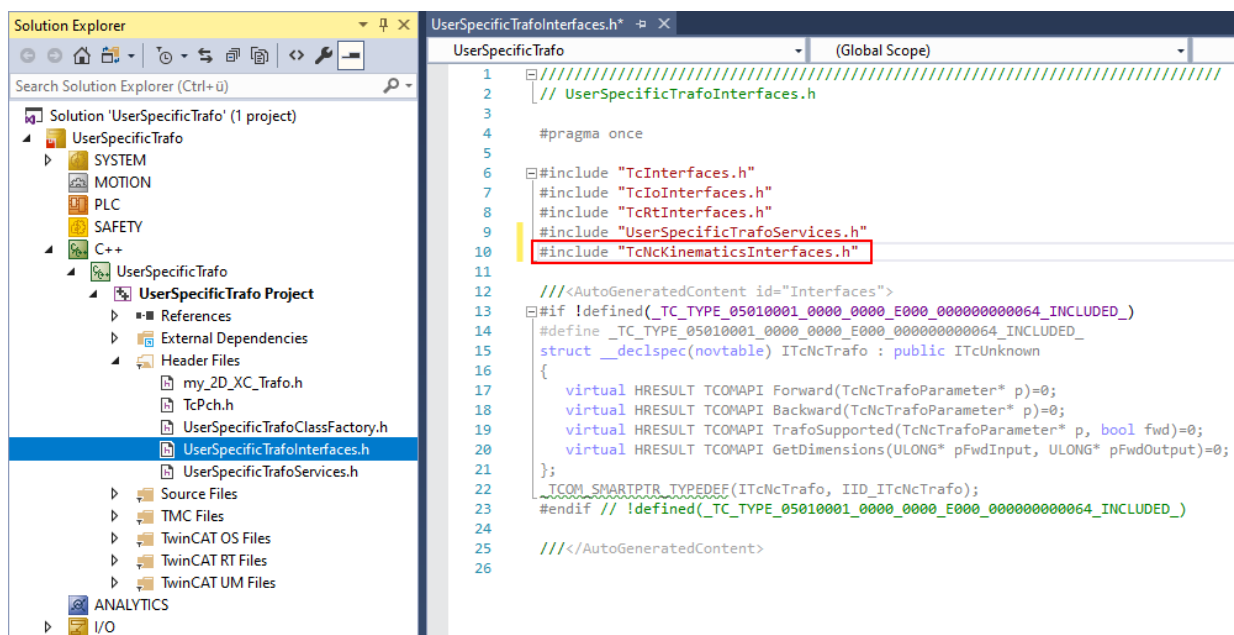
⇒ After the changes to the tmc file are completed, the TwinCAT TMC code generator can be executed.

### Run TwinCAT TMC code generator

10. Go via the TMC Editor menu or make a right click on the *C++ project > TwinCAT TMC Code Generator* in the Solution tree.



11. Add the header `TcNcKinematicsInterfaces.h` in the header `<ProjectName>Interfaces.h`.



12. The functions `Forward`, `Backward`, `TrafoSupported`, `GetDimensions` have been automatically created at `Source Files\<TrafoName>.cpp`, but do not yet contain a valid implementation. In the function `GetDimensions`, a check of the number of ACS and MCS axes has to be implemented.

```
HRESULT CMyTrafo::GetDimensions(ULONG* pFwdInput, ULONG* pFwdOutput)
{
    HRESULT hr = S_OK;

    if (pFwdInput && pFwdOutput)
    {
        *pFwdInput = 2;
        *pFwdOutput = 2;
    }
    else
    {
        hr = E_POINTER; //pointer error
    }
    return hr;
}
```

13. TwinCAT calls the function `GetDimensions` during activation. If required, you can implement additional checks in the function that should be performed during activation. For example, checking user-specific license files.
14. When building the kinematic group (FB `KinConfigGroup` [► 78]) the function `TrafoSupported` is called. In addition, it is recommended to call the function also in the functions `Forward` and `Backward`. Axis dimensions and parameter values should be checked for validity in the function.

```

HRESULT CMyTrafo::TrafoSupported(TcNcTrafoParameter* p, bool fwd)
{
    HRESULT hr = S_OK;

    if (p)
    {
        if (fwd)
        {
            if (p->dim_i != 2 || p->dim_o != 2)
            {
                // kinematics transformation error: invalid dimension
                hr = MAKE_ADS_HRESULT(NCERR_KINTRAF0_INVALIDDIM);
            }
        }
        else
        {
            if (p->dim_i != 2 || p->dim_o != 2)
            {
                // kinematics transformation error: invalid dimension
                hr = MAKE_ADS_HRESULT(NCERR_KINTRAF0_INVALIDDIM);
            }
            if (p->i[1] > m_ArmLength)
            {
                // kinematics transformation error: invalid position
                hr = MAKE_ADS_HRESULT(NCERR_KINTRAF0_INVALIDAXISPOS);
            }
        }
    }
    else
    {
        hr = E_POINTER;
    }
    return hr;
}

```

#### 15. Implement the (position) transformations in the functions `Forward` and `Backward`.

The parameters "o" and "i" are available for this purpose. Where "i" are always the input values and "o" the output values, the same applies to the `Forward` function:

```

o[0]    = Position of first MCS_axis
d_o[0]  = Velocity of first MCS_axis
dd_o[0] = Acceleration of first MCS_axis

i[0]    = Position of first ACS_axis
d_i[0]  = Velocity of first ACS_axis
dd_i[0] = Acceleration of first ACS_axis

```

It is not necessary to implement the transformations for velocity and acceleration.

At the beginning of both functions the function `TrafoSupported` should be called to check if the positions are valid.

```

HRESULT CMyTrafo::Forward(TcNcTrafoParameter* p)
{
    HRESULT hr = TrafoSupported(p, true);
    if (SUCCEEDED(hr))
    {
        if (p->i && p->o)
        {
            p->o[0] = p->i[0] + m_ArmLength*cos_((p->i[1])*PI/180);
            p->o[1] = m_ArmLength * sin_((p->i[1])*PI / 180);
        }

        if (p->d_i && p->d_o)
        {
            p->d_o[0] = p->d_i[0];
            p->d_o[1] = p->d_i[1];
        }

        if (p->dd_i && p->dd_o)
        {
            p->dd_o[0] = p->dd_i[0];
            p->dd_o[1] = p->dd_i[1];
        }
    }

    return hr;
}

```

```

HRESULT CMyTrafo::Backward(TcNcTrafoParameter* p)
{
    HRESULT hr = TrafoSupported(p, false);

    if (p->i && p->o)
    {
        p->o[1] = asin_(p->i[1] / m_ArmLength) * 180 / PI;
        p->o[0] = (p->i[0] - (cos_(p->o[1] * PI / 180) * m_ArmLength));
    }

    if (p->d_i && p->d_o)
    {
        p->d_o[0] = p->d_i[0];
        p->d_o[1] = p->d_i[1];
    }

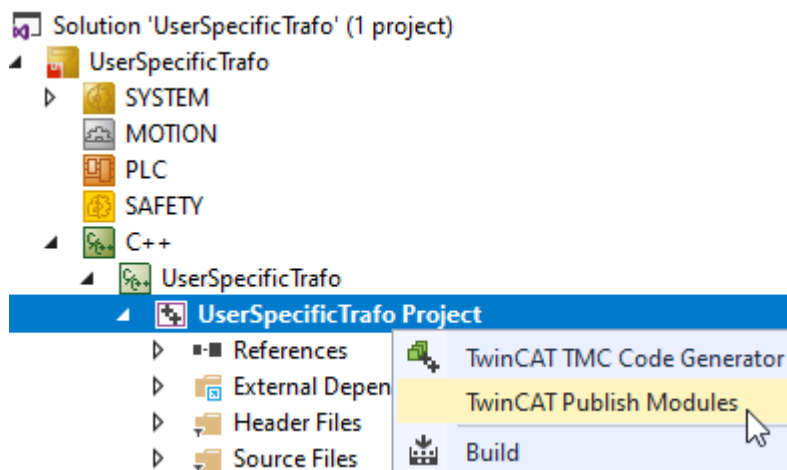
    if (p->dd_i && p->dd_o)
    {
        p->dd_o[0] = p->dd_i[0];
        p->dd_o[1] = p->dd_i[1];
    }

    return hr;
}

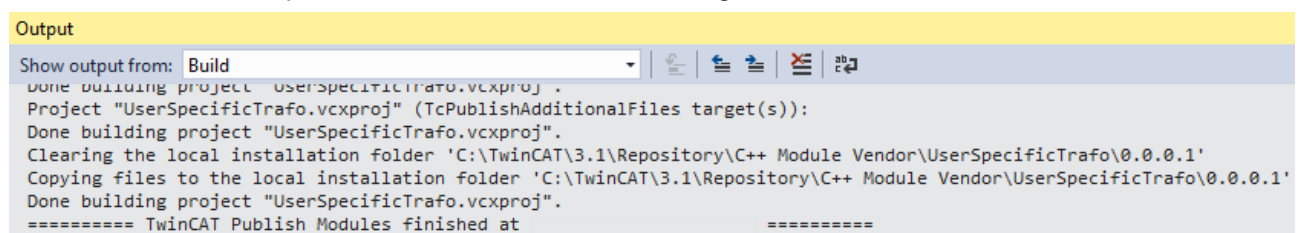
```

### Creating and publishing the modules

16. Right-click on *C++ -> UserSpecificTrafo Project* in the Solution tree and select *TwinCAT Publish Modules*.

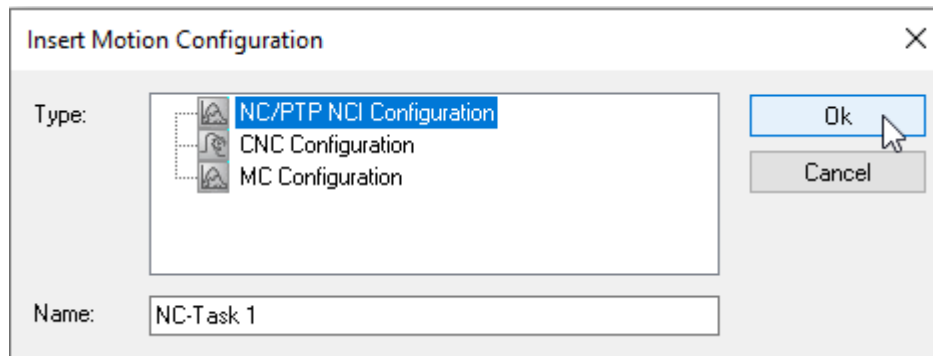
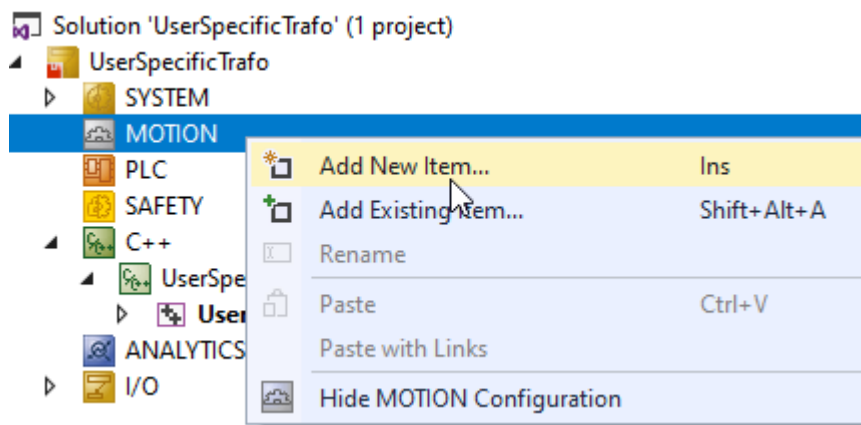


17. Check the Build output. There should be no error message here.

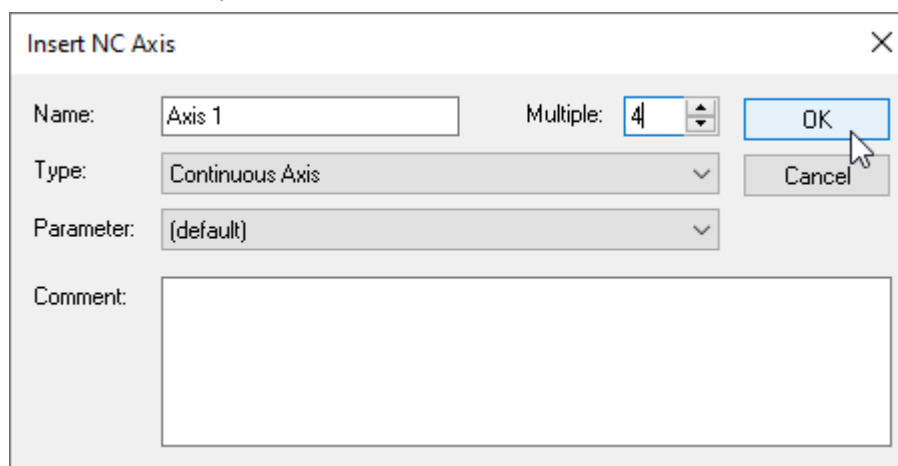


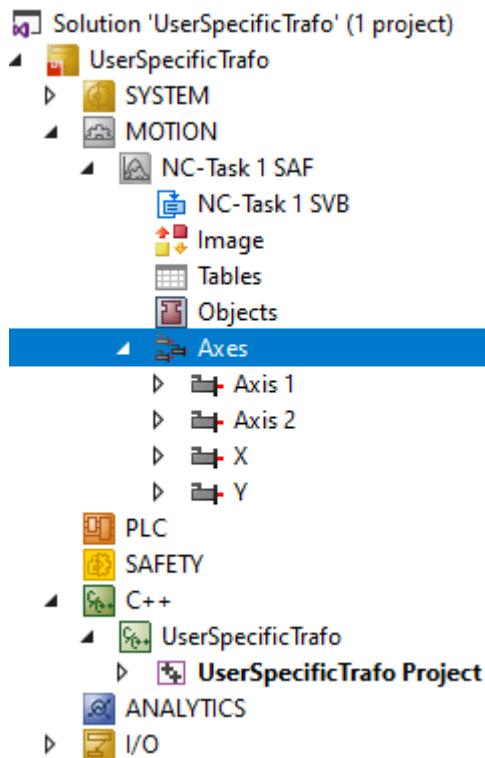
### NC/PTP configuration

18. Create an NC project in the MOTION subtree.



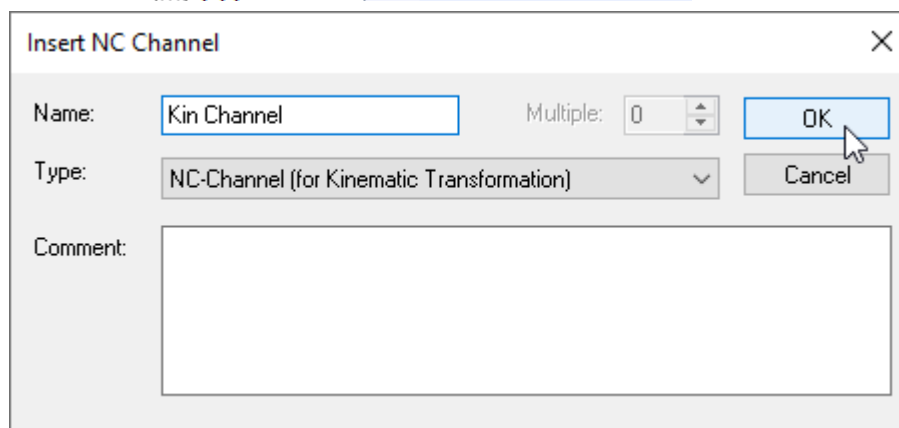
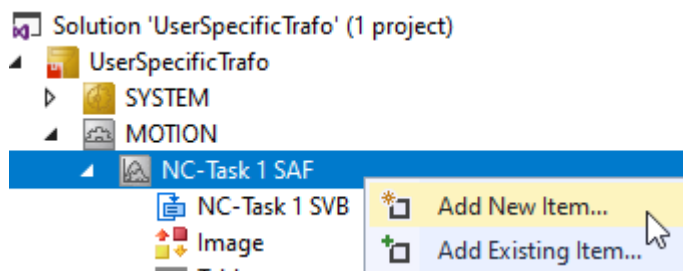
19. Create the required PTP axes.



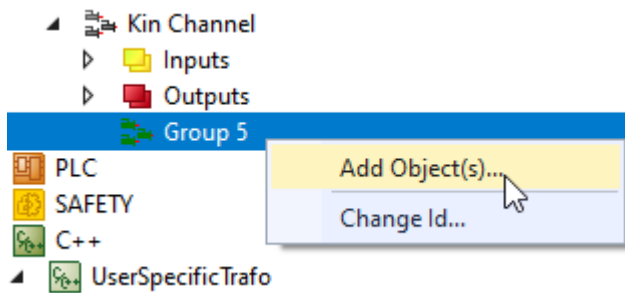


### Creating a transformation

20. Create an additional NC channel (for Kinematic Transformation) in the NC project.

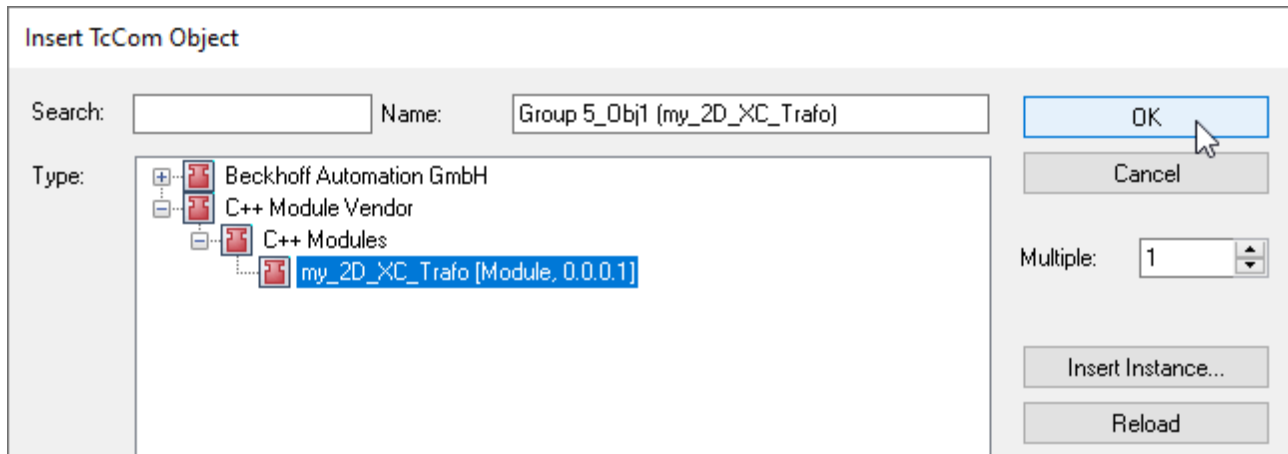


21. Then add the created transformation to the group of the kinematic channel.  
To do so, right-click on the *Group* > *Add Object(s)...*



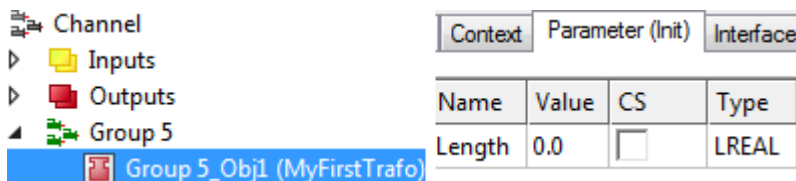
⇒ The Insert TcCom Object dialog opens.

22. Select your transformation module and confirm your selection with **OK**. If you cannot find your module, refresh the view via **Reload**.



23. Parameterize the object parameters according to the kinematics used.

⇒ The XAE configuration is now complete.



⇒ The transformation can now be activated via the PLC (see [PLC library](#) [► 77]). Define a cyclic channel interface in the PLC to address the transformation and link it to the I/O of the kinematic channel.

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



## 8 Plc Library

Function block	Description
<b>Kinematic Transformation</b>	
FB_KinConfigGroup [► 78]	Configures ACS and MCS axes according to the kinematic transformation group and enables cartesian mode or joint mode (ACS).
FB_KinResetGroup [► 80]	Resets the kinematic transformation group.
F_KinGetChnOperationState [► 93]	Reads the status of the kinematic transformation group cyclically.
F_KinGetAcsMcsAxisIds [► 94]	Reads the active ACS and MCS axes of the kinematic group.
<b>Transformation calculation</b>	
FB_KinCalcTrafo [► 82]	Calculates the Kinematic Transformation without link to the axes.
FB_KinCalcMultiTrafo [► 84]	Calculates the Kinematic Transformation for several positions.
<b>Edit parameters and coordinate systems online</b>	
FB_KinLockTrafoParam [► 89]	Locks the parameters of the kinematic transformation group, denies write access.
FB_KinUnlockTrafoParam [► 86]	Unlocks the parameters of the kinematic transformation group, enables write access.
<b>Extended rotation range</b>	
FB_KinExtendedRotationRange [► 90]	Saves and restores the rotational state of the kinematic group.
FB_KinPresetRotation [► 91]	Sets the rotational state.

### Structures and enumerations

Name	Description
ST_KinAxes [► 95]	Structure of the ACS and MCS axes, which form the kinematics
E_KinStatus [► 96]	Status of the kinematic group (enum)

Development environment	Target system	PLC libraries to include
TwinCAT 3	PC or CX (x86, x64)	Tc2_NcKinematicTransformation

### Function blocks for compatibility with existing programs



#### Function blocks for compatibility

The purpose of the function blocks listed is to ensure compatibility with existing projects. It is not advisable to use these function blocks for new projects. Instead, the equivalent function blocks shown in the table above should be used.

Function block	Description
FB_KinCheckActualStatus [► 98]	Reads the status of the kinematic transformation group acyclically

## 8.1 Function Blocks

### 8.1.1 FB\_KinConfigGroup



The function block FB\_KinConfigGroup is used to configure axes according to the kinematic transformation. These are axes for the ACS (joint) and the MCS (Cartesian). The function block takes the ACS and MCS axes defined in the **stAxesList** and configures them in the kinematic group of **stKinRefIn**.

#### Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.
bCartesianMode	BOOL	If FALSE, the ACS axes (joint) can be moved directly. If TRUE, the movement described in the MCS axes (Cartesian) is transformed into a movement of the ACS axes (joint). The ACS axes cannot be moved directly.

#### Inputs/outputs

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

Name	Type	Description
stAxesList	ST_KinAxes	Determines the ACS and MCS axes included in the configuration. See ST_KinAxes.
stKinRefIn	NCTOPLC_NCCHANNEL_REF	Determines the kinematic group of the configuration.

#### Outputs

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

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.

Name	Type	Description
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

### Sample

```

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

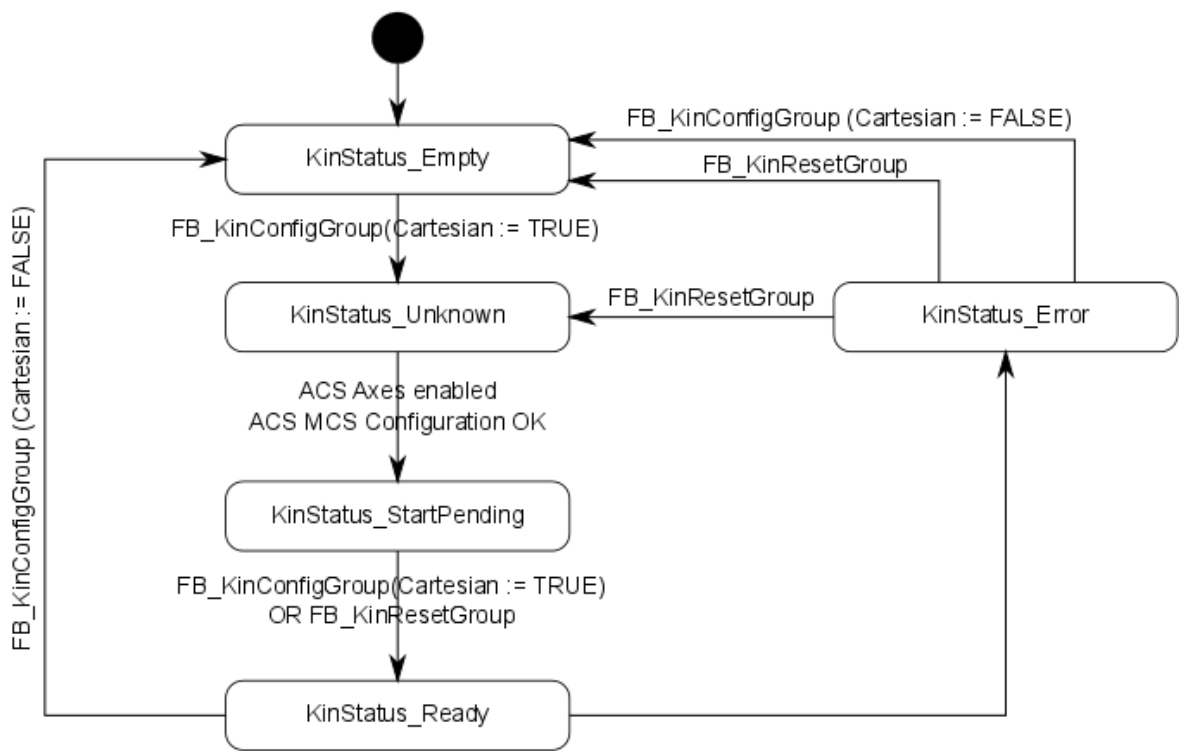
(* read the IDs from the cyclic axis interface so the axes can mapped later to the kinematic group *)
stAxesConfig.nAxisIdsAcs[1] := io_M1.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[2] := io_M2.NcToPlc.AxisId;
stAxesConfig.nAxisIdsAcs[3] := io_M3.NcToPlc.AxisId;
stAxesConfig.nAxisIdsMcs[1] := io_X.NcToPlc.AxisId;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

```

State of the kinematic group



**i** **Enable configuration**

The ACS axes must be enabled through MC\_Power, to ensure that the state can reach the value **KinStatus\_Ready**. If the ACS axes are not enabled, enable the axes and then call up FB\_KinConfigGroup or FB\_KinResetGroup.

8.1.2 FB\_KinResetGroup

**FB\_KinResetGroup**

bExecute *BOOL*

nItpChannelId *UDINT*

↔ stKinRefIn *Reference To NCTOPLC\_NCICHANNEL\_REF*

↔ stAxesList *Reference To ST\_KinAxes*

*BOOL* bBusy

*BOOL* bDone

*BOOL* bError

*UDINT* nErrorId

The kinematics group is reset with the function block FB\_KinResetGroup. All ACS and MCS axes are reset. In addition, the input *nItpChannelId* can be used for specifying the corresponding interpolation channel. The channel is reset, if the *nItpChannelId* is not 0.

When all axes are enabled and the group was in Cartesian mode, the group returns to state **KinStatus\_Ready**. If the group was not in Cartesian mode, the group returns to state **KinStatus\_Empty**. If the axes are not enabled, the group remains in state **KinStatus\_Empty**.

Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.

Name	Type	Description
nItpChannelId	UDINT	ID of the corresponding interpolation channel. If the input is not 0, the corresponding interpolation channel is reset.

### Inputs/outputs

```

VAR_IN_OUT
    stAxesList          : ST_KinAxes;
    stKinRefIn          : NCTOPLC_NCICCHANNEL_REF;
END_VAR

```

Name	Type	Description
stAxesList	ST_KinAxes	Determines the ACS and MCS axes included in the configuration. See ST_KinAxes.
stKinRefIn	NCTOPLC_NCICCHANNEL_REF	Determines the kinematic group of the configuration.

### Outputs

```

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

```

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

### Sample

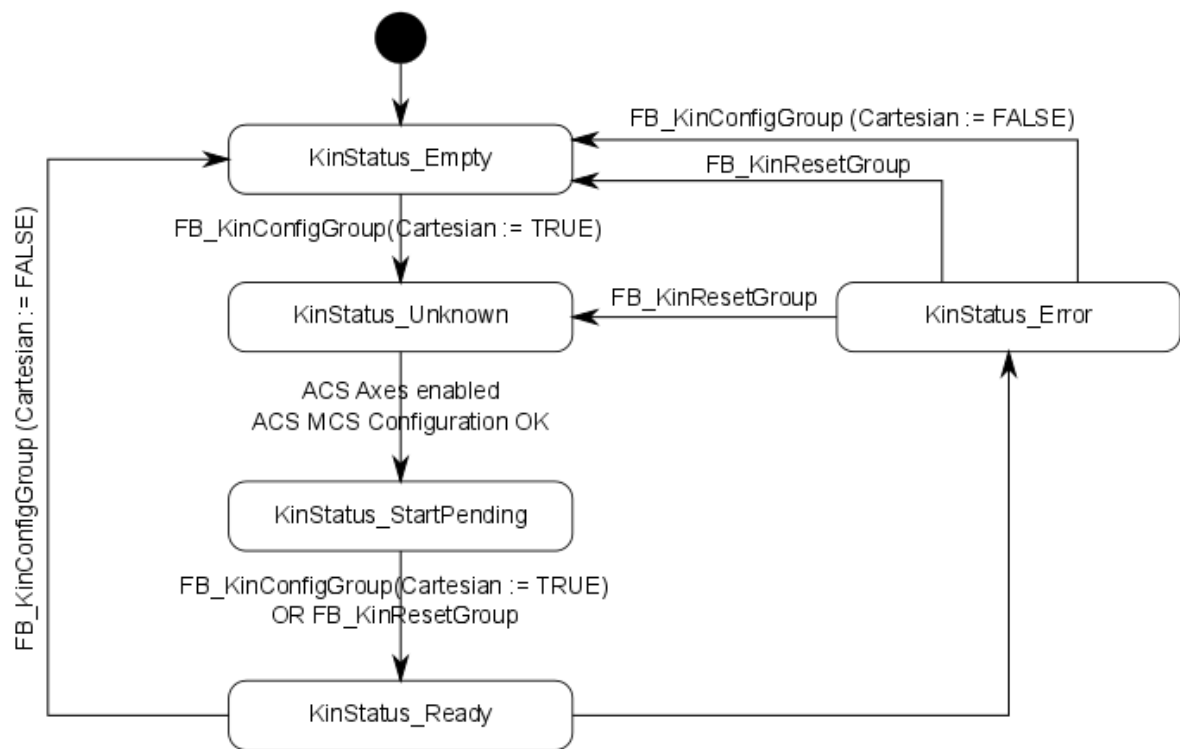
```

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

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

```

State of the kinematic group



8.1.3 FB\_KinCalcTrafo



The function block FB\_KinCalcTrafo is used to calculate the forward or backward transformation, even if no kinematic group has been created with **FB\_KinConfigGroup** [► 78].

Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.
bForward	BOOL	Determines whether the forward or backward transformation is calculated.
oidTrafo	UDINT	Object ID of the kinematic transformation object. See sample below.

## Inputs/outputs

```

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

```

Name	Type	Description
stAxesPosIn	ARRAY[1..8] OF LREAL	Array containing the input positions of the transformation. For the calculation of a forward transformation they represent the joint positions. For the calculation of a backward transformation they represent the Cartesian axis positions.
stAxesPosOut	ARRAY[1..8] OF LREAL	Array containing the result positions of the transformation. For the calculation of a forward transformation they represent the Cartesian axis positions. For the calculation of a backward transformation they represent the joint positions.
uMetaInfoIn	<a href="#">U_KinMetaInfo [► 97]</a>	In cases where different robot configurations lead to a solution, the preferred solution can be selected (see <a href="#">sample [► 83]</a> ). For kinematics in which this parameter is not required, a dummy variable can be assigned to this input.
uMetaInfoOut	<a href="#">U_KinMetaInfo [► 97]</a>	If different solutions are possible for a transformation, the solution that was found is specified. For kinematics in which this parameter is not required, a dummy variable can be assigned to this input.

## Outputs

```

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

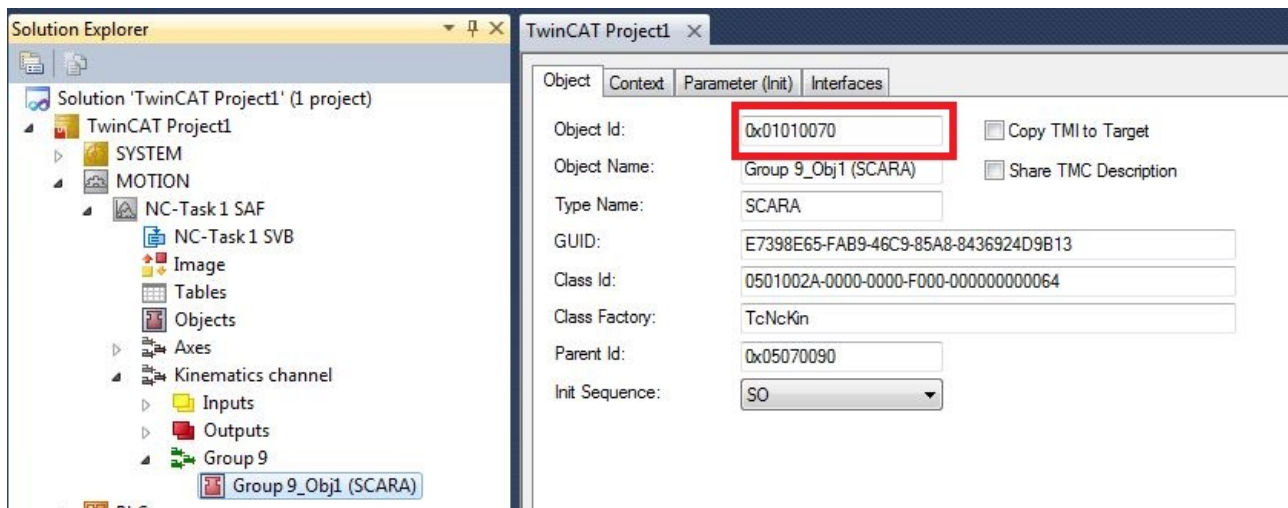
```

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

## Sample

The object ID of the transformation is shown in the transformation object under the kinematic channel.

[SCARA transformation \[► 48\]](#) - sample object ID



```

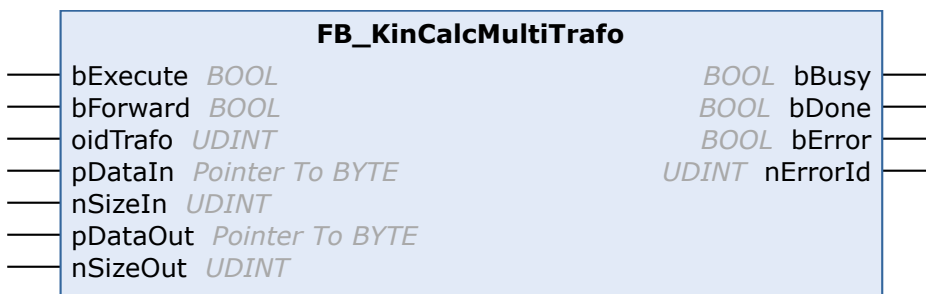
VAR
  fbKinCalcTrafo      : FB_KinCalcTrafo;
  stAxesPosIn         : ARRAY[1..8] OF LREAL;
  stAxesPosOut        : ARRAY[1..8] OF LREAL;
  bUserExecute        : BOOL;
  bUserCalcFwdTrafo   : BOOL;
  uScaraMetaInfoIn    : U_KinMetaInfo;
  uScaraMetaInfoOut   : U_KinMetaInfo;
END_VAR

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



The function block FB\_KinCalcMultiTrafo is used to calculate the forward or backward transformation for several positions, even if no kinematic group has been created with FB\_KinConfigGroup [► 78].

Alternatively, the function block FB\_KinCalcTrafo [► 82] can be used to calculate the kinematic transformations individually.

#### Inputs

```

VAR_INPUT
  bExecute      : BOOL;
  bForward      : BOOL;
  oidTrafo      : UDINT;
  pDataIn       : Pointer to BYTE;
  nSizeIn       : UDINT;

```



```

    pDataOut      : Pointer to BYTE;
    nSizeOut      : UDINT;
END_VAR

```

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.
bForward	BOOL	Determines whether the forward or backward transformation is calculated.
oidTrafo	UDINT	Object ID of the kinematic transformation object to be calculated.
pDataIn	POINTER TO BYTE	Pointer to the input data, consisting of an instance of <a href="#">ST_KinMultiTrafoHeader</a> [► 97] and an array of input positions. For the calculation of a forward transformation they represent the joint positions. For the calculation of a backward transformation they represent the Cartesian axis positions.
nSizeIn	UDINT	Size of the input data to which pDataIn points
pDataOut	POINTER TO BYTE	Pointer to the output data.
nSizeOut	UDINT	Size of the output data to which pDataOut points.

## Outputs

```

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

```

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

## Sample

### 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) );

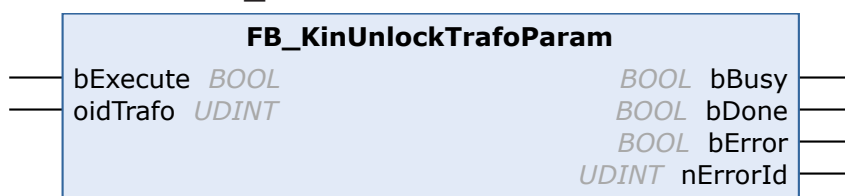
    IF NOT fbKinCalcMultiTrafo.bBusy THEN
        fbKinCalcMultiTrafo(bExecute:= FALSE, bForward:= TRUE, oidTrafo:= oidKinematic,
                            pDataIn:=ADR(stKinCalcMultiIn), nSizeIn:= SIZEOF(stKinCalcMultiIn),
                            pDataOut:=ADR(stKinCalcMultiOut), nSizeOut:=
SIZEOF(stKinCalcMultiOut) );
        nState := nState + 10;
    END_IF
END_CASE

```

## System requirements

Development environment	Target system	PLC libraries to include
Advanced Motion Pack V3.1.10.51	PC or CX (x64)	Tc2_NcKinematicTransformation

## 8.1.5 FB\_KinUnlockTrafoParam



The function block FB\_KinUnlockTrafoParam is used to unlock transformation parameters that have an influence on the position so that these can be written.

Once the kinematic parameters have been unlocked, the PLC has write access via ADSWRITE. The required index group is the object ID and the index offset is the parameter ID. The written parameters are not persistent. Parameters that have no influence on the position (e.g. torques and masses) can be written without calling FB\_KinUnlockTrafoParam.

### ⚠ CAUTION

#### Changing the parameters can lead to discontinuities.

Please note that utmost caution is required. Redefinition of kinematic parameters can lead to position setpoint step changes in the kinematic chain.

After kinematic parameters have been written, writing with FB\_LockTrafoParam can be locked again.

#### 🔧 Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.
oidTrafo	UDINT	Object ID of the kinematic transformation object. See sample below.

#### 🔧 Outputs

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

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

#### Sample

The object ID and parameter ID required for enabling a transformation parameter and for writing a corresponding new value can be read from the transformation object in the XAE.

**Object Properties:**

- Object Id: 0x01010070
- Object Name: Group 9\_Obj1 (SCARA)
- Type Name: SCARA
- GUID: E7398E65-FAB9-46C9-85A8-8436924D9B13
- Class Id: 0501002A-0000-0000-F000-000000000064
- Class Factory: TcNcKin
- Parent Id: 0x05070090
- Init Sequence: SO

**Parameter (Init) Table:**

PTCID	Name	Value	Online	CS	Unit	Type
0x05010069	MCS offset	...		<input type="checkbox"/>		
0x05010100	MCS to reference-relation	...		<input type="checkbox"/>		
0x05010020	inner arm length	380.0		<input type="checkbox"/>	mm	LREAL
0x05010021	Outer arm length	950.0		<input type="checkbox"/>	mm	LREAL
0x0501007C	Tool offset OID	00000000		<input type="checkbox"/>		OTCID

```

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



Once the transformation parameters have been modified with the aid of [FB\\_KinUnlockTrafoParam](#) [► 86], the function block FB\_KinLockTrafoParam locks the transformation parameters again, so that write access is no longer possible.

#### Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.
oidTrafo	UDINT	Object ID of the kinematic transformation object. See sample below.

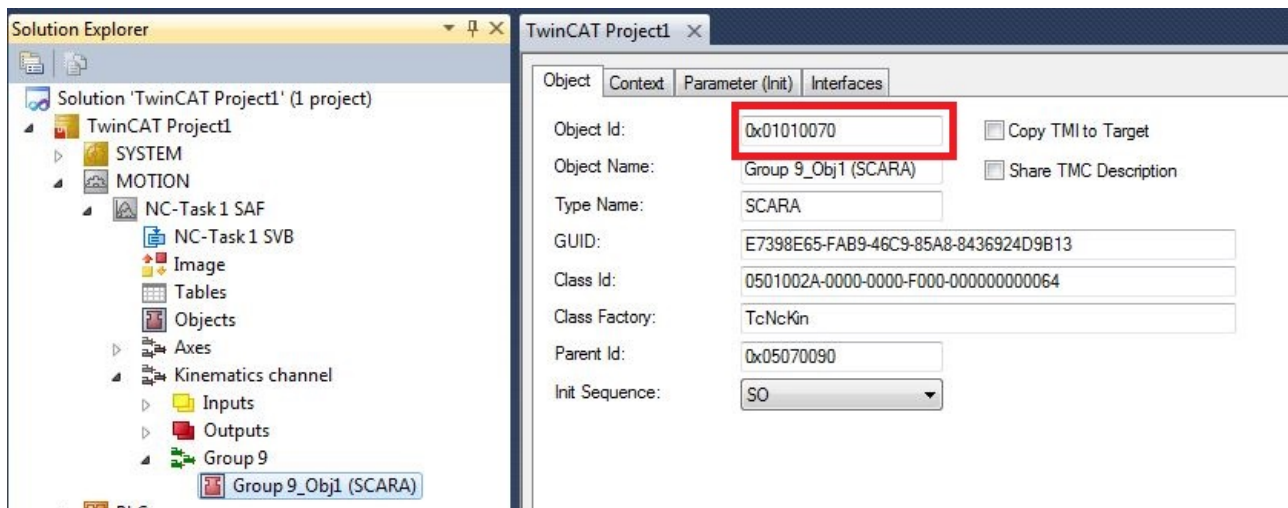
#### Outputs

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

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

#### Sample

SCARA transformation - sample object ID



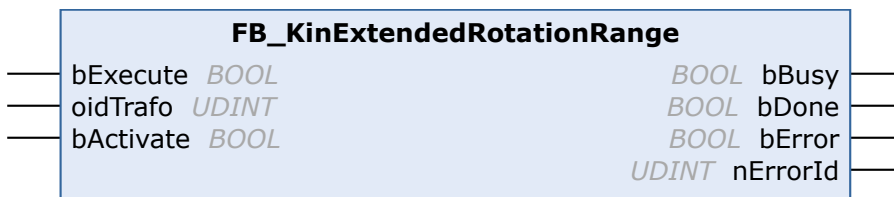
```

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



#### Extended rotation range

✓ For a unique solution the standard rotation range is limited to:

- a) Rotation1: -180 to 180 degrees,
- b) Rotation2: -90 to 90 degrees,
- c) Rotation3: -180 to 180 degrees.

⇒ In some 6-axis applications it is desirable to be able to rotate beyond this rotation range. The function blocks FB\_KinExtendedRotationRange and FB\_KinPresetRotation enable the rotational state to be extended, saved and restored beyond the default values.

The function block FB\_KinExtendedRotationRange is used to save and restore the rotation state of the kinematic group.

If the function block is executed with bActivate:=TRUE, the rotational state is saved until the kinematic group is resolved. If the kinematic group is subsequently built or reset, the saved rotational state is restored. If the rotation deviates significantly (>10.0 degrees per axis), the saved rotational state is not restored and FB\_KinConfigGroup or FB\_KinResetGroup fail with error 0x815D.

If the function block is executed with bActivate:=FALSE, the rotational state is not saved or restored (default behavior).

### Inputs

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

Name	Type	Description
bExecute	BOOL	The command is executed with a rising edge.
oidTrafo	UDINT	Object ID (OTCID) of the kinematic transformation object. See sample below.
bActivate	BOOL	If set to TRUE, the extended rotation range is activated.

### Outputs

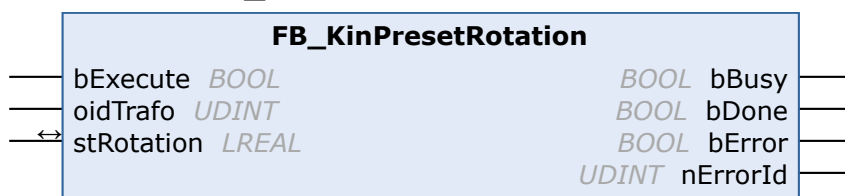
```
VAR_OUTPUT
    bBusy         : BOOL;
    bDone         : BOOL;
    bError        : BOOL;
    nErrorId      : UDINT;
END_VAR
```

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

### Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC or CX (x86 or x64)	Tc2_KinematicTransformation (V3.2.7.3 or later)

## 8.1.8 FB\_KinPresetRotation



The function block FB\_KinPresetRotation is used to set the rotational state.

The rotational state is not persistent and must be reset after a TwinCAT restart or if a path is started after an ACS axis movement (direct mode).



### Extended rotation range

✓ For a unique solution the standard rotation range is limited to:

- a) Rotation1: -180 to 180 degrees,
- b) Rotation2: -90 to 90 degrees,
- c) Rotation3: -180 to 180 degrees.

⇒ In some 6-axis applications it is desirable to be able to rotate beyond this rotation range. The function blocks *FB\_KinExtendedRotationRange* and *FB\_KinPresetRotation* enable the rotational state to be extended, saved and restored beyond the default values.



### Inputs

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

Name	Type	Description
bExecute	BOOL	The command is executed with a rising edge.
oidTrafo	UDINT	Object ID (OTCID) of the kinematic transformation object.

### Example: Equivalent rotations (same tool orientation)

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

*FB\_KinPresetRotation* must be used before *FB\_KinConfigGroup* or *FB\_KinCalcTrafo* perform the forward transformation.



To use the extended rotation range with *FB\_KinCalcTrafo(bForward:=TRUE)* without a kinematic group, the meta information *uMetaInfo.aData[4] := 1* must be set.



### Inputs/outputs

```
VAR_IN_OUT
    stRotation    : ARRAY[1..3] OF LREAL;
END_VAR
```

Name	Type	Description
stRotation	ARRAY[1..3] OF LREAL	Presetting of MCS Rotation1, Rotation2 and Rotation3



### Outputs

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

Name	Type	Description
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy



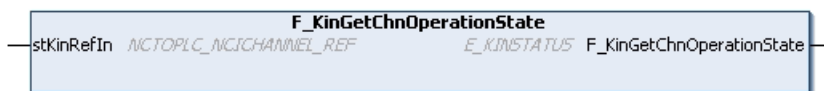
Name	Type	Description
		becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

### Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC or CX (x86 or x64)	Tc2_KinematicTransformation (V3.2.7.3 or later)

## 8.2 Functions

### 8.2.1 F\_KinGetChnOperationState



This function returns the operating state of the kinematic channel.

#### Function F\_KinGetChnOperationState : E\_KINSTATUS

```
VAR_IN_OUT
    stKinRefIn : NCTOPLC_NCCHANNEL_REF
END_VAR
```

Name	Type	Description
stKinRefIn	NCTOPLC_NCCHANNEL_REF	Determines the kinematic group of the configuration.

#### Return value

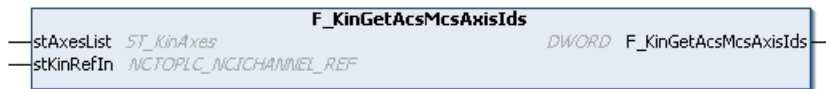
E\_KINSTATUS [► 96]: State of the kinematic channel (see below). If an invalid version of the cyclic interface is used, *KinStatus\_InvalidItfVersion* is returned.

#### Sample

```
VAR
    stKinRefIn AT %I*      : NCTOPLC_NCCHANNEL_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



This function reads the configured ACS and MCS axes of the cyclic interface. The IDs are written to `stAxesList`.

### FUNCTION F\_KinGetAcsMcsAxisIds : UDINT

```
VAR_IN_OUT
    stAxesList : ST_KinAxes;
    stKinRefIn : NCTOPLC_NCICHANNEL_REF;
END VAR
```

Name	Type	Description
stAxesList	ST_KinAxes	List of axis IDs for the axis coordinate system (ACS) and the machine coordinate system (MCS).
stKinRefIn	NCTOPLC_NCICHANNEL_REF	The structure of the cyclic channel interface between the kinematic channel and the PLC. This structure is only accessed for reading.

### Return value

**UDINT:** error code

## Sample

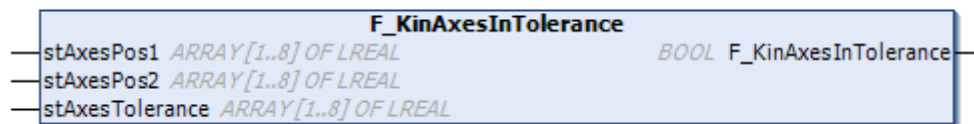
```

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



The function block **F\_KinAxesInTolerance** compares two arrays element by element.

The function returns TRUE if the difference between the respective array elements is within the expected tolerance.

## Inputs

```

VAR_INPUT
    stAxesPosIn1      : ARRAY[1..8] OF LREAL;
    stAxesPosIn2      : ARRAY[1..8] OF LREAL;
    stAxesTolerance    : ARRAY[1..8] OF LREAL;
END VAR

```

Name	Type	Description
stAxesPosIn1	ARRAY[1..8] OF LREAL	First array. This is compared with the second array.
stAxesPosIn2	ARRAY[1..8] OF LREAL	Second array. This is compared with the first array.
stAxesTolerance	ARRAY[1..8] OF LREAL	Contains the tolerance for each array element to be compared.

### Return value

**BOOL:** The function returns TRUE if the difference between the respective array elements is within the expected tolerance.

### Requirements

Development environment	Target platform	PLC libraries to include
TwinCAT V3.1.4024.7 Advanced Motion Pack V3.1.10.1	PC or CX (x86 or x64)	Tc2_KinematicTransformation (V3.2.7.3 or later)

## 8.3 Datatypes

### 8.3.1 ST\_KinAxes

This structure defines the axes, which form a kinematic system.

```

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

```

Name	Type	Description
nAxisIdsMcs	ARRAY[1..8] OF DWORD	List of axis IDs of the axes that form the MCS. Usually, the first three array elements specify the Cartesian axes (X,Y,Z), the subsequent array elements specify the rotational axes.
nAxisIdsAcs	ARRAY[1..8] OF DWORD	List of axis IDs of the axes that form the ACS.

### Sample

```

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;

```

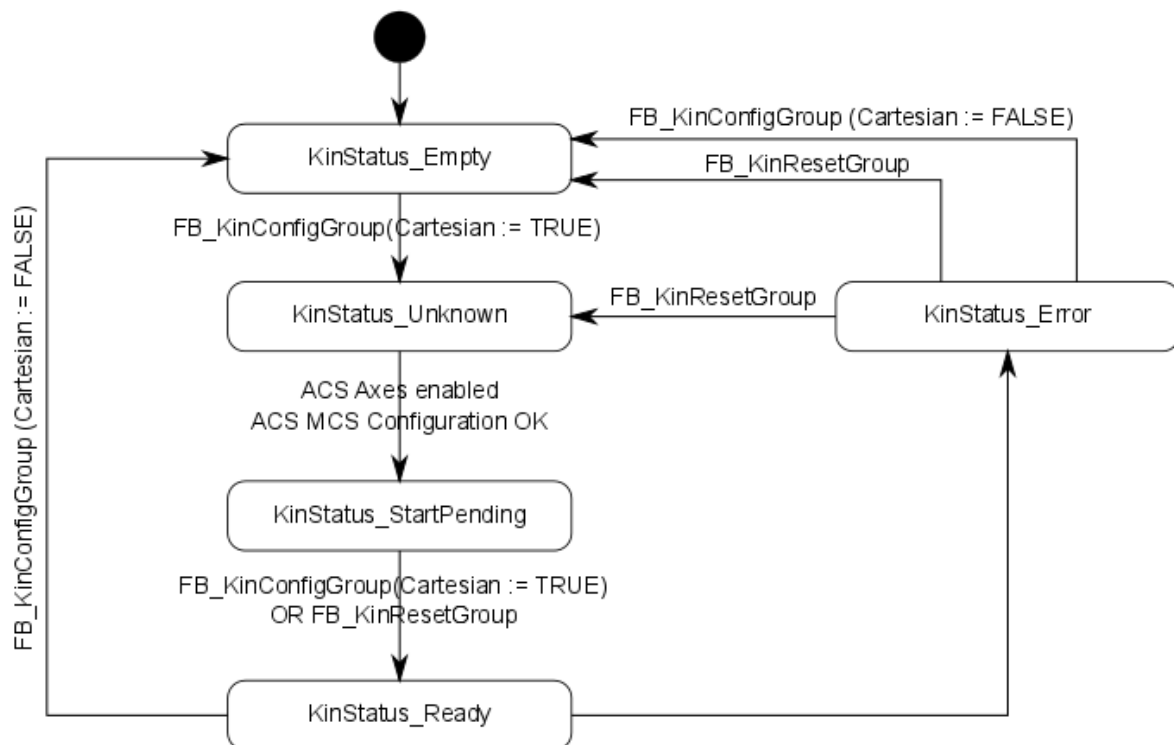
## 8.3.2 E\_KinStatus

This enumeration defines the state of the kinematic group.

```

TYPE E_KinStatus :
(
  KinStatus_Error,
  KinStatus_Empty,
  KinStatus_Unknown,
  KinStatus_StartPending,
  KinStatus_Ready,
  KinStatus_InvalidItfVersion := 16#4000
);
END_TYPE

```



Name	Description
KinStatus_Empty	ACS axes can be moved. No transformation enabled.
KinStatus_Ready	MCS axes can be moved. Transformation active.
KinStatus_InvalidItfVersion	A function or function block is not supported by this version of the cyclic channel interface. An update is required in order to be able to use the function.



### Enable configuration

The ACS axes must be enabled by MC\_Power so that the status can reach the value **KinStatus\_Ready**.

## 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 for defining the arm position for 4D-SCARA [► 48] kinematics.

```

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

```

### 8.3.3.3 ST\_KinMultiTrafoHeader

Header of the input data structure for the function block FB\_KinCalcMultiTrafo [► 84].

```

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

```

Name	Type	Description
nColumnsIn	UDINT	Real number of columns in the input array.
nColumnsOut	UDINT	Real number of columns in the output array.
nLines	UDINT	Number of lines to be calculated, may be less than the actual number of lines declared.
bGetMetaInfo	BOOL	If TRUE, MetaInfos are also output. Accordingly, memory of the source data structure must be available for this purpose.
uMetaInfo	<u>U_KinMetaInfo</u> [► 97]	

### 8.3.3.4 U\_KinMetaInfo

```

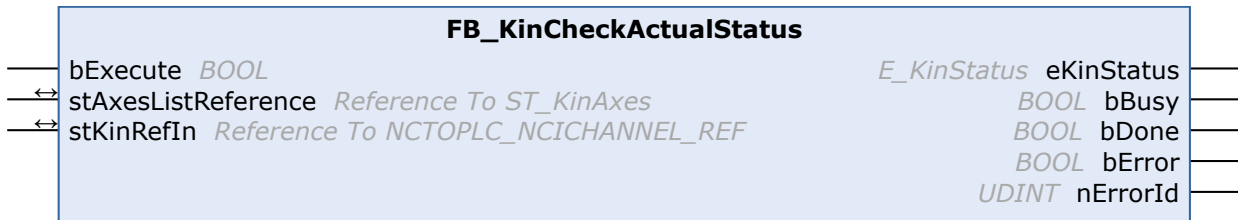
Type U_KinMetaInfo
UNION
    aData      : ARRAY[1..4] OF UDINT;
    eScara     : E_KinMetaInfoScara;
    e5dType1   : E_KinMetaInfo5DType1;
END_UNION
END_TYPE

```

Name	Type	Description
aData	ARRAY[1..4] OF UDINT	
eScara	<u>E_KinMetaInfoScara</u> [► 97]	Definition of the arm position with <u>4D-SCARA</u> [► 48] kinematics.
e5dType1	<u>E_KinMetaInfo5DType1</u> [► 96]	

## 8.4 Legacy

### 8.4.1 FB\_KinCheckActualStatus



#### Outdated version

The sole purpose of the function block is to ensure compatibility with existing projects. For new projects, please use [F\\_KinGetChnOperationState \[► 93\]](#). This function block needs more than one PLC cycle to read the status of the kinematic channel. To obtain it for each cycle, use [F\\_KinGetChnOperationState \[► 93\]](#).

The function block FB\_KinCheckActualStatus returns the status of the kinematic channel.

#### Inputs

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

Name	Type	Description
bExecute	BOOL	The command is triggered by a rising edge at this input.

#### Inputs/outputs

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

Name	Type	Description
stAxesList	ST_KinAxes	Determines the ACS and MCS axes included in the configuration. See ST_KinAxes.
stKinRefIn	NCTOPLC_NCICCHANNEL_REF	Determines the kinematic group of the configuration.

#### Outputs

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

Name	Type	Description
eKinStatus	E_KINSTATUS	Returns the status of the kinematic channel. See <a href="#">E_KINSTATUS [► 96]</a> .
bBusy	BOOL	The output becomes TRUE when the command is started with bExecute and remains TRUE as long as the function block executes the command. While bBusy is TRUE, no new command is accepted at the inputs. If bBusy becomes FALSE again, the function block is ready for a new command. At the same time one of the outputs bDone or bError is set.

Name	Type	Description
bDone	BOOL	The output becomes TRUE when the command was executed successfully.
bError	BOOL	The output <i>bError</i> is set to TRUE, if an error occurred during the command execution.
nErrorId	UDINT	Contains the command-specific error code of the most recently executed command. Details of the error code can be found in the ADS error documentation or in the NC error documentation (error codes from 0x4000).

### Sample

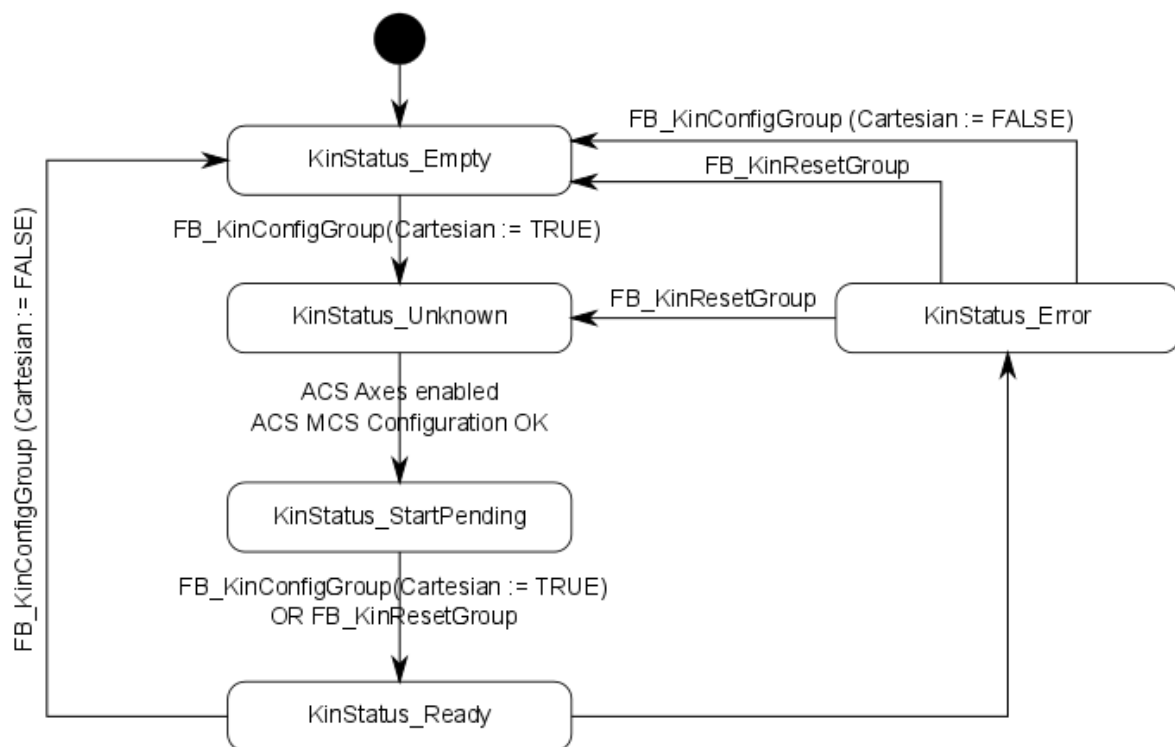
```

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

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

```

### State of the kinematic group



## **Trademark statements**

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.



More Information:  
**[www.beckhoff.com/tf5110](http://www.beckhoff.com/tf5110)**

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

